Iterative Critique-and-Routing Controller for Multi-Agent Systems with Heterogeneous LLMs

Wenzhi Fang[1], **Liangqi Yuan**[1], **Guangchen Lan**[1],
Dong-Jun Han[2], **Christopher G. Brinton**[1]

Abstract

Multi-agent large language model (LLM) systems often rely on a controller to coordinate a pool of heterogeneous models, yet existing controllers are typically limited to one-shot routing: they select a model once and return its output directly. Such routing-only designs provide no mechanism to critique intermediate drafts or support iterative refinement. To address this limitation, we propose a *critique-and-routing controller* that casts multi-agent coordination as a sequential decision problem. At each turn, the controller evaluates the current draft, decides whether to stop or continue, and, if needed, selects the next agent for further refinement. We formulate this process as a finite-horizon Markov Decision Process (MDP) with explicit agent-utilization constraints, design a composite reward for controller decisions across turns, and optimize the controller via policy gradients under a Lagrangian-relaxed objective. Extensive experiments across multiple heterogeneous multi-agent systems and seven reasoning benchmarks show that our method consistently outperforms state-of-the-art baselines and substantially narrows the gap to the strongest agent, while using it for fewer than 25% of total calls.

1 Introduction

Multi-agent large language model (LLM) systems are becoming increasingly popular in practice [34, 16, 35, 18]. A representative deployment pattern is an *agent pool* coordinated by a controller that selects among heterogeneous agents to balance quality and cost [4]. Most existing coordination mechanisms treat the controller primarily as a one-shot router, selecting a single model and returning the resulting response to the user [3, 20]. However, many queries, especially those requiring sophisticated reasoning, are hard to solve in one turn: an initial draft may be incomplete or incorrect, and feedback can enable substantial refinement in subsequent turns [22]. Yet one-shot routing-based coordination offers no such feedback loop: once an agent is selected, its response is returned without an opportunity for correction.

To mitigate the limitations of one-shot routing, recent work has explored multi-turn coordination from different perspectives. Prior work [24] proposes a contextual-bandit framework where the router updates its selection strategy over rounds using user feedback. Router-R1 [37] trains an LLM router via reinforcement learning (RL) to perform multi-turn routing and aggregation, but primarily targets retrieval-style settings in which a query is decomposed into sub-queries and routed for information collection. Researchers in [11] study cost-controllable coordination by conditioning the controller on externally provided budget modes, enabling different routing behavior under different budgets. Despite this progress, these approaches largely emphasize *routing* and underexplore controller-to-agent *interactive critique*, as well as explicit workload balancing across heterogeneous agents.

Based on these observations, we pose the following two questions:

Q1: Can we design a controller that jointly (i) routes to an appropriate agent, (ii) critiques and verifies intermediate drafts, and (iii) decides when to stop refining?

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Q2: Can we optimize such a controller with a principled objective that jointly considers final-answer quality and system-level constraints such as balanced utilization across agents?

Challenges. Answering these questions raises three key challenges. First, *Joint routing and verification:* Unlike one-shot routers that only select a model, the controller must simultaneously assess response draft quality, decide whether to stop or continue refining, and select the next agent when refinement is needed. These decisions are tightly coupled: the verification shapes the next generation, and the stopping decision depends on the controller's assessment of the current draft. Second, *Trajectory-level credit assignment:* in a multi-turn interaction process, routing and verification decisions made at different turns of the same trajectory can contribute unequally to the final answer. However, commonly used outcome-based rewards blur these differences, obscuring which turns are primarily responsible for success or failure. Third, *Quality-utilization trade-off:* optimizing only for answer quality encourages overuse of the strongest, often the most expensive, agent, undermining the purpose of maintaining a heterogeneous agent pool. A practical controller should therefore be able to improve response quality through multi-turn interaction while balancing utilization across agents.

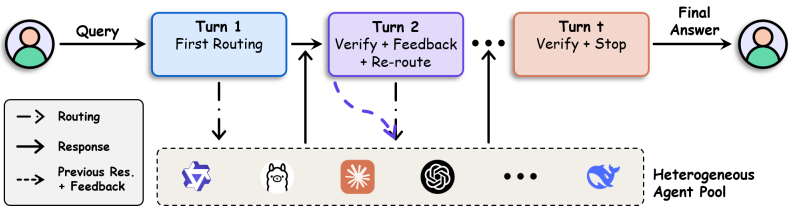


Figure 1: Controller interacts with a heterogeneous pool of agents over multiple rounds, evaluating the previous response from an agent and deciding whether to exit or refine the response.

1.1 Contributions

To address these challenges, we study an iterative critique-and-routing controller and develop an RL-based framework for multi-turn coordination, which formulates multi-agent routing as sequential interaction control. The controller serves as both critic/verifier and router: at each turn, it evaluates the draft response produced by the agent selected in the previous turn and decides whether to stop or invoke another agent for further refinement. We formalize this as a finite-horizon Markov Decision Process (MDP) with explicit agent utilization constraints. Our main contributions are as follows:

- **Sequential interaction formulation.** We propose a multi-turn controller with dual roles as critic and router, and formulate multi-agent coordination as a finite-horizon MDP for cumulative reward maximization. In this formulation, each controller action jointly specifies an evaluation of the previous response, a stop/continue decision, and the next agent to invoke. To prevent over-reliance on the strongest agent, we further impose explicit agent-utilization constraints.
- **Reward and utilization-aware objective.** We introduce a composite rule-based reward that evaluates *both* critique and routing decisions over trajectories. To facilitate optimization, we transform the constrained problem into an unconstrained one via Lagrangian relaxation, yielding a modified reward-maximization objective that can be optimized with standard policy gradient methods.
- **Empirical evaluation.** Through extensive experiments on diverse heterogeneous multi-agent systems with different controllers, agents, and benchmarks, we demonstrate that our approach significantly outperforms baselines while maintaining comparable usage of the strongest agent.

1.2 Related Work

Multi-agent collaboration architectures. Prior work on multi-agent LLM systems can be broadly categorized into decentralized *role-based* collaboration and centralized *controller-agent* orchestration. In role-based systems, agents are assigned distinct roles and interact through predefined workflows [13, 35, 9, 26]. A closely related variant is *cascaded* specialization, in which different agents are invoked in a fixed sequence for generation, critique, and refinement, as in actor-critic style collaboration [5] and generate-verify-refine pipelines [22]. Recent work also considers query decomposition and routing, where complex tasks are decomposed into subproblems that are assigned to specialized

agents [30, 15, 27]. Another line of work studies centralized *controller-agent* systems with a star-topology architecture, where a single controller coordinates a heterogeneous pool of experts and decides which agent to invoke for each query [1, 3, 23, 4, 11]. Our work focuses on this latter setting, and we discuss the most relevant controller design literature below.

Centralized controller-agent orchestration. Within controller-agent systems, coordination strategies range from lightweight classifier-based routing to reasoning-driven decision-making. Early approaches formulate coordination as a one-shot model-selection problem, using classification [20, 4] or representation-learning-based routers [21, 2] to dispatch each query to a single agent. More recently, the authors in [24] cast multi-agent coordination as a contextual bandit problem, accounting for multi-turn user-agent interaction and enabling the response quality to improve over time from observed human feedback. Router-R1 [37] and CoRL [11] further adopt reasoning-capable LLMs as controllers and apply RL to cultivate reasoning-aware routing policies. Despite this progress, existing methods largely treat coordination as isolated routing decisions, rather than as a sequential interaction process. Moreover, they typically lack explicit utilization constraints, leaving balanced and efficient use of heterogeneous agents underexplored.

2 Problem Background

2.1 Controller Design for Multi-Agent Systems

We consider a multi-agent LLM system with a heterogeneous agent pool $\mathcal{M} = \{\pi_1, \pi_2, \ldots, \pi_K\}$ as shown in Figure 1, where agents differ in capability and cost. A controller coordinates this pool to respond to incoming queries. Unlike one-shot routing, where the controller selects a single model and directly returns its response without verification, we study a setting in which the controller coordinates the model pool over multiple turns by selecting an agent, evaluating/verifying the selected agent's response, and deciding whether to continue or terminate the interaction. The problem is therefore not merely to identify the single best model for a query in one turn, which is often difficult in practice, but to learn a controller policy that maximizes final task performance over multi-turn interactions while satisfying system-level usage constraints across agents. The controller should allocate calls across heterogeneous agents so as to improve answer quality through sequential interaction, without collapsing to always invoking the strongest or most expensive model.

2.2 Aren't the Existing Solutions Good Enough?

Although existing works have explored controller design in multi-agent systems from different perspectives, they still suffer from several limitations. Below, we discuss the two most relevant lines of work and describe their limitations in detail.

Router-R1. Researchers in [37] introduce an RL-based controller for multi-agent LLM systems in information retrieval settings. Similar to Search-R1 [12], their approach decomposes a complex query into sub-queries and routes them to agents sequentially for information collection. However, this design is primarily tailored to retrieval-oriented scenarios, where queries are decomposable and the information returned by each agent is generally assumed to be correct and directly usable for subsequent reasoning, without explicit verification. Under such a design, incorrect or misleading intermediate outputs may propagate through the reasoning chain. In addition, Router-R1 is mainly optimized with outcome-level task rewards, which provide limited credit assignment in multi-turn trajectories: routing decisions made at different stages may contribute unequally to the final outcome, yet the reward signal does not clearly identify which intermediate actions should be rewarded.

Cost-controllable routing. CoRL [11] studies routing from the perspective of cost control by inducing different controller behaviors under different inference budgets. To achieve this, they manually design separate prompt templates for low-, medium-, and high-budget modes, and the routing policy adapts accordingly. However, the budget mode must be explicitly specified at inference time, rather than inferred autonomously from the query and interaction history. Furthermore, their framework focuses on routing under externally imposed budget settings and does not model iterative controller-agent interaction with critique, verification, and stopping decisions. Their objective also does not explicitly encourage balanced utilization across heterogeneous agents.

3 RL-based Multi-Turn Coordination Methodology

In this section, we propose an *iterative critique-and-routing controller* that models the agent coordination as a sequential decision problem. We formalize this as a finite-horizon MDP [25] with explicit load-balancing constraints and optimize the controller using a policy-gradient algorithm.

3.1 Multi-Turn Coordination for Multi-Agent LLM Systems

Given a user query $\mathbf{q}$, the controller sequentially selects agents from the pool $\mathcal{M} = \{\pi_1, \pi_2, \ldots, \pi_K\}$ and interacts with them to iteratively refine a response before producing the final output. The coordination process unfolds over multiple turns as follows.

- **Turn 1 (initial routing).** For an incoming query $\mathbf{q}$, the controller, denoted as $\pi_{\boldsymbol{\theta}}$ and parameterized by $\boldsymbol{\theta}$, selects an agent to produce the initial response:

$$\mathbf{y}_1 = \pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{q}, \mathbf{c}), \quad m_1 = f(\mathbf{y}_1) \quad \longrightarrow \quad \mathbf{g}_1 = \pi_{m_1}(\cdot \mid \mathbf{q}),$$

  where $\mathbf{c}$ denotes system prompt used for the initial routing and $f : \mathcal{Y} \to \mathcal{M}$ is a parsing function that extracts the discrete routing decision from controller's output text $\mathbf{y}$.
- **Turn 2 (evaluation and re-routing).** After observing agent response $\mathbf{g}_1$, the controller evaluates its quality. If the response is deemed insufficient, the controller re-routes the query to another agent and proceeds to the next turn:

$$\mathbf{y}_2 = \pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{q}, \mathbf{c}', \mathbf{g}_1), \quad m_2 = f(\mathbf{y}_2) \quad \longrightarrow \quad \mathbf{g}_2 = \pi_{m_2}(\cdot \mid \mathbf{q}, \mathbf{g}_1, \mathbf{y}_2).$$

  Here, $\mathbf{y}_2$ contains both a re-routing decision and an evaluation of $\mathbf{g}_1$, where the evaluation can be viewed as feedback that conditions the next generation. For turns $t > 1$, the prompt $\mathbf{c}'$ differs from the first-turn prompt $\mathbf{c}$ by instructing the model to assess the draft response in addition to routing.
- **Turn t (general case).** For subsequent turns, the controller repeats the same pattern:

$$\mathbf{y}_t = \pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{q}, \mathbf{c}', \mathbf{g}_{t-1}), \quad m_t = f(\mathbf{y}_t) \quad \longrightarrow \quad \mathbf{g}_t = \pi_{m_t}(\cdot \mid \mathbf{q}, \mathbf{g}_{t-1}, \mathbf{y}_t).$$

  The interaction continues until the controller chooses to terminate or the turn limit T is reached.

In summary, the controller serves a dual role: (i) *router*, selecting an agent from $\mathcal{M}$ and (ii) *critic*, assessing response quality and generating feedback. We instantiate $\pi_{\boldsymbol{\theta}}$ as an LLM and design prompts $\mathbf{c}$ and $\mathbf{c}'$, provided in Appendix B, to guide its behavior. To further strengthen these capabilities, we optimize the controller using RL, as detailed in the remainder of this section.

3.2 Markov Decision Process Formulation

We model the sequential interaction described in the previous subsection as a finite-horizon MDP defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, T)$. The state at turn $t > 1$ is $s_t = (\mathbf{q}, \mathbf{c}', \mathbf{g}_{t-1})$, comprising the user query, the system prompt, and the response generated in the previous turn. At the initial turn, $s_1 = (\mathbf{q}, \mathbf{c})$, where $\mathbf{c}$ is the initial system prompt. The action at turn t is the full controller output $\mathbf{a}_t := \mathbf{y}_t = \pi_{\boldsymbol{\theta}}(\cdot \mid s_t)$, from which we parse a routing decision $m_t = f(\mathbf{y}_t)$. Upon executing action $\mathbf{a}_t$, the selected model m_t produces a new response $\mathbf{g}_t = \pi_{m_t}(\cdot \mid \mathbf{q}, \mathbf{g}_{t-1}, \mathbf{y}_t)$, and the state transitions to $s_{t+1} = (\mathbf{q}, \mathbf{c}', \mathbf{g}_t)$. The episode terminates when the controller deems the current response draft sufficient, or when the maximum number of turns T is reached.

Reward. Following the seminal work [7], we adopt a rule-based action reward. Each action $\mathbf{a}_t := \mathbf{y}_t$ jointly encodes routing and verification (i.e., evaluation). We define a binary verification reward $R_v(\mathbf{s}_t, \mathbf{a}_t)$ and a binary routing reward $R_r(\mathbf{s}_t, \mathbf{a}_t)$. The composite reward is:

$$\begin{aligned} r_t := R(\mathbf{s}_t, \mathbf{a}_t) &= 0.5\, R_r(\mathbf{s}_t, \mathbf{a}_t) + 0.5\, R_v(\mathbf{s}_t, \mathbf{a}_t), \\ R_r(\mathbf{s}_t, \mathbf{a}_t) &= \mathbf{1}\left(C_r(\mathbf{s}_t, \mathbf{a}_t)\right), \quad R_v(\mathbf{s}_t, \mathbf{a}_t) = \mathbf{1}\left(C_v(\mathbf{s}_t, \mathbf{a}_t)\right), \end{aligned} \tag{1}$$

where the reward mixing coefficient is set to 0.5 to equally weight routing and verification. $C_r(\mathbf{s}_t, \mathbf{a}_t)$ and $C_v(\mathbf{s}_t, \mathbf{a}_t)$ indicate whether the routing and verification decisions are correct, respectively. Notably, a routing decision is considered correct if the selected agent produces a correct solution to the problem. For boundary cases where one component is not applicable, we set the corresponding reward to zero (i.e., $R_v(\mathbf{s}_1, \mathbf{a}_1) = 0$ and $R_r(\mathbf{s}_T, \mathbf{a}_T) = 0$). Similar to [11, 37], our current reward formulation targets verifiable domains where the correctness of both decisions is well-defined. Extending the framework to non-verifiable domains would require learned or LLM-based judges.

Training Objective. Under the above MDP formulation, executing the controller policy from the initial state $\mathbf{s}_1$ induced by the query $\mathbf{q}$ induces a trajectory $\tau = (\mathbf{s}_1, \mathbf{a}_1, r_1,\ \mathbf{s}_2, \mathbf{a}_2, r_2,\ \ldots,\ \mathbf{s}_T, \mathbf{a}_T, r_T)$. Although some episodes terminate before turn T, we formulate the problem using a fixed horizon T for notation convenience; after termination, the process accrues zero reward and no further agent invocation. The controller is trained to maximize the expected discounted return:

$$\begin{aligned} &\underset{\boldsymbol{\theta}}{\text{maximize}}\ \ \mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\,\mathbb{E}_{\mathbf{a}_t\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)}\left[\sum_{t=1}^{T}\gamma^{t-1}R(\mathbf{s}_t,\mathbf{a}_t)\right], \\ &\text{subject to}\ \ \sum_{t=1}^{T}\mathbb{E}\big[\mathbf{1}(f(\mathbf{a}_t)=k)\big] \le \rho_k \sum_{k'=1}^{K}\sum_{t=1}^{T}\mathbb{E}\big[\mathbf{1}(f(\mathbf{a}_t)=k')\big], \quad \forall k, \end{aligned} \tag{2}$$

where $\gamma \in (0, 1]$ is a discount factor, query $\mathbf{q}$ is encoded in $\mathbf{s}_t$, $\mathbf{1}(f(\mathbf{a}_t) = k)$ indicates whether agent k is invoked at turn t, $\sum_{t=1}^{T}\mathbb{E}[\mathbf{1}(f(\mathbf{a}_t) = k)]$ is the expected invocation count of agent k, and $\sum_{k=1}^{K}\sum_{t=1}^{T}\mathbb{E}[\mathbf{1}(f(\mathbf{a}_t) = k)]$ is the expected total invocation count across all agents. The constraint, configured by ρ_k, limits the usage of each agent and promotes load balance across the agent pool.

3.3 Lagrangian Relaxation

Problem (2) is a constrained reward maximization that is difficult to optimize directly using existing policy gradient methods. We therefore reformulate it via Lagrangian relaxation [29]. The resulting unconstrained objective is given in Proposition 3.1. The proof is provided in Appendix D.

Proposition 3.1. Suppose that the discount factor $\gamma = 1$. Define the modified composite reward and routing reward as:

$$R'(\mathbf{a}_t, \mathbf{s}_t) := 0.5R'_r(\mathbf{a}_t, \mathbf{s}_t) + 0.5\,R_v(\mathbf{s}_t, \mathbf{a}_t),\ \ R'_r(\mathbf{a}_t, \mathbf{s}_t) := R_r(\mathbf{s}_t, \mathbf{a}_t) - \textstyle\sum_{k=1}^{K} 2\xi_k \cdot \mathbf{1}(f(\mathbf{a}_t)=k), \tag{3}$$

where $\xi_k \ge 0$ is the tunable Lagrange multiplier associated with the usage constraint on agent k. Problem (2) can be optimized through the following unconstrained modified-reward maximization:

$$\underset{\boldsymbol{\theta}}{\text{maximize}}\ \ \mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\,\mathbb{E}_{\mathbf{a}_t\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)}\left[\sum_{t=1}^{T}\gamma^{t-1}R'(\mathbf{a}_t,\mathbf{s}_t)\right]. \tag{4}$$

Remark 3.2 (Interpretation of the Modified Routing Reward)**.** The modified routing reward $R'_r(\mathbf{a}_t, \mathbf{s}_t)$ penalizes over-reliance on individual agents by subtracting a usage cost proportional to ξ_k whenever agent k is selected. When the invocation frequency of agent k exceeds its budget, increasing ξ_k reduces its effective routing reward, discouraging the controller from repeatedly routing to that agent. This mechanism provides a configurable knob to balance workload across the pool without modifying the underlying optimization algorithm.

Remark 3.3 (Justification of the Relaxation)**.** The constraints in (2) are designed to encourage load-balanced utilization and are not necessarily hard requirements at deployment time. While the relaxation in (4) is not strictly equivalent to the original constrained formulation, it yields a theoretically grounded surrogate objective that is well suited to standard policy-gradient algorithms.

3.4 Policy Gradient Methods for Controller Training

Problem (4) can be optimized with established policy-gradient algorithms. In this work, we adopt a commonly used variant of policy-gradient methods in LLM, *Group Relative Policy Optimization* (GRPO) [31], which samples multiple rollouts per query and optimizes the policy using group-relative advantage. The detailed justification of tailoring GRPO to Problem (4) is provided in Appendix E.1. In this section, we convert the multi-turn discounted reward into group-relative advantage, a key component of GRPO.

Following the interaction procedure in Section 3.1, GRPO samples multiple rollouts for each query at each turn. To distinguish these trajectories, we introduce an additional index b, and denote by $\mathbf{s}_t^b$ the state at turn t along rollout b (see Figure 2). This notation is used only to facilitate the recursive computation of discounted returns under group sampling. From each state $\mathbf{s}_t^b$, we sample G actions $\{\mathbf{a}_t^{(b,i)}\}_{i=1}^{G}$ (i.e., responses) from the controller $\pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{s}_t^b)$. Each sampled action yields an immediate modified action reward via (3),

$$r_t^{(b,i)} = R'(\mathbf{s}_t^b, \mathbf{a}_t^{(b,i)}),$$

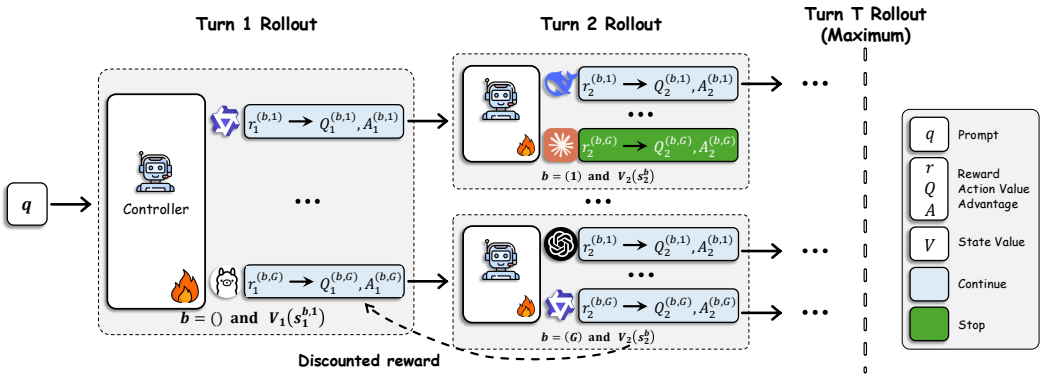


Figure 2: Multi-turn rollout tree for GRPO training of the critique-and-routing controller. From each state $\mathbf{s}_t^b$ with prefix b (starting from the root prefix $()$), the controller samples a group of actions $\mathbf{a}_t^{(b,i)}$, yielding a group of new prefixes $\{(b,i)\}_{i=1}^G$. Non-terminal actions invoke an agent to produce the next state $\mathbf{s}_{t+1}^{(b,i)}$, while each sampled action receives a modified reward that incorporates routing quality, verification quality, and an agent-utilization penalty.

and induces a next state $\mathbf{s}_{t+1}^{(b,i)}$. The discounted return defined in Problem (4) at state $\mathbf{s}_t^b$ is thus

$$\hat{V}_t(\mathbf{s}_t^b) = \frac{1}{G}\sum_{i=1}^{G}\left[r_t^{(b,i)} + \gamma \hat{V}_{t+1}(\mathbf{s}_{t+1}^{(b,i)})\right], \tag{5}$$

with terminal condition $\hat{V}_{T+1}(\cdot) = 0$. Using this discounted return, we define the group relative advantage for action $\mathbf{a}_t^{(b,i)}$ as

$$A_t^{(b,i)} = \hat{Q}_t^{(b,i)} - \text{mean}\left(\{\hat{Q}_t^{(b,j)}\}_{j=1}^G\right), \quad \hat{Q}_t^{(b,i)} = r_t^{(b,i)} + \gamma \hat{V}_{t+1}\left(\mathbf{s}_{t+1}^{(b,i)}\right). \tag{6}$$

With this advantage estimate, we then optimize the controller policy using the standard GRPO algorithm, which is detailed in Appendix E.2.

4 Experiments

Datasets. We train the controller on the training split of MATH-lighteval [8] and evaluate it on the corresponding test split and six additional math benchmarks: MATH500 [8], Omni-MATH [6], MinervaMath [17], AGI-Eval-MATH [38], AMC23 [14], and AIME24. Together, these benchmarks provide a comprehensive testbed for evaluating effectiveness and generalization of our method.

Model Pools. We consider two heterogeneous multi-agent systems with different controllers and agent pools. In System 1, the controller is initialized with Qwen2.5-7B-Base, and the candidate agents are Qwen3-30B-A3B-Instruct, Qwen2.5-7B-Instruct, and Qwen2.5-1.5B-Instruct [10, 36]. In System 2, we use Qwen3-4B-Base as the controller, and the candidate agents are Qwen3-30B-A3B-Instruct, Ministral-3-8B-Instruct-2512, and Llama-3-2-1B-Instruct [10, 36]. We also study additional controller choices; the corresponding results are reported in Appendix A.1.

Baselines. We compare our method with the following baselines.

- Router-R1: a state-of-the-art RL-based reasoning router [37]. Since the query is not decomposable in our setting, the controller routes the full problem to a single agent.
- RouterDC [2]: a contrastive-learning-based router that maps queries and agents into a shared embedding space and selects agents based on embedding similarity.
- RoBERTa Router: trains a RoBERTa-based classifier with supervised learning to predict the best agent for a given query [3, 23].
- Random Router: randomly selects an agent without any learned routing policy.
- Controller V1: uses the controller itself as the solver and trains it with RL-based post-training (GRPO) using an outcome-based reward [7].
- Controller V2: trains the controller with our method and directly uses it as a standalone solver at inference time.

Table 1: **Accuracy comparison for two heterogeneous multi-agent systems with different controllers and agent pools.** We abbreviate MATH-lighteval as M-LEval, Omni-MATH as Omni, MinervaMath as Minerva, and AGI-Eval-MATH as AGI-M. Overall, our method achieves the highest average accuracy among baselines in both systems under *usage constraints* on Agents 1 and 2.

Methods	M-LEval	MATH-500	Omni	Minerva	AGI-M	AMC23	AIME24	Avg.
Controller: Qwen2.5-7B-Base								
Agent 1: Qwen3-30B-A3B-Instruct; **Agent 2**: Qwen2.5-7B-Instruct; **Agent 3**: Qwen2.5-1.5B-Instruct.								
Agent 1	0.956	0.938	0.551	0.828	0.933	0.975	0.833	0.859
Agent 2	0.757	0.750	0.254	0.524	0.734	0.500	0.100	0.517
Agent 3	0.554	0.542	0.157	0.273	0.547	0.275	0.067	0.345
Controller V1	0.626	0.596	0.205	0.247	0.597	0.450	0.100	0.403
Controller V2	0.749	0.760	0.246	0.423	0.724	0.600	0.200	0.529
Random Router	0.756	0.744	0.304	0.537	0.737	0.575	0.275	0.561
RoBERTa Router	0.763	0.750	0.436	0.417	0.729	0.550	0.667	0.616
RouterDC	0.784	0.764	0.445	0.419	0.736	0.600	0.275	0.575
Router-R1	0.776	0.778	0.495	0.432	0.763	0.825	0.767	0.691
Ours	**0.901**	**0.888**	**0.534**	**0.678**	**0.879**	**0.925**	**0.833**	**0.805**
Controller: Qwen3-4B-Base								
Agent 1: Qwen3-30B-A3B-Instruct; **Agent 2**: Ministral-3-8B-Instruct; **Agent 3**: Llama-3-2-1B-Instruct.								
Agent 1	0.956	0.938	0.551	0.828	0.933	0.975	0.833	0.859
Agent 2	0.724	0.720	0.180	0.599	0.681	0.400	0.100	0.486
Agent 3	0.275	0.260	0.067	0.088	0.265	0.200	0.000	0.165
Controller V1	0.761	0.738	0.255	0.460	0.736	0.550	0.133	0.519
Controller V2	0.883	0.804	0.328	0.419	0.809	0.775	0.333	0.623
Random Router	0.670	0.659	0.244	0.528	0.640	0.500	0.267	0.501
RoBERTa Router	0.659	0.662	0.458	0.376	0.628	0.775	0.833	0.627
RouterDC	0.681	0.674	0.463	0.386	0.654	0.600	0.467	0.561
Router-R1	0.686	0.682	0.492	0.405	0.666	0.925	0.800	0.665
Ours	**0.915**	**0.896**	**0.541**	**0.718**	**0.881**	**0.975**	**0.833**	**0.823**

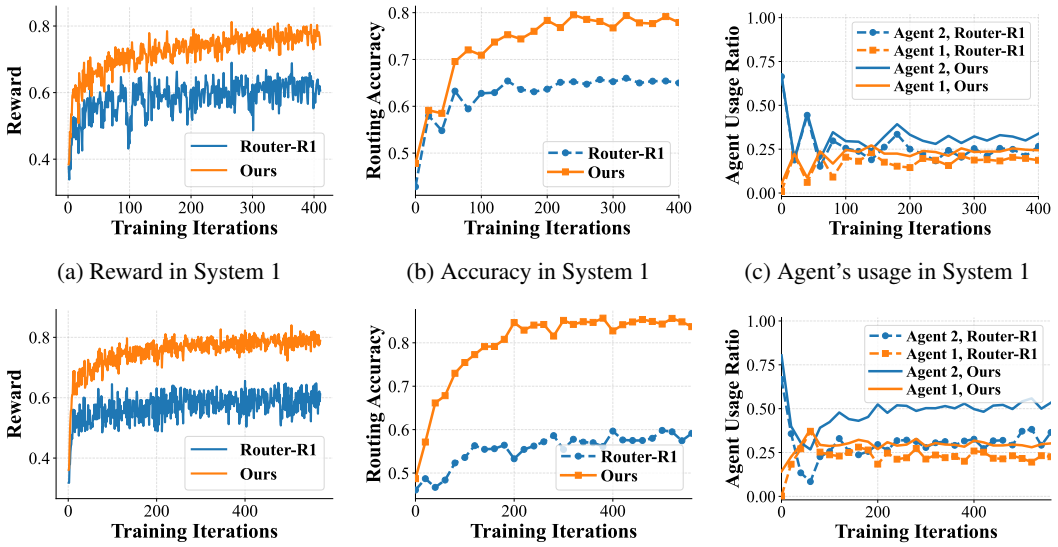
Resource Constraints. To avoid over-reliance on the strongest agent and to encourage balanced utilization, we impose a usage budget on that agent during the training. Specifically, we set its usage constraint to $\rho_1 = 0.25$, meaning that it can be selected in at most 25% of the whole invocations. In addition, ρ_2 and ρ_3 are set to be 0.5 and 1, respectively. These constraints can be satisfied by setting appropriate Lagrange multipliers $\{\xi_k\}_{k=1}^3$ (see Table 9). We discuss the choice of ρ_1 and ρ_2 and further study the sensitivity of ξ_k in Appendix A.2.

Other Details. Our experiments are implemented using the verl framework [32]. We conduct all experiments on a cluster with 4 NVIDIA H100 GPUs, each with 80 GB of memory. Additional implementation details and hyperparameter settings are provided in Appendix C.

4.1 Main Results

In Table 1, we compare the accuracy of our method with several baselines across seven math benchmarks under two heterogeneous multi-agent systems. In System 1, where the controller is Qwen2.5-7B-Base and the agents have sizes 30B, 7B, and 1.5B, our method achieves an average accuracy of 80.5%. It outperforms the strongest routing baseline, Router-R1 (69.1%), by 11.4 percentage points. The advantage is consistent across these benchmarks and is particularly pronounced on challenging benchmarks such as AGI-Eval-MATH and MinervaMath, highlighting the effectiveness of our approach in coordinating multiple agents for complex reasoning tasks. In System 2, with Qwen3-4B-Base as the controller and agents of sizes 30B, 8B, and 1B, our method further improves the average accuracy to 82.3%, surpassing Router-R1 (66.5%) by 15.8 percentage points. The strong results across both systems indicate that our method with *explicit multi-turn interaction* generalizes well across different controller capacities and agent compositions. Furthermore, the comparison with baselines Controller V1 and Controller V2 suggests that the gains are related not only to controller strength, but also to more effective coordination behavior.

In Figure 3, we present the training dynamics of the controllers in the two multi-agent systems under Router-R1 and our method, in terms of reward, routing accuracy, and agent usage ratio throughout



(a) Reward in System 1 (b) Accuracy in System 1 (c) Agent's usage in System 1

(d) Reward in System 2 (e) Accuracy in System 2 (f) Agent's usage in System 2

Figure 3: **Training dynamics for two multi-agent systems with different controller and agent-pool configurations.** System 1 uses Qwen2.5-7B-Base as the controller with agents Qwen3-30B-A3B-Instruct (Agent 1), Qwen2.5-7B-Instruct (Agent 2), and Qwen2.5-1.5B-Instruct. System 2 uses Qwen3-4B-Base as the controller with agents Qwen3-30B-A3B-Instruct (Agent 1), Ministral-3-8B-Instruct (Agent 2), and Llama-3-2-1B-Instruct. Compared with Router-R1, our method significantly improves reward and accuracy with slightly higher usage on the strongest agent.

training. The agent usage ratio is defined as the number of times an agent is invoked, normalized by the dataset size. Other baselines are excluded since their training dynamics are not directly comparable.

Reward trajectories: From Figures 3a and 3d, we see that our method achieves rapid reward improvement across both systems, reaching stable high-reward plateaus within the first 100 steps, with final rewards substantially higher than Router-R1. This demonstrates that the controller learns both effective routing and verification behaviors over multiple turns that lead to a higher reward.

Routing accuracy trends: The routing accuracy curves in Figures 3b and 3e show that our method learns a substantially more accurate routing policy than Router-R1 in both systems. This improvement suggests that the *multi-turn interaction* mechanism provides additional opportunities for the controller to refine or correct earlier routing decisions. It also confirms that the proposed method achieves meaningful gains in routing policy optimization, which is a critical factor underlying the overall accuracy improvement shown in Table 1.

Agent utilization patterns: Figures 3c and 3f report the usage ratios of the strongest and intermediate agents. Overall, our method leads to only a modest increase in strongest-agent usage relative to Router-R1, suggesting that the performance gains are not simply due to over-reliance on the strongest agent. This can be attributed to the hierarchical reward design that encourages more effective use of stronger agents while still respecting the usage constraint. In System 2, our method also shows higher utilization of the intermediate-sized agent, which is consistent with the explicit multi-turn interaction mechanism in our framework. In summary, these results indicate that our method achieves a favorable balance between performance and agent utilization.

4.2 Ablation Studies

To assess the contribution of each component in our framework, we conduct ablation studies on the number of interaction turns and the Lagrange multipliers.

Impact of the Number of Turns. We vary the maximum number of interaction turns T at inference time and report the accuracy on MATH-lighteval benchmark in Figure 4 for both Systems 1 and 2. Notably, the maximum number of interaction turns used during training does not need to match

Table 2: **Comparison on seven math benchmarks for two heterogeneous multi-agent systems with and without Lagrangian multipliers.** We abbreviate MATH-lighteval as M-LEval, Omni-MATH as Omni, MinervaMath as Minerva, and AGI-Eval-MATH as AGI-M.

Methods	M-LEval	MATH-500	Omni	Minerva	AGI-M	AMC23	AIME24	Avg.
				System 1				
W/O Lagrangian	0.756	0.746	0.304	0.537	0.737	0.575	0.300	0.565
W Lagrangian	**0.901**	**0.888**	**0.534**	**0.678**	**0.879**	**0.925**	**0.833**	**0.805**
				System 2				
W/O Lagrangian	0.670	0.660	0.245	0.529	0.640	0.500	0.267	0.501
W Lagrangian	**0.915**	**0.896**	**0.541**	**0.718**	**0.881**	**0.975**	**0.833**	**0.823**

that used during inference. Multi-turn training equips the controller with the ability to interact over multiple rounds, without requiring an exact match between training and inference horizons. As shown in Figure 4, the accuracy increases significantly when the maximum number of turns is increased from 1 to 2, which further validates the effectiveness of multi-turn interaction in boosting the accuracy. We also observe that the performance gains saturate as the number of turns continues to increase, suggesting that a small turn budget is already sufficient for most prompts. This is further supported by Figure 5, which shows that only a small fraction of prompts exhaust the turn budget under larger maximum-turn settings.

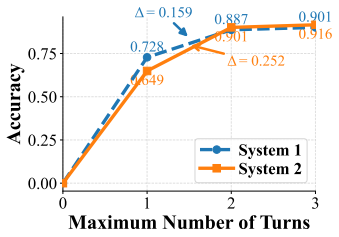


Figure 4: Accuracy under different maximum turn budgets

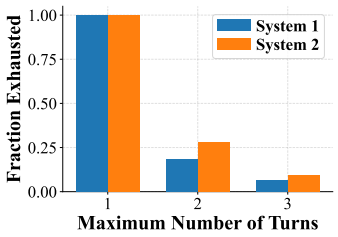


Figure 5: Percentage of prompts exhausting the turn budget.

Impact of the Lagrange Multipliers. We introduce Lagrange multipliers ξ_k to discourage over-reliance on the k-th agent. To evaluate their effect, we compare system performance with and without these multipliers. For the three-agent systems considered here, the setting with Lagrange multipliers uses $\xi_1 = 0.25$, $\xi_2 = 0.125$, and $\xi_3 = 0$, while the setting without them sets $\xi_1 = \xi_2 = \xi_3 = 0$ in Equation (3). We observe that removing the Lagrange multipliers results in uncontrolled agent usage, where all the queries are routed to the strongest agent. We report the overall accuracy in Table 2 under the usage constraints $\rho_1 = 0.25$ and $\rho_2 = 0.5$. During evaluation, once an agent's budget is exhausted, incoming requests are automatically routed to the next-best available agent for both our method and the baselines. Table 2 demonstrates that the Lagrangian penalty effectively enables the system to achieve higher accuracy under constrained agent usage.

5 Conclusion and Limitations

We studied controller design for multi-turn coordination in heterogeneous multi-agent LLM systems. We proposed an iterative critique-and-routing controller that evaluates intermediate drafts, decides whether to stop or continue, and selects the next agent for further refinement when needed. We formulated this process as a finite-horizon MDP with explicit agent-utilization constraints, designed a composite reward for decisions across turns, and optimized the controller via policy gradients under a Lagrangian-relaxed objective. Extensive experiments across multiple heterogeneous multi-agent systems and seven reasoning benchmarks demonstrate that our method consistently outperforms strong routing baselines and substantially narrows the gap to the strongest agent while using it for fewer than 25% of total calls. Overall, these results highlight the value of iterative refinement in multi-agent LLM systems. A limitation of our work is that experiments primarily focus on math reasoning benchmarks, where routing and verification rewards are relatively straightforward to define. In addition, the multi-turn design introduces extra inference latency and computational overhead compared to one-shot routing, reflecting a trade-off between performance and efficiency.

References

[1] Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

[2] Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems*, 37:66305–66328, 2024.

[3] Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks VS Lakshmanan, and Ahmed Hassan Awadallah. Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618*, 2024.

[4] Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Carmen Hipolito Garcia, Menglin Xia, Laks VS Lakshmanan, Qingyun Wu, and Victor Rühle. Best-route: Adaptive llm routing with test-time optimal compute. *arXiv preprint arXiv:2506.22716*, 2025.

[5] Andrew Estornell, Jean-Francois Ton, Yuanshun Yao, and Yang Liu. Acc-collab: An actor-critic approach to multi-agent llm collaboration. *arXiv preprint arXiv:2411.00053*, 2024.

[6] Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, et al. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*, 2024.

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

[8] 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 *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[9] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *The twelfth international conference on learning representations*, 2023.

[10] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

[11] Bowen Jin, TJ Collins, Donghan Yu, Mert Cemri, Shenao Zhang, Mengyu Li, Jay Tang, Tian Qin, Zhiyang Xu, Jiarui Lu, et al. Controlling performance and budget of a centralized multi-agent llm system with reinforcement learning. *arXiv preprint arXiv:2511.02755*, 2025.

[12] 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.

[13] Guangchen Lan, Dong-Jun Han, Abolfazl Hashemi, Vaneet Aggarwal, and Christopher Brinton. Asynchronous federated reinforcement learning with policy gradient updates: Algorithm design and convergence analysis. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

[14] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 2022.

[15] Ao Li, Yuexiang Xie, Songze Li, Fugee Tsung, Bolin Ding, and Yaliang Li. Agent-oriented planning in multi-agent systems. *arXiv preprint arXiv:2410.02189*, 2024.

[16] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in neural information processing systems*, 36:51991–52008, 2023.

[17] Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

[18] Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. A survey on llm-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth*, 1(1):9, 2024.

[19] Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025.

[20] Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. Routing to the expert: Efficient reward-guided ensemble of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1964–1974, 2024.

[21] Kai Mei, Wujiang Xu, Minghao Guo, Shuhang Lin, and Yongfeng Zhang. Omnirouter: Budget and performance controllable multi-llm routing. *ACM SIGKDD Explorations Newsletter*, 27(2):107–116, 2025.

[22] 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.

[23] Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data. *arXiv preprint arXiv:2406.18665*, 2024.

[24] Manhin Poon, XiangXiang Dai, Xutong Liu, Fang Kong, John CS Lui, and Jinhang Zuo. Online multi-llm selection via contextual bandits under unstructured context evolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 24855–24863, 2026.

[25] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[26] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 15174–15186, 2024.

[27] Sumedh Rasal and Evi J Hauer. Navigating complexity: Orchestrated problem solving with multi-agent llms. *arXiv preprint arXiv:2402.16713*, 2024.

[28] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

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

[30] Chenyang Shao, Xinyang Liu, Yutang Lin, Fengli Xu, and Yong Li. Route-and-reason: Scaling large language model reasoning with reinforced model router. *arXiv preprint arXiv:2506.05901*, 2025.

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

[32] 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*, pages 1279–1297, 2025.

[33] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

[34] Yashar Talebirad and Amirhossein Nadiri. Multi-agent collaboration: Harnessing the power of intelligent llm agents. *arXiv preprint arXiv:2306.03314*, 2023.

[35] 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 conversations. In *First conference on language modeling*, 2024.

[36] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[37] Haozhen Zhang, Tao Feng, and Jiaxuan You. Router-r1: Teaching llms multi-round routing and aggregation via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[38] Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models. In *Findings of the association for computational linguistics: NAACL 2024*, pages 2299–2314, 2024.

Appendix

A Further Experiments **14**

A.1 Other Controller Choices 14

A.2 Agent Usage Constraint and Sensitivity Analysis of the Lagrangian Multipliers . . 14

A.3 Limitation of One-Turn Routing 15

A.4 Controller's Capability in Verification 16

B Prompt Template **17**

C Details on Datasets and Hyperparameters **18**

D Proof of Proposition 3.1 **18**

E Algorithm Details **19**

E.1 Problem (4) and GRPO 19

E.1.1 Policy Gradient of Problem (4) 19

E.1.2 Policy Gradient in GRPO 20

E.2 Practical On-Policy GRPO Procedure for Problem (4) 21

F Case Study **22**

Table 3: Accuracy comparison on seven math datasets for two multi-agent systems using alternative controller models under *usage constraints* on Agents 1 and 2. We abbreviate MATH-lighteval as M-LEval, Omni-MATH as Omni, MinervaMath as Minerva, and AGI-Eval-MATH as AGI-M. The performance advantage of our method is preserved in both systems.

Methods	M-LEval	MATH-500	Omni	Minerva	AGI-M	AMC23	AIME24	Avg.
Controller: Qwen3-4B-Base								
Agent 1: Qwen3-30B-A3B-Instruct; **Agent 2**: Qwen2.5-7B-Instruct; **Agent 3**: Qwen2.5-1.5B-Instruct.								
Agent 1	0.956	0.938	0.551	0.828	0.933	0.975	0.833	0.859
Agent 2	0.757	0.750	0.254	0.524	0.734	0.500	0.100	0.517
Agent 3	0.554	0.542	0.157	0.273	0.547	0.275	0.067	0.345
Controller V1	0.761	0.738	0.255	0.460	0.736	0.550	0.133	0.519
Controller V2	0.883	0.804	0.328	0.419	0.809	0.775	0.333	0.622
Random Router	0.756	0.744	0.304	0.537	0.737	0.575	0.275	0.561
RoBERTa Router	0.763	0.750	0.436	0.417	0.729	0.550	0.667	0.616
RouterDC	0.784	0.764	0.445	0.419	0.736	0.600	0.275	0.575
Router-R1	0.811	0.808	0.501	0.480	0.798	0.900	0.767	0.724
Ours	**0.905**	**0.878**	**0.535**	**0.665**	**0.878**	**0.925**	**0.833**	**0.803**
Controller: Qwen3-8B-Base								
Agent 1: Qwen3-30B-A3B-Instruct; **Agent 2**: Ministral-3-8B-Instruct; **Agent 3**: Llama-3-2-1B-Instruct.								
Agent 1	0.956	0.938	0.551	0.828	0.933	0.975	0.833	0.859
Agent 2	0.724	0.720	0.180	0.599	0.681	0.400	0.100	0.486
Agent 3	0.275	0.260	0.067	0.088	0.265	0.200	0.000	0.165
Controller V1	0.760	0.738	0.261	0.524	0.735	0.750	0.167	0.562
Controller V2	0.810	0.792	0.280	0.608	0.776	0.600	0.200	0.581
Random Router	0.670	0.659	0.244	0.528	0.640	0.500	0.267	0.501
RoBERTa Router	0.659	0.662	0.458	0.376	0.628	0.775	0.833	0.627
RouterDC	0.681	0.674	0.463	0.386	0.654	0.600	0.467	0.561
Router-R1	0.640	0.628	0.477	0.409	0.638	0.950	0.800	0.649
Ours	**0.915**	**0.876**	**0.539**	**0.727**	**0.893**	**0.833**	**0.975**	**0.823**

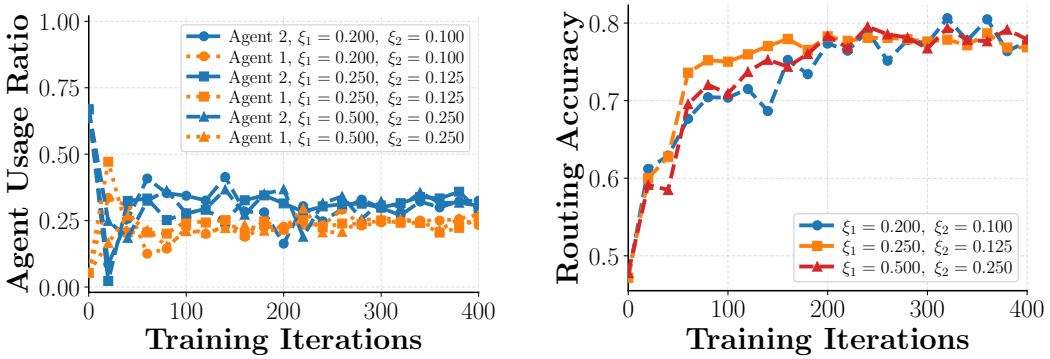
A Further Experiments

A.1 Other Controller Choices

Table 3 reports an additional study under the same experimental setup as Table 1, where we replace the controller backbone with alternative models while keeping the agent pools, benchmarks, and baselines unchanged. This allows us to isolate the effect of the controller and evaluate whether our training method generalizes across different controller capacities. We observe that our method consistently achieves the best performance across all benchmarks and both system configurations, regardless of the underlying controller. In particular, under the Qwen3-4B-Base controller, our method attains an average accuracy of 80.3%, outperforming the strongest baseline (Router-R1, 72.4%) by a substantial margin. Similarly, when using Qwen3-8B-Base as the controller, our method further improves the average accuracy to 82.3%, exceeding Router-R1 (64.9%) by 17.4 percentage points. Overall, these results, together with Table 1, suggest that the proposed method exhibits robustness to the choice of controller across a range of settings, supporting its applicability in heterogeneous multi-agent systems.

A.2 Agent Usage Constraint and Sensitivity Analysis of the Lagrangian Multipliers

Discussion on Agent Usage Constraints. For routing in a multi-agent system with heterogeneous models, there exists an optimal partition of queries across agents. Ideally, queries that can be correctly handled by the weakest agent should be routed to that agent, and the remaining queries should be routed to stronger agents as needed. For a three-agent system, let the accuracies of the three agents be p_1, p_2, and p_3, where agent 3 is the weakest and agent 1 is the strongest. Then, the fraction of queries routed to agents 2 and 1 should be no larger than $1 - p_3$ and $1 - p_2$, respectively. This is the principle behind setting ρ_1 and ρ_2, which correspond to the usage constraints of agents 1 and 2.



(a) Agent usage under System 1.

(b) Routing accuracy under System 1.

Figure 6: Sensitivity of our method to different Lagrangian multipliers. We vary the multiplier settings (ξ_1, ξ_2) and report the resulting agent-usage and routing-accuracy dynamics. Across these choices, the learned controller remains stable and performs well: the usage ratios stay well controlled, while the routing accuracy remains consistently high after convergence.

In our experiments, we treat these values as fixed system-level parameters chosen according to this principle.

Sensitivity Analysis of the Lagrangian Multipliers. We further evaluate the sensitivity of our method to the choice of Lagrangian multipliers. Specifically, we vary (ξ_1, ξ_2) under System 1 and track both the agent-usage ratios and routing accuracy throughout training. Notably, in System 1, the controller is initialized with Qwen2.5-7B-Base, and the candidate agents are Qwen3-30B-A3B-Instruct, Qwen2.5-7B-Instruct, and Qwen2.5-1.5B-Instruct. As shown in Figure 6, within the moderate range of multiplier values considered here, the training dynamics are broadly similar across settings: the usage ratios remain reasonably controlled, while the routing accuracy converges to a comparable level. These results suggest that, within a moderate range of multiplier values, the controller can automatically learn a stable partition of queries across agents, while maintaining a favorable balance between satisfying the usage constraints and preserving routing performance.

A.3 Limitation of One-Turn Routing

To better understand the limitation of one-turn routing, we analyze its difficulty prediction behavior through a routing confusion matrix. In one-turn routing, the controller makes a one-shot estimate of the problem difficulty and then dispatches the query to one of three models accordingly. In System 1, Qwen2.5-1.5B-Instruct for easy problems, Qwen2.5-7B-Instruct for medium problems, and Qwen3-30B-A3B-Instruct for hard problems. In System 2, Llama-3-2-1B-Instruct for easy problems, Ministral-3-8B-Instruct for medium problems, and Qwen3-30B-A3B-Instruct for hard problems. The rows of the confusion matrix denote the ground-truth difficulty level, assigned according to the minimum-capability model that can correctly solve the problem: easy, medium, and hard correspond to problems solvable by the weakest, medium, and strongest model, respectively. The columns denote the model selected by the router. Correct routing appear on the diagonal, whereas off-diagonal entries indicate under-routing or over-routing errors. In particular, predicting a lower difficulty than the ground truth may route a problem to an insufficiently capable model, while predicting a higher difficulty than the ground truth may cause unnecessary reliance on stronger models. The results are reported in Tables 4 and 5. The controller is both trained and evaluated on the MATH-lighteval dataset, with a total of 5000 samples in the test split.

From Tables 4 and 5, we observe that the router exhibits a systematic bias toward selecting weaker models. Notably, such bias, either toward weaker or stronger models, is inherent in one-shot routing, since perfectly partitioning queries by their required model capability is generally difficult. Under our resource constraints, where the usage of stronger models is explicitly limited, the router may further favor weaker models when the difficulty estimate is uncertain. This motivates our multi-turn coordination, which allows the controller to revise its routing decisions based on intermediate interactions.

Table 4: Routing confusion matrix for one-turn routing in System 1. Rows denote the ground-truth problem difficulty, and columns denote the model selected by the router. The weakest, medium, and strongest models correspond to Qwen2.5-1.5B-Instruct, Qwen2.5-7B-Instruct, and Qwen3-30B-A3B-Instruct, respectively. Off-diagonal entries indicate routing errors that either under-estimate or over-estimate the required model capability.

Problem Difficulty	**Routed Model**		
	Weakest	**Medium**	**Strongest**
Easy	2254	302	90
Medium	**515**	394	228
Hard	**211**	**330**	676

Table 5: Routing confusion matrix for one-turn routing in System 2. Rows denote the ground-truth problem difficulty, and columns denote the model selected by the router. The weakest, medium, and strongest models correspond to Llama-3-2-1B-Instruct, Ministral-3-8B-Instruct, and Qwen3-30B-A3B-Instruct, respectively. Off-diagonal entries indicate routing errors that either under-estimate or over-estimate the required model capability.

Problem Difficulty	**Routed Model**		
	Weakest	**Medium**	**Strongest**
Easy	1064	179	49
Medium	**1080**	782	468
Hard	**136**	**248**	994

A.4 Controller's Capability in Verification

Verification (i.e., evaluation) is a key component of our sequential coordination controller. In our multi-turn framework, re-routing occurs only when the controller identifies that the response produced in the current turn is not sufficiently reliable. Therefore, the controller must not only select an appropriate agent, but also judge whether the intermediate response should be accepted or further refined.

To evaluate this capability, we focus on the controller's verification behavior at the second turn, where the number of active queries is still sufficiently large before decreasing in later turns. We present confusion matrices over the two heterogeneous multi-agent systems considered in Table 1. In particular, here we do not verify responses generated by the strongest agent. In System 1 and System 2, the controllers are instantiated as Qwen2.5-7B-Base and Qwen3-4B-Base, respectively. Rows denote the ground-truth correctness of the current response, while columns denote the controller's verification decision. Correct verification decisions appear on the diagonal, whereas off-diagonal entries indicate false acceptance or false rejection. A false acceptance occurs when the controller incorrectly accepts an incorrect response, while a false rejection occurs when the controller rejects a correct response and continues the interaction unnecessarily. These results are reported in Tables 6 and 7.

Table 6: Confusion matrix of controller verification decisions in System 1. Controller is trained and tested on MATH-lighteval task.

	Predicted Incorrect	Predicted Correct
Actual Incorrect	**892**	271
Actual Correct	216	2805

From Tables 6 and 7, we observe that the controller achieves high verification accuracy, with most entries concentrated on the diagonal. In particular, the number of false rejections remains relatively small, 216 in System 1 and 199 in System 2, indicating that the controller rarely discards correct responses. At the same time, the controller correctly identifies a large portion of incorrect responses, 892 in System 1 and 1247 in System 2, corresponding to incorrect-response recall rates of 76.69% and 80.71%. This demonstrates the controller's ability to filter out low-quality responses. Overall,

Table 7: Confusion matrix of controller verification decisions in System 1. Controller is trained and tested on MATH-lighteval task.

	Predicted Incorrect	Predicted Correct
Actual Incorrect	**1247**	298
Actual Correct	199	2563

the controller maintains meaningful rejection capability while preserving a high acceptance rate for correct responses. These results help explain why the multi-turn framework leads to improved performance: it enables the controller to identify insufficient intermediate responses and trigger re-routing.

B Prompt Template

Table 8: **Controller prompt templates for multi-turn coordination.** Template I corresponds to the initial-turn prompt $\mathbf{c}$, which performs routing based solely on the input query. Template II corresponds to the subsequent-turn prompt $\mathbf{c}'$, which verifies the previous agent's solution and either terminates the interaction or routes the problem to a stronger agent.

Template I (initial routing prompt $\mathbf{c}$).
System prompt: You are a router.
Given a math problem, your task is to route the problem to the appropriate model based on its difficulty.
Rules:

- The available models, ordered from weaker to stronger, are: `AGENT LIST`.
- Choose the appropriate model based on the problem difficulty.

Output format:

- First, put your thinking process inside `<thinking>...</thinking>` tags.
- Then, output your choice in the format `<model>`AGENT NAME`</model>`.
- Your thinking should briefly explain the difficulty only.
- You don't need to solve the problem.

Template II (subsequent-turn verification and re-routing prompt $\mathbf{c}'$).
System prompt: You are a math solution verifier and router.
Given a math problem and a proposed solution from `AGENT NAME`, determine whether the proposed solution is fully correct.
Rules:

- Check the proposed solution step by step.
- If any step is incorrect, unjustified, incomplete, or relies on invalid assumptions, return `False` and route the problem to a stronger model.
- If the solution is fully correct, return `True` and stop.
- The available models, ordered from weaker to stronger, are: `AGENT LIST`.
- If routing is needed, select exactly one model that is stronger than `AGENT NAME`.

Output format (you MUST use exactly this structure):

- First, put your verification inside `<checking>...</checking>`.
- Then output either `<verdict>`True`</verdict>` or `<verdict>`False`</verdict>`.
- If and only if the verdict is `False`, output in the next line: `<model>`AGENT NAME`</model>`

To instantiate the controller in Section 3.1, we design two system prompt templates provided in Table 8. The initial-turn prompt $\mathbf{c}$ is used for routing at turn $t = 1$, where the controller selects an agent based solely on the input query. The subsequent-turn prompt $\mathbf{c}'$ is used for turns $t > 1$, where the controller acts as both verifier and router: it evaluates the previous response and either terminates the interaction or re-routes the query to a stronger agent.

C Details on Datasets and Hyperparameters

Details on Datasets. We fine-tune the controller on MATH-lighteval [8]. The MATH-lighteval dataset comprises 12500 problems drawn from mathematics competitions, covering topics such as algebra, geometry, counting and probability, number theory, and precalculus. We use 7500 samples (e.g., train split) for training and 5000 (e.g., test split) for testing. Additionally, we evaluate the system fine-tuned on the MATH-lighteval dataset against other 6 widely used mathematical benchmarks: MATH500 [8], Omni-MATH [6], MinervaMath [17], AGI-Eval-MATH [38], AMC23 [14], and AIME24.

Hyperparameter Details. The hyperparameters used by our method in the main experiments for Systems 1 and 2 are summarized in Table 9. These settings are used for the results reported in Table 1 and Figure 3.

Table 9: Hyperparameter settings used by our method in the main experiments for Systems 1 and 2, corresponding to the results in Table 1 and Figure 3.

Hyperparameter	System 1, controller: Qwen2.5-7B	System 2, controller: Qwen3-4B
Batch size $\lvert\mathcal{D}_b\rvert$	128	128
Group size G	4	4
Max prompt length	3072	3072
Max response length	2048	2048
Learning rate η	1e-6	1e-6
Total training iterations S	400	600
Sampling temperature for training	1.0	1.0
Sampling temperature for evaluation	0	0
Routing/Verification reward	1	1
Discount factor γ	1	1
Maximum number of turns T	3	3
Agent usage constraint $\rho_1/\rho_2/\rho_3$	0.25/0.5/1	0.25/0.5/1
Lagrangian multiplier $\xi_1/\xi_2/\xi_3$	0.25/0.125/0	0.25/0.125/0

D Proof of Proposition 3.1

Noting that $\sum_{k=1}^{K}\sum_{t=1}^{T}\mathbb{E}[\mathbf{1}(f(\mathbf{a}_t)=k)] \le T$. We first rewrite the constraint explicitly as:

$$\begin{aligned} &\underset{\boldsymbol{\theta}}{\text{maximize}}\ \ \mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\,\mathbb{E}_{\mathbf{a}_t\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)}\left[\sum_{t=1}^{T}\gamma^{t-1}R(\mathbf{s}_t,\mathbf{a}_t)\right], \\ &\text{subject to}\ \ \sum_{t=1}^{T}\mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\,\mathbb{E}_{\mathbf{a}_t\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)}\big[\mathbf{1}(f(\mathbf{a}_t)=k)\big] \le \rho_k\,T, \quad \forall\, k. \end{aligned} \tag{7}$$

Dualizing the constraints in (7) yields the Lagrangian relaxation [28, 29]:

$$\begin{aligned}\underset{\boldsymbol{\theta}}{\text{maximize}}\ \ & \mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\,\mathbb{E}_{\mathbf{a}_t\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)}\left[\sum_{t=1}^{T}\gamma^{t-1}R(\mathbf{s}_t,\mathbf{a}_t)\right] \\ & -\sum_{k=1}^{K}\xi_k\left(\sum_{t=1}^{T}\mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\,\mathbb{E}_{\mathbf{a}_t\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)}\big[\mathbf{1}(f(\mathbf{a}_t)=k)\big]-\rho_k\,T\right),\end{aligned} \tag{8}$$

where $\xi_k \geq 0$ is the Lagrange multiplier associated with the usage constraint on agent k, treated as a tunable hyperparameter. Dropping the constant term $\sum_{k=1}^{K}\xi_k\rho_k T$ and rearranging, we obtain:

$$\underset{\boldsymbol{\theta}}{\text{maximize}}\ \ \mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\,\mathbb{E}_{\mathbf{a}_t\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)}\left[\sum_{t=1}^{T}\gamma^{t-1}R(\mathbf{s}_t,\mathbf{a}_t)-\sum_{t=1}^{T}\sum_{k=1}^{K}\xi_k\,\mathbf{1}(f(\mathbf{a}_t)=k)\right]. \tag{9}$$

Under the reward definition in (1) and $\gamma = 1$, Problem (9) simplifies to:

$$\underset{\boldsymbol{\theta}}{\text{maximize}}\ \ \mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\,\mathbb{E}_{\mathbf{a}_t\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)}\left[\sum_{t=1}^{T}\left(0.5R_r(\mathbf{s}_t,\mathbf{a}_t)-\sum_{k=1}^{K}\xi_k\,\mathbf{1}(f(\mathbf{a}_t)=k)+0.5R_v(\mathbf{s}_t,\mathbf{a}_t)\right)\right]. \tag{10}$$

With the modified routing reward and composite reward defined in (3), we arrive at the final unconstrained objective (4). This completes the proof of Proposition 3.1.

E Algorithm Details

E.1 Problem (4) and GRPO

E.1.1 Policy Gradient of Problem (4)

Problem (4) is a finite-horizon discounted MDP over controller states and actions, and the immediate reward is given by the modified reward $R'(\mathbf{s}_t, \mathbf{a}_t)$. Therefore, its objective is

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{q}\sim\mathcal{D}}\mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=1}^{T}\gamma^{t-1}R'(\mathbf{s}_t,\mathbf{a}_t)\right]. \tag{11}$$

It admits the standard policy-gradient form. In particular, fixing the query $\mathbf{q}$, we have [33]

$$\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta}) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=1}^{T}\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(\mathbf{a}_t\mid\mathbf{s}_t)\,Q_t^{\pi}(\mathbf{s}_t,\mathbf{a}_t)\right]. \tag{12}$$

Following the multi-turn interaction procedure in Section 3.1, we introduce an additional index b to distinguish different rollout prefixes under group sampling. For each prefix state $\mathbf{s}_t^b$, we sample a group of G controller actions $\{\mathbf{a}_t^{(b,i)}\}_{i=1}^{G} \sim \pi_{\boldsymbol{\theta}_{\text{old}}}(\cdot\mid\mathbf{s}_t^b)$. Each sampled action yields an immediate modified reward

$$r_t^{(b,i)} = R'(\mathbf{s}_t^b, \mathbf{a}_t^{(b,i)}), \tag{13}$$

and induces a next state $\mathbf{s}_{t+1}^{(b,i)}$. We then recursively estimate the discounted return at state $\mathbf{s}_t^b$ as

$$\hat{V}_t(\mathbf{s}_t^b) = \frac{1}{G}\sum_{i=1}^{G}\left[r_t^{(b,i)}+\gamma\hat{V}_{t+1}(\mathbf{s}_{t+1}^{(b,i)})\right], \qquad \hat{V}_{T+1}(\cdot)=0. \tag{14}$$

Define the corresponding sample-based action-value estimate

$$\hat{Q}_t^{(b,i)} = r_t^{(b,i)} + \gamma\hat{V}_{t+1}(\mathbf{s}_{t+1}^{(b,i)}). \tag{15}$$

Then a GRPO-style estimated advantage is obtained by subtracting the group average,

$$A_t^{(b,i)} = \hat{Q}_t^{(b,i)} - \frac{1}{G}\sum_{j=1}^{G}\hat{Q}_t^{(b,j)}. \tag{16}$$

Using (16), we define the sample-based gradient estimator

$$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_{t=1}^{T} \frac{1}{G} \sum_{i=1}^{G} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b)\, A_t^{(b,i)}. \tag{17}$$

We next justify its relation to the standard policy gradient in (12). Substituting (16) into (17), we obtain

$$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_{t=1}^{T} \frac{1}{G} \sum_{i=1}^{G} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b) \left(\hat{Q}_t^{(b,i)} - \frac{1}{G} \sum_{j=1}^{G} \hat{Q}_t^{(b,j)} \right). \tag{18}$$

Taking expectation conditioned on $\mathbf{s}_t^b$, and using that the G actions are sampled i.i.d. from $\pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{s}_t^b)$, the cross terms (i.e., $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b) \hat{Q}_t^{(b,j)}, \forall j \neq i$) vanish since $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b)$ and $\hat{Q}_t^{(b,j)}$ are independent of each other and

$$\mathbb{E}\Big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b) \mid \mathbf{s}_t^b \Big] = 0.$$

Therefore, we have

$$\mathbb{E}\Big[\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \Big] = \frac{G-1}{G} \sum_{t=1}^{T} \mathbb{E}\Big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t \mid \mathbf{s}_t)\, \hat{Q}_t(\mathbf{s}_t, \mathbf{a}_t) \Big]. \tag{19}$$

Thus, the estimator in (17) is proportional to the standard policy gradient, with proportionality constant $(G-1)/G$. Equivalently, if one wishes to match the standard policy gradient in expectation exactly, one may rescale the estimator by $\frac{G}{G-1}$, yielding

$$\widehat{\nabla}_{\boldsymbol{\theta}}^{\text{corr}} J(\boldsymbol{\theta}) = \frac{G}{G-1} \sum_{t=1}^{T} \frac{1}{G} \sum_{i=1}^{G} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b)\, A_t^{(b,i)}. \tag{20}$$

In practice, this constant only rescales the gradient magnitude and does not change the ascent direction, so it can be absorbed into the learning rate. For simplicity, we therefore use the uncorrected form in (17), which is fully aligned with the GRPO-style group-relative advantage used later.

E.1.2 Policy Gradient in GRPO

We next show that optimizing a GRPO surrogate with the advantage in (16) is aligned with optimizing Problem (4).

First, we consider the conventional single-turn form of GRPO. Following [19], the simplified GRPO objective can be written as

$$\begin{aligned} &\max_{\boldsymbol{\theta}} \ \mathbb{E}_{\mathbf{q}\sim\mathcal{D}, \{\mathbf{y}_i\}_{i=1}^{G} \sim \pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{q})} \frac{1}{G} \sum_{i=1}^{G} \left\{ \min \left[\frac{\pi_{\boldsymbol{\theta}}(\mathbf{y}_i)}{\pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{y}_i)} A_i, \text{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(\mathbf{y}_i)}{\pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{y}_i)}, 1-\varepsilon, 1+\varepsilon \right) A_i \right] \right\}, \\ &\qquad \text{where } A_i = (r_i - \text{mean}(\{r_i\}_{i=1}^{G}))/\text{std}(\{r_i\}_{i=1}^{G}). \end{aligned} \tag{21}$$

By removing clipping (on-policy GRPO, $\boldsymbol{\theta} = \boldsymbol{\theta}_{\text{old}}$) and standard deviation normalization, the GRPO objective reduces to

$$\max_{\boldsymbol{\theta}} \ \mathbb{E}_{\mathbf{q}\sim\mathcal{D}, \{\mathbf{y}_i\}_{i=1}^{G} \sim \pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{q})} \frac{1}{G} \sum_{i=1}^{G} \left[\frac{\pi_{\boldsymbol{\theta}}(\mathbf{y}_i)}{\pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{y}_i)} A_i \right], \text{ where } A_i = r_i - \text{mean}(\{r_i\}_{i=1}^{G}). \tag{22}$$

We cast it into multi-turn form under our notation:

$$\mathcal{L}^{\text{GRPO}}(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{t=1}^{T} \frac{1}{G} \sum_{i=1}^{G} \frac{\pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b)}{\pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b)} A_t^{(b,i)} \right]. \tag{23}$$

At $\boldsymbol{\theta} = \boldsymbol{\theta}_{\text{old}}$, we have the following unbiased gradient estimator for $\mathcal{L}^{\text{GRPO}}(\boldsymbol{\theta})$:

$$\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{L}^{\text{GRPO}}(\boldsymbol{\theta}) = \sum_{t=1}^{T} \frac{1}{G} \sum_{i=1}^{G} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(b,i)} \mid \mathbf{s}_t^b)\, A_t^{(b,i)}. \tag{24}$$

Algorithm 1 GRPO-Based Training for the Multi-Turn Controller

Require: Controller policy $\pi_{\boldsymbol{\theta}}$, agent pool $\mathcal{M}$, training set $\mathcal{D}$, horizon T, group size G, discount factor γ, learning rate η

for iteration $= 1, \ldots, S$ **do**
  Sample a mini-batch $\mathcal{D}_b \subset \mathcal{D}$
  for each query $\mathbf{q} \in \mathcal{D}_b$ **do**
    Initialize the rollout tree with root $\mathbf{s}_1^{(1)} \leftarrow (\mathbf{q}, \mathbf{c})$ and active prefix set $\mathcal{B}_1 \leftarrow \{()\}$
    for $t = 1, \ldots, T$ **do**
      for each prefix $b \in \mathcal{B}_t$ **do**
        Sample $\{\mathbf{a}_t^{(b,i)}\}_{i=1}^{G} \sim \pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{s}_t^b)$
        for $i = 1, \ldots, G$ **do**
          Parse $m_t^{(b,i)} = f(\mathbf{a}_t^{(b,i)})$
          if $\mathbf{a}_t^{(b,i)}$ invokes an agent **then**
            Query agent $m_t^{(b,i)}$ to obtain response $\mathbf{g}_t^{(b,i)}$
            Set $\mathbf{s}_{t+1}^{(b,i)} \leftarrow (\mathbf{q}, \mathbf{c}', \mathbf{g}_t^{(b,i)})$
          else
            Mark child (b, i) as terminal
          end if
          Compute $r_t^{(b,i)} = R'(\mathbf{a}_t^{(b,i)}, \mathbf{s}_t^b)$
        end for
      end for
      Set $\mathcal{B}_{t+1} \leftarrow \{(b, i) : b \in \mathcal{B}_t,\ i = 1, \ldots, G,\ \mathbf{s}_{t+1}^{(b,i)} \text{ is non-terminal}\}$
    end for
    Set $\hat{V}_{T+1}(\cdot) \leftarrow 0$
    for $t = T, \ldots, 1$ **do**
      for each prefix $b \in \mathcal{B}_t$ **do**
        Compute $\hat{Q}_t^{(b,i)} = r_t^{(b,i)} + \gamma \hat{V}_{t+1}(\mathbf{s}_{t+1}^{(b,i)})$ for $i = 1, \ldots, G$
        Compute $\hat{V}_t(\mathbf{s}_t^b) = \frac{1}{G} \sum_{i=1}^{G} \hat{Q}_t^{(b,i)}$
        Compute $A_t^{(b,i)} = \hat{Q}_t^{(b,i)} - \frac{1}{G} \sum_{j=1}^{G} \hat{Q}_t^{(b,j)}$ for $i = 1, \ldots, G$
      end for
    end for
    Calculate the gradient $\widehat{\nabla} \mathcal{L}_{\mathbf{q}}^{\mathrm{GRPO}}(\boldsymbol{\theta})$ over the nodes $\{(\mathbf{s}_t^b, \mathbf{a}_t^{(b,i)}, A_t^{(b,i)})\}_{t,b,i}$ in the rollout traces $\mathcal{B}_T$, and scale it by $1/|\mathcal{B}_T|$, following (24) (or (17))
  end for
  Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \nabla_{\boldsymbol{\theta}} \frac{1}{|\mathcal{D}_b|} \sum_{\mathbf{q} \in \mathcal{D}_b} \widehat{\nabla} \mathcal{L}_{\mathbf{q}}^{\mathrm{GRPO}}(\boldsymbol{\theta})$
end for

As we can see, (24) is exactly the same estimator as (17). Therefore, optimizing the on-policy GRPO surrogate, without standard deviation normalization, yields the same update direction as optimizing our discounted multi-turn objective up to a constant factor $(G-1)/G$, which can be absorbed into the learning rate. Hence, our discounted multi-turn objective can be optimized using the standard GRPO framework after replacing the trajectory-level reward in conventional GRPO by the recursively computed discounted return estimate and removing the standard deviation normalization.

E.2 Practical On-Policy GRPO Procedure for Problem (4)

The previous subsection establishes that, in our setting, the on-policy GRPO gradient is aligned with the policy gradient of Problem (4) when the conventional trajectory-level reward is replaced by the recursively computed discounted return and standard deviation normalization is omitted. We now summarize the resulting practical training procedure used in our method in Algorithm 1. For each query, training constructs a multi-turn rollout tree rooted at the initial state $\mathbf{s}_1 = (\mathbf{q}, \mathbf{c})$: each active prefix state $\mathbf{s}_t^b$ branches into G sampled controller actions, the selected agent responses determine the child states, and the modified reward $R'(\mathbf{a}_t^{(b,i)}, \mathbf{s}_t^b)$ is computed at every node. The algorithm

then performs a backward recursion over the tree to compute $\hat{Q}_t^{(b,i)}$, $\hat{V}_t(\mathbf{s}_t^b)$, and the group-relative advantages $A_t^{(b,i)}$, which are finally aggregated into the GRPO surrogate for updating $\boldsymbol{\theta}$.

F Case Study

We present representative case studies illustrating how the controller coordinates multi-turn interactions with agents through verification, feedback, and re-routing decisions. These examples are based on System 1, where the controller is initialized from Qwen2.5-7B-Base and the candidate agents are Qwen3-30B-A3B-Instruct, Qwen2.5-7B-Instruct, and Qwen2.5-1.5B-Instruct.

Case Study: Multi-Turn Controller-Agent Interaction

Example 1: Exit in 2 Turns

Question: The points (x, y) represented in this table lie on a straight line. The point $(13, q)$ lies on the same line. What is the value of $p + q$?

Turn 1 (Controller): <thinking>This is a linear equation problem involving finding the slope and using it to determine the value of p and q. It is a medium-difficult problem because it requires understanding of linear equations and the concept of slope. The problem involves finding the slope between two points, using the slope to find the value of p, and then using the value of p to find q. Given the need for multiple steps and the application of algebraic concepts, I will route this problem to the Qwen2.5-7B-Instruct model, which is suitable for handling medium-difficult mathematical problems involving linear equations and algebraic manipulations.</thinking>
<model>Qwen2.5-7B-Instruct</model>

Response from Qwen2.5-7B-Instruct: To determine the value of $p + q$, we first need to find the equation of the line that passes through the given points. The points given are $(2, -5)$, $(p, -14)$, and $(p + 2, -17)$.
First, we calculate the slope of the line using the points $(2, -5)$ and $(p, -14)$:

$$\text{slope} = \frac{-14 - (-5)}{p - 2} = \frac{-9}{p - 2}$$

Next, we use the points $(2, -5)$ and $(p + 2, -17)$:

$$\text{slope} = \frac{-17 - (-5)}{(p + 2) - 2} = \frac{-12}{p}$$

Since both expressions represent the slope of the same line,

$$\frac{-9}{p - 2} = \frac{-12}{p}$$

Cross-multiplying gives:

$$-9p = -12(p - 2)$$
$$-9p = -12p + 24$$
$$3p = 24$$
$$p = 8$$

Using the point $(2, -5)$ and slope $-\frac{3}{2}$, the equation is:

$$y - (-5) = -\frac{3}{2}(x - 2)$$
$$y + 5 = -\frac{3}{2}x + 3$$
$$y = -\frac{3}{2}x - 2$$

When $x = 13$,

$$q = -\frac{3}{2}(13) - 2 = -\frac{39}{2} - \frac{4}{2} = -\frac{43}{2}$$

Finally,

$$p + q = 8 - \frac{43}{2} = \frac{16}{2} - \frac{43}{2} = -\frac{27}{2}$$

Thus, the value of $p + q$ is $\boxed{-\frac{27}{2}}$.

Turn 2 (Controller): <checking> Let's trace the proposed solution step by step and verify its correctness.
1. Calculate the slope using the points (2, -5) and (p, -14):

$$\text{slope} = \frac{-14 - (-5)}{p - 2} = \frac{-9}{p - 2}$$

2. Calculate the slope using the points (2, -5) and (p+2, -17):

$$\text{slope} = \frac{-17 - (-5)}{(p+2) - 2} = \frac{-12}{p}$$

3. Set the slopes equal to each other and solve for p:

$$\frac{-9}{p-2} = \frac{-12}{p}$$

Cross-multiplying gives:

$$-9p = -12(p-2)$$

$$-9p = -12p + 24$$

$$3p = 24$$

$$p = 8$$

4. Verify the equation of the line using the point (2, -5) and the slope $-\frac{3}{2}$:

$$y + 5 = -\frac{3}{2}(x-2)$$

$$y = -\frac{3}{2}x + 3 - 5$$

$$y = -\frac{3}{2}x - 2$$

5. Find the value of q when $x = 13$:

$$q = -\frac{3}{2}(13) - 2 = -\frac{39}{2} - 2 = -\frac{39}{2} - \frac{4}{2} = -\frac{43}{2} = -21.5$$

6. Calculate $p + q$:

$$p + q = 8 + (-21.5) = -13.5$$

`</checking>`

`<verdict>True</verdict>`

Example 2: Exit in 3 Turns

Question: Rationalize the denominator: $\frac{1}{2\sqrt{7}}$.

Turn 1 (Controller): `<thinking>`This is a simple problem involving rationalizing the denominator. It is a basic algebraic manipulation problem that does not require advanced mathematical knowledge or complex problem-solving skills. Therefore, I will route this problem to the Qwen2.5-1.5B-Instruct model, which is suitable for easy problems.`</thinking>`
`<model>`Qwen2.5-1.5B-Instruct`</model>`

Response from Qwen2.5-1.5B-Instruct: To rationalize the denominator of the expression $\frac{1}{2\sqrt{7}}$, we need to eliminate the square root from the denominator. We can do this by multiplying both the numerator and the denominator by the conjugate of the denominator. The conjugate of $2\sqrt{7}$ is $2\sqrt{7}$ itself, because it is the same number.
Step 1: Multiply the numerator and the denominator by $2\sqrt{7}$

$$\frac{1}{2\sqrt{7}} \cdot \frac{2\sqrt{7}}{2\sqrt{7}}$$

Step 2: Simplify the expression.

$$\frac{2\sqrt{7}}{2\sqrt{7} \cdot 2\sqrt{7}}$$

Step 3: Simplify the denominator.

$$\frac{2\sqrt{7}}{2 \cdot 7}$$

Step 4: Simplify the fraction.

$$\frac{2\sqrt{7}}{14}$$

Step 5: Simplify further by dividing both the numerator and the denominator by 2.

$$\frac{\sqrt{7}}{7}$$

Therefore, the rationalized form of the expression is $\boxed{\frac{\sqrt{7}}{7}}$.

Turn 2 (Controller): `<checking>` Let's trace the proposed solution step by step and verify its correctness.
1. The given expression is $\frac{1}{2\sqrt{7}}$.
2. To rationalize the denominator, we multiply the numerator and the denominator by $2\sqrt{7}$.
3. This gives us $\frac{2\sqrt{7}}{2\sqrt{7}\cdot 2\sqrt{7}} = \frac{\sqrt{7}}{2\cdot 7} = \frac{\sqrt{7}}{14}$.
4. The proposed solution seems to have an error in step 4.
We need to route the problem to a stronger agent to refine the solution. Qwen2.5-7B-Instruct is good choice, which is stronger than Qwen2.5-1.5B-Instruct.

`</checking>`

`<verdict>False</verdict>`

`<model>Qwen2.5-7B-Instruct</model>`

Response from Qwen2.5-7B-Instruct: To rationalize the denominator of the expression $\frac{1}{2\sqrt{7}}$, we eliminate the square root from the denominator as follows:
Identify the denominator: The denominator is $2\sqrt{7}$.
Multiply by a suitable factor: To rationalize the denominator, multiply both the numerator and denominator by $\sqrt{7}$:

$$\frac{1}{2\sqrt{7}} \cdot \frac{\sqrt{7}}{\sqrt{7}}$$

Simplify the expression:

$$\frac{1 \cdot \sqrt{7}}{2\sqrt{7} \cdot \sqrt{7}} = \frac{\sqrt{7}}{2 \cdot 7}$$

Simplify the denominator:

$$2 \cdot 7 = 14$$

Final result:

$$\frac{\sqrt{7}}{14}$$

Thus, the rationalized form of $\frac{1}{2\sqrt{7}}$ is

$$\boxed{\frac{\sqrt{7}}{14}}.$$

Turn 3 (Controller): `<checking>` Let's trace the proposed solution step by step and verify its correctness.
1. The given expression is $\frac{1}{2\sqrt{7}}$.
2. To rationalize the denominator, we multiply the numerator and the denominator by $\sqrt{7}$.
3. This gives us $\frac{\sqrt{7}}{2\sqrt{7}\cdot\sqrt{7}} = \frac{\sqrt{7}}{2\cdot 7} = \frac{\sqrt{7}}{14}$.
4. The proposed solution is correct.

`</checking>`

`<verdict>True</verdict>`

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction (Section 1) clearly state the claims made, including the contributions made in the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitation is included in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: All assumptions for the theoretical results are clearly stated in Section 3.3, and the proofs are provided in Appendix D.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper discloses all necessary information to reproduce the main experimental results in Section 4

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All the datasets used in this work are open-sourced while the information about the models and parameters are clearly reported in Section 4. The implementation code of our project is included in the supplementary material of our submission.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The details about the setting of our experiments are reported in Section 4.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We do not include error bars because the gains are large and consistent across seven datasets and two system configurations. Since evaluation is conducted on fixed benchmark test sets, the reported improvements are unlikely to be explained by random variation alone.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the information of the computer resources used for this work including the GPU type and memory information in Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our work adheres to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The focus of this work is on the technical advancement of multi-agent coordination. While this research has potential societal impacts, it primarily addresses technical challenges and does not necessitate a specific discussion on societal consequences.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cited the original paper that produced datasets used in our work in Section 4.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: Our paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Our paper does not involve research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.