Scaling Behavior of Single LLM-Driven Multi-Agent Systems

Anonymous ACL submission

Abstract

The burgeoning field of LLM-based Multi-Agent Systems (MAS) promises to tackle complex tasks through collaborative intelligence, yet fundamental questions regarding their scaling behavior and intrinsic collective dynamics remain underexplored. This paper systematically investigates how the performance of a homogeneous MAS evolves as the number of agents increases, isolating the variable of collaboration from model or knowledge heterogeneity. We propose the Sequential Iterative Multi-Agent System (SIMAS) framework, a minimalist architecture centered on sequential inter-agent communication, to clearly observe scaling effects. Through extensive experiments across diverse tasks and model scales, we establish that MAS performance does not scale monotonically with agent count but follows a pattern of diminishing returns, governed by a trade-off between collaborative synergy and coordination overhead. Our findings reveal that effective MAS requires a sufficiently capable base LLM, that task type critically modulates the optimal agent count, and that collective intelligence is an emergent property contingent on strategic interaction design rather than a guaranteed outcome of agent plurality. This work provides a foundational understanding of MAS scaling laws, offering practical guidance for designing efficient collaborative systems and challenging the prevailing assumption that more agents invariably lead to better performance.

1 Introduction

In recent years, Large Language Models (LLMs) have demonstrated remarkable capabilities in text generation, complex reasoning, and decision-making, establishing themselves as the core foundation for constructing intelligent systems, often referred to as "agents". Though individual agent has exhibited excellent problem-solving ability in expansive fields with performance enhanced—for instance, through Chain-of-Thought (CoT) prompting (Wei et al., 2022) to elicit step-by-step reasoning, or by enabling models to leverage external APIs via frameworks like Toolformer (Schick et al., 2023), however, many real-world challenges, such as sophisticated software development or multi-faceted problem-solving, inherently require collaborative efforts. This necessity has driven the emergence of LLM-based Multi-Agent Systems (MAS), a field that has rapidly evolved from early exploratory frameworks to complex systems (Xi et al., 2023; Luo et al., 2025), where multiple agents interact to achieve common goals.

In MAS, collaboration enables agents to share knowledge, assign specialized roles, and integrate their complementary strengths, thereby tackling tasks beyond the scope of any single agent—ranging from handling extensive contexts (Zhang et al., 2025) to complex social simulations (Qian et al., 2023). This vision is supported by a growing series of frameworks: AutoGen (Wu et al., 2023) facilitates customizable agent conversations, CAMEL (Li et al., 2023) explores role-playing for idea exploration, and MetaGPT (Hong et al., 2023) assigns standardized workflows to simulate a software company. These systems often deploy agents as "reasoning experts" or "fact-checkers" to boost collective output. However, despite these engineering advances and recent efforts to probe the robustness of agent communication (He et al., 2025), a fundamental scientific understanding of MAS collective behavior remains under-explored. *First*, in order to isolate the scaling effects of collaboration without the confounding variable of model heterogeneity, we ask: when all agents are built using the same underlying language model, does increasing the number of agents consistently improve the system's performance? *Second*, are there common patterns in how these systems operate, regardless of how they are designed? *Third*, how do the collaboration of MAS differ from the reasoning of a

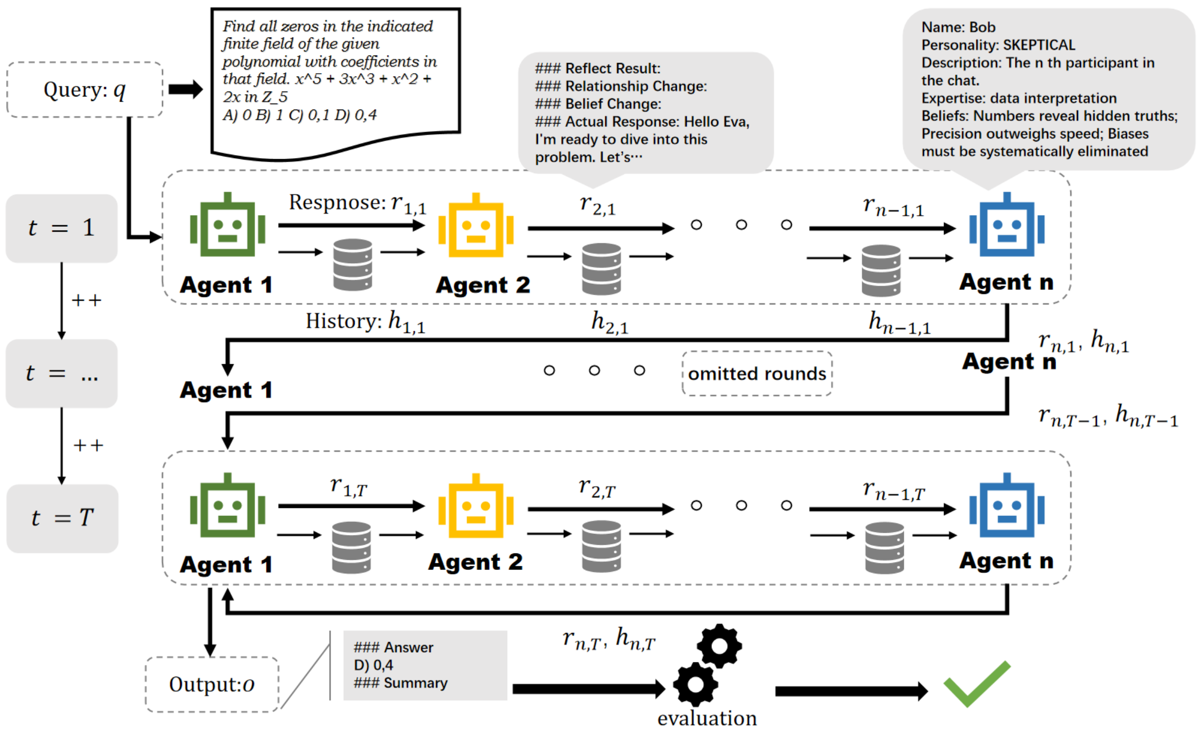


Figure 1: Workflow of the SIMAS. A group of n agents $(a_1, a_2, ..., a_n)$, each configured with a distinct profile (personality, core beliefs, expertise), engages in T rounds of sequential discussion. In each round, every agent generates a response based on the progressively accumulated conversation history $h_{i-1,t}$, which monotonically expands to $h_{n,t}$. After T rounds, the first agent a_1 synthesizes the final output o from the complete history $h_{n,T}$.

single agent?

To provide a systematic analyisis of the scaling laws in collaboration of MAS, we establish a investigation into how the performance of MAS evolves as the number of agents increases. We propose the Sequential Iterative Multi-Agent System (SIMAS) framework, which strips away scenario-specific components and retains only sequential inter-agent communication as its core mechanism. This design minimizes architectural complexity, enabling clear observation of performance scaling with agent count and identification of general patterns in MAS. Through a rigorous comparison between MAS and single-agent baselines, we aim to quantitatively delineate the boundaries where collaboration yields positive returns versus where it introduces diminishing returns.

Our investigation yields several principal findings: *Model capability is a prerequisite for effective MAS.* We observe that only models above a certain performance threshold (typically larger-scale models) can effectively power MAS. *LLM-based MAS scalability is governed by a fundamental trade-off between collaborative synergy and coordination overhead.* This reframes the goal from "more agents" to "optimal number," highlighting that excessive collaboration can be detrimental, particularly in reasoning-intensive tasks where focus and coherence are critical. *Collective intelligence in MAS is an emergent property contingent on interaction architecture, not an automatic outcome of agent plurality.* Without strategic architectural design, an MAS risks achieving only the illusion of collaboration while failing to surpass the capabilities of a well-prompted individual.

2 Related Works

2.1 Architectures for Multi-Agent Collaboration

The foundation of MAS lies in defining how agents interact to extend the capabilities of individual LLM. Early research focused on unstructured communicative frameworks. For instance, CAMEL (Li et al., 2023) introduced "inception prompting" to facilitate autonomous role-playing, demonstrating that distinct personas can guide solution exploration. Building on this, AutoGen (Wu et al., 2023) provided a flexible infrastructure allowing for arbitrary graph topologies, enabling dynamic conversations between human and agent proxies. While these flexible frameworks encourage creativity, recent work has shifted towards structured collaboration to enhance reliability. MetaGPT (Hong et al., 2023) incorporates Standard Operating Procedures (SOPs) from human organizations, assigning rigid roles (e.g., Product Manager, Engineer) to streamline complex workflows like software generation. Similarly, AgentVerse (Chen et al., 2023) and the recent Agent-Pro (Zhang et al., 2024b) introduce iterative optimization mechanisms, where agents engage in "reflect-and-refine" loops to correct errors dynamically.

More recently, research has begun to address the specific limitations of LLMs through collaboration. (Zhang et al., 2025) proposed *Chain of Agents*, which aggregates information across multiple agents to handle long-context tasks that overwhelm single models. However, while these architectures demonstrate that collaboration can improve performance, they predominantly focus on engineering novel interaction patterns. They often overlook the fundamental scaling laws of these interactions: specifically, whether adding more agents consistently yields better results when the underlying model remains constant, a gap our study aims to address.

2.2 Domain-Specialized Multi-Agent Systems

Beyond general frameworks, significant research serves to adapt MAS to domain-specific constraints, utilizing specialized knowledge bases and verification tools. In software engineering, ChatDev (Qian et al., 2023) mimics the waterfall model of software development, proving that decomposing coding tasks into sequential sub-tasks significantly reduces bug rates compared to monolithic generation. In scientific discovery, systems like SciAgents (Yang et al., 2024b) and VirSci (Fan et al., 2024) simulate academic review processes, leveraging the "wisdom of the crowd" to refine hypotheses. The medical and financial fields have seen similar adaptations; for example, Agent Hospital (Li et al., 2024) utilizes simulation-based training to improve diagnostic accuracy, while FinCon (Zhang et al., 2024a) employs adversarial debates to mitigate risk in financial decision-making.

A critical analysis of these domain-specific works reveals a common trend: performance gains are often attributed to the injection of domain knowledge or tool use rather than the collective intelligence of the agents themselves. Furthermore, as highlighted in recent surveys (Luo et al., 2025), these systems utilize heterogeneous agents (different prompts/models), making it difficult to isolate the source of improvement. Recent "Red-Teaming" studies (He et al., 2025) have even suggested that complex communication structures can introduce new vulnerabilities or coordination overheads. Unlike these application-driven studies, our work isolates the variable of agent quantity within a controlled, homogeneous environment. This allows us to disentangle the benefits of collaboration from the benefits of domain specialization, offering a clearer view of the intrinsic scaling properties of LLM-based MAS.

Symbol	Definition
$\mathcal{Q}$	Input question space
q	A specific input question instance ($q \in \mathcal{Q}$)
$\mathcal{A}$	The set of agents $\{a_1, a_2, \ldots, a_n\}$
n	Total number of agents
T	Total number of discussion rounds
$\mathcal{H}$	Conversation history space
$h_{i,t}$	Specific conversation history after agent a_i in round t
Π	Response generation function (LLM)
$\mathcal{O}$	Output space
o	Final answer instance ($o \in \mathcal{O}$)

Table 1: Notations used in the SIMAS framework.

3 Methodology

3.1 Preliminaries

To provide a clear overview of our research question, we first define the notations and the mathematical formulation of the Multi-Agent System (MAS). The key symbols used throughout this paper are listed in Table 1.

Definition 1 (Multi-Agent System Formulation)**.** *The multi-agent system $\mathcal{M}$ is formally defined as a sextuple:*

$$\mathcal{M} = \langle \mathcal{A}, \mathcal{Q}, \mathcal{H}, T, \Pi, \mathcal{O} \rangle$$

where $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ denotes the set of agents, with $n \in \mathbb{N}^+$ representing the number of agents; $\mathcal{Q}$ denotes the input question space; $\mathcal{H}$ denotes the conversation history space; $T \in \mathbb{N}^+$ denotes the number of discussion rounds; $\Pi : \mathcal{H} \times \mathcal{A} \to \mathcal{R}$ denotes the response generation function, where $\mathcal{R}$ is the response space; and $\mathcal{O} : \mathcal{H} \to \mathcal{A}$ denotes the final answer synthesis function.

3.2 The SIMAS Framework

Based on the formulation above, we designed a specific implementation named Sequential Iterative Multi-Agent-System (SIMAS). The process is

Algorithm 1 Multi-Agent Collaborative Process

Require: Question $q \in \mathcal{Q}$, agent set $\mathcal{A}$, rounds T
Ensure: Final answer o

1: Initialize history $h_0 \leftarrow \{q\}$
2: **for** round $t = 1$ to T **do**
3: **for** each agent $a_i \in \mathcal{A}$ **do**
4: $r_{i,t} \leftarrow \Pi(h_{i-1,t}, a_i)$ ▷ Generate response based on current history
5: $h_{i,t} \leftarrow h_{i-1,t} \cup \{r_{i,t}\}$ ▷ Update conversation history
6: **end for**
7: **end for**
8: **return** $\Pi(h_{n,T}, a_1)$ ▷ First agent generates final answer based on complete history

structured as follows: The discussion is conducted over T rounds (default 3 rounds). In each round, every agent generates a response based on the original question and the entire conversation history from all previous agents and rounds.

Property 1 (Sequential Communication)**.** *The system employs sequential inter-agent communication within each round:*

$$\forall t \in [1, T], \quad h_{n,t} = h_{1,t} \cup \bigcup_{i=2}^{n} \{\Pi(h_{i-1,t}, a_i)\}$$

Property 2 (Progressive History Accumulation)**.** *The conversation history grows monotonically in one round:*

$$\forall t \in [1, T], h_{1,t} \subset h_{2,t} \subset \cdots \subset h_{n,t}$$

After all agents have responded in a round, the process repeats for the next round, ensuring each agent has multiple opportunities to contribute to the discussion. The final answer is synthesized by the first agent based on the complete conversation history. The whole process of our MAS can be seen in Alg. 1 and Figure 1.

Definition 2 (System Output)**.** *For an input question $q \in \mathcal{Q}$, the system output is defined as:*

$$o = f_{\mathcal{M}}(q) = \Pi_{output}(h_{n,T}, a_1)$$

where $h_{n,T} \in \mathcal{H}$ represents the complete conversation history after T rounds of discussion.

3.3 Setting of Single Agent

In this experiment, each agent $a_i \in \mathcal{A}$ was driven by LLM and assigned descriptions, personalities, beliefs, and strengths, with the descriptions set as "the i-th assistent in the group chat" and therefore not considered as influencing factors. All agents are powered by the same LLM, sharing identical model parameters and knowledge base. To introduce diversity in reasoning styles, each agent is assigned a unique profile before the discussion begins.

The profile consists of three attributes: ***personality, core beliefs, and expertise***. ***Personality*** is a general disposition (e.g., "Skeptical" or "Aggressive") that guides the tone and focus of their reasoning, ***expertise*** is a specific domain of knowledge (e.g., "Data Interpretation" or "Logical Reasoning") they are instructed to prioritize, and ***core beliefs*** are a set of principles (e.g., "Precision outweighs speed") that influence their approach to problem-solving.

These attributes are automatically generated by the language model for each agent to simulate a group of diverse experts. The specific configurations used in our experiments are detailed in the Appendix.

Definition 3 (Agent Configuration)**.** *Each agent $a_i \in \mathcal{A}$ is configured as a quadruple:*

$$a_i = \langle LLM, d_i, p_i, b_i, e_i \rangle$$

where LLM represents the underlying language model (shared across all agents), $d_i \in \mathcal{D}$ denotes the description, $p_i \in \mathcal{P}$ denotes the personality trait, $b_i \in \mathcal{B}$ denotes the core beliefs, $e_i \in \mathcal{E}$ denotes the expertise, with $\mathcal{D}, \mathcal{P}, \mathcal{B}, \mathcal{E}$ representing the respective attribute spaces.

Property 3 (Model Homogeneity)**.** *All agents share the same underlying language model:*

$$\forall a_i, a_j \in \mathcal{A}, \quad LLM(a_i) = LLM(a_j)$$

Property 4 (Configuration Heterogeneity)**.** *Agents achieve diversity through distinct configuration parameters:*

$$\forall a_i, a_j \in \mathcal{A}, i \neq j \Rightarrow (d_i, p_i, b_i) \neq (d_j, p_j, b_j)$$

4 Experiments

4.1 Experimental Setup

To systematically examine the scaling effects of agent collaboration, we vary the number of agents n from 1 to 8 in increments of 1. This granular range allows us to capture the nuanced relationship between team size and performance. Regarding agent configurations, we established a baseline

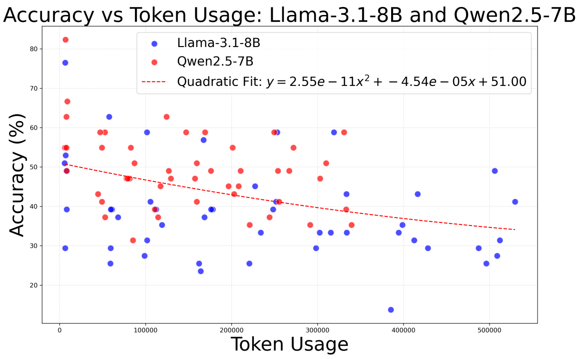


Figure 2: Result of models with small parameters on all subjects

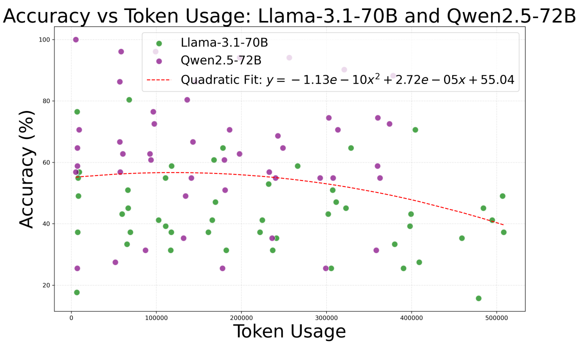


Figure 3: Result of models with large parameters on all subjects

condition where agents possess personality traits, expertise, and beliefs.

For performance evaluation, we employ three complementary benchmarks, each targeting distinct cognitive capabilities: **The Massive Multitask Language Understanding (MMLU) dataset** (Hendrycks et al., 2021), covering 57 diverse subjects across STEM, humanities, and social sciences, to assess general knowledge and broad reasoning capabilities. **The E-KAR dataset** (Luo et al., 2024), specifically designed for evaluating analogical reasoning through entity-based knowledge graph analogies. **The AIME 2025** (Mathematical Association of America, 2025) competition problems, consisting of advanced mathematical reasoning challenges, which we use in Section 5 to compare MAS against single-agent reasoning on highly complex tasks.

From MMLU, we select six representative disciplines (see Table 3) and sample 17 questions each to balance efficiency with robustness. Each experimental configuration is repeated three times, with results averaged to ensure stability.

We employ two widely-used, open-source model families: Llama-3.1 (AI, 2024) and Qwen2.5 (Yang et al., 2024a). To understand the impact of model scale, we use versions with 8B/7B and 70B/72B parameters from each family. These models provide a strong baseline for general-purpose reasoning, allowing for a clear analysis of multi-agent scaling effects with balance of capabilities and generation speed.

To isolate the contribution of each attribute, we compare this baseline against four alternative configurations: one lacking all three attributes, and three configurations each removing exactly one of the three attributes This structured ablation enables us to analyze how each factor contributes to system performance and stability. For these ablation studies, we specifically use the Llama-3.1-70B model with agent number $n \in \{2, 4, 6, 8\}$, chosen to represent key points in the scaling range while reducing experimental overhead.

4.2 Experimental Results and Analysis

To investigate the scaling behavior of MAS performance $A(\mathcal{M}_n)$ with respect to agent count n, we first establish a criterion to distinguish productive collaboration from mere agent plurality.

Definition 4 (sufficiently driven)**.** *For a given task T with evaluation metric $A : \mathcal{M} \to [0, 1]$ measuring accuracy, let $A(\mathcal{M}_n)$ denote the accuracy of the system with n agents.*

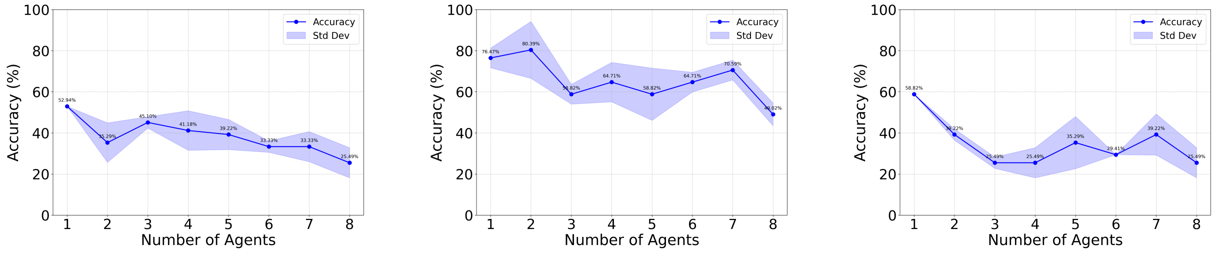
Given a set of experimental conditions C held constant, $\mathcal{M}$ is said to be sufficiently driven *if and only if:*

$$\exists\, n_1, n_2 \in \mathbb{N},\ n_1 \neq n_2,\ n_1, n_2 \geq 2$$

$$A(\mathcal{M}_{n_1}) \geq A(\mathcal{M}_1) \quad \textit{and} \quad A(\mathcal{M}_{n_2}) \geq A(\mathcal{M}_1)$$

where the comparison is performed under identical conditions C.

A system failing this condition exhibits monotonic degradation, indicating an inability to leverage collaboration. Our experiments reveal this state is intrinsically linked to base model capability, as small-scale models (7B/8B parameters) consistently fail to meet this condition, showing monotonic performance decay with increasing n (Figure. 2). As for a sufficiently driven MAS, intuitively, with more agents it should show better performance for its adequate collaboration fully utilizing model capability. But according to the overall experimental results, the actual situation is not the case.



(a) Performance on Abstract Algebra (b) Performance on Philosophy (c) Performance on Formal Logic

Figure 4: Task-type modulation of scaling on Llama-3.1-70B. Reasoning tasks suffer sharp declines post-peak; knowledge tasks show more tolerance.

Finding 1

If a MAS is sufficiently driven, the performance will scale with diminishing returns, following an inverted-U relationship with agent count.

For sufficiently driven MAS, a clear scaling law emerges. Performance initially improves due to collaborative synergy—diverse perspectives and mutual error correction—but peaks at an optimal n^*. Beyond this point, coordination overhead, manifested as information redundancy and conflicting reasoning paths, dominates and causes performance to decline. This overhead is quantifiable, with computational cost (token count) scaling quadratically with n within the SIMAS architecture, rendering large teams inefficient, as is shown in Figure 3, which presents the results of large-parameter models as well as the sufficiently driven SIMAS across all subjects, demonstrating the relationship between accuracy and token usage in MAS. The precise location of n^* and the steepness of the decline are not universal but are modulated by several factors.

To illustrate the impact of model scale, we compared the experimental results of large-parameter models and small-parameter models across all disciplines, with a particular focus on the representative case of College Physics for detailed analysis. Results are demonstrated in Figure 2 and 3. More detailed information is shown in the Appendix B.2.

Finding 2

Model scale provides necessary conditions for effective multi-agent collaboration, but does not ensure stable performance gains alone.

Crucially, comparing Figure 2 and Figure 3, it

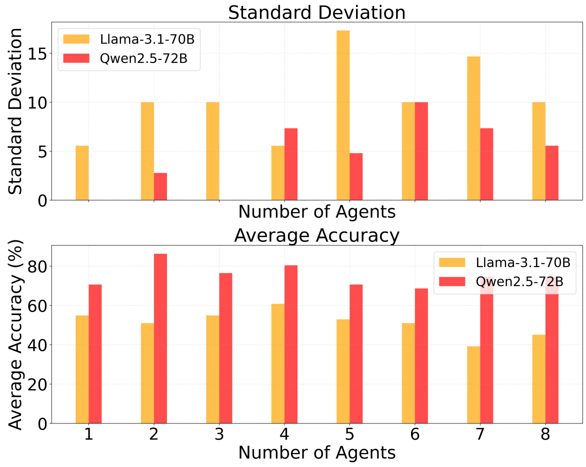


Figure 5: Model Type Impact on College Physics for Models with Large Parameters

is obviously revealed that smaller models (7B/8B parameters) proved fundamentally inadequate for multi-agent collaboration, showing monotonic performance degradation with additional agents, suggesting a minimum capability threshold for effective multi-agent interactions. While larger models consistently outperformed their smaller counterparts in absolute accuracy, this advantage did not uniformly extend to stability metrics. The Qwen2.5-72B model demonstrated also high stability, but in contrast, Llama-3.1-70B showed significant volatility despite its parameter advantage, with standard deviations reaching 17.32%, indicating that scale alone cannot ensure consistent multi-agent performance. (see Figure 8 and 9 in Appendix B.2)

Finding 3

Task type dictates the optimal agent number size and tolerance for scaling.

We use the Llama-3.1-70B model for analyzing the impact of task type. Figures 4a to 4c illustrate that the impact of agent scaling is fundamentally mediated by task characteristics. In logical reasoning domains such as abstract algebra and formal logic, we observed consistent and substantial performance degradation as agent numbers increased, with accuracy dropping by 27.45% from 1 to 8 agents. This degradation stems from the inherent sensitivity of logical reasoning chains to disruptions caused by information redundancy and inconsistent reasoning paths across multiple agents. Conversely, fact retrieval tasks like global facts and philosophy demonstrated more complex, non-linear patterns where initial performance declines could be partially mitigated through knowledge diversity at higher agent counts. Complete results across seven task types are provided in Appendix B.3.

Finding 4

When the number of agents in MAS is overly large, it will exhibit "pseudo stability" due to its poor performance.

We also analyze output stability across experimental runs. Generally, stability decreases as agent count increases, reflecting the growing complexity and unpredictability of interactions. However, when the number of agents grows beyond a certain point, its stability may paradoxically improve, as shown in Figures 4a, 4b, and 4c—a state we term *pseudo-stability*. This occurs not due to robust collaboration, but because the system has entered a regime of consistent failure.

To showcase the impact of model type, we select results of Llama3.1-70B and Qwen2.5-72B on College Physics for demonstration, as shown in Figures 5. While the inverted-U pattern is consistent across model families, the specific n^*, peak performance, and output stability vary. This indicates that model architecture influences collaboration efficiency beyond mere parameter count.

Furthermore, ablation studies on agent profiles reveal that attributes like personality, belief, and expertise can shift absolute performance levels and stability metrics, but they do not alter the fundamental inverted-U scaling law. This underscores that the law is governed by the interaction dynamics inherent to collaboration itself.The analysis is detailed in Appendix B.5

5 Comparison with Reasoning

We now benchmark our SIMAS framework against a strong single-agent baseline: CoT prompting.

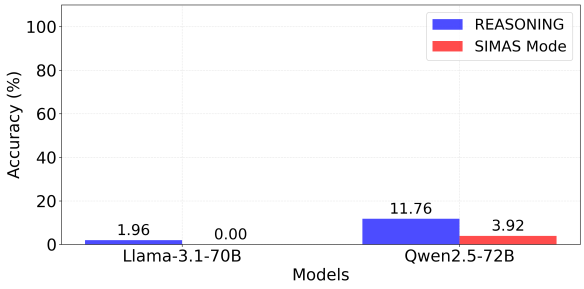


Figure 6: CoT vs. SIMAS on AIME 2025. SIMAS fails catastrophically, highlighting its inadequacy for complex, multi-step reasoning.

This baseline utilizes the same underlying LLM to produce a step-by-step reasoning trace followed by a final answer within a single, focused pass. This comparison tests whether minimalist multi-agent collaboration reliably outperforms focused, single-agent reasoning from the same underlying LLM. The concrete prompts are exhibited in Appendix C.1.

Finding 5

For reasoning tasks, the primary failure mode of minimalist MAS is the fragmentation of coherent thought.

On reasoning-intensive benchmarks such as abstract algebra (Figure. 13a) and AIME 2025 (Figure. 6), CoT consistently matches or significantly surpasses the best SIMAS configuration. The performance gap widens with problem complexity, with SIMAS failing catastrophically on the challenging AIME problems. This failure mode stems from the fragmentation of coherent reasoning chains across sequential agent turns without a mechanism for synthesis and correction. In complex, multi-step reasoning, maintaining a consistent logical thread is crucial. In SIMAS, each agent builds upon the incomplete and potentially divergent outputs of its predecessors. Without a dedicated mechanism to synthesize these fragments into a unified line of reasoning or to correct accumulating errors and inconsistencies, the collective output often becomes disjointed or drifts from the correct solution path.

On open-ended generation (Figure 13c) and coding (Figure 7) tasks, results are mixed and model-dependent. For instance, on coding tasks, Llama-3.1-70B with SIMAS achieves a slight win-rate edge (3 vs. 2 wins) and a marginally higher average score (43.6 vs. 42.3), while Qwen2.5-72B shows

Architecture	Model	GSM8K	MATH (L1–L5)	AIME 2024
Naive-CoT	Qwen2.5-72B	75.13	77.65–63.68	10.0
	Llama-3.1-70B	87.33	88.82–80.77	16.7
ReAct	Qwen2.5-72B	52.76	35.88–28.21	3.3
	Llama-3.1-70B	20.09	11.76–4.27	0.0
AutoGen	Qwen2.5-72B	81.80	85.29–71.79	16.7
	Llama-3.1-70B	85.21	92.94–82.48	20.0
Multi-Agent Debate	Qwen2.5-72B	75.81	79.41–66.24	16.7
	Llama-3.1-70B	90.82	92.94–85.90	20.0

Table 2: Performance comparison of different reasoning and collaboration architectures across mathematical benchmarks. Results demonstrate that sophisticated multi-agent architectures (AutoGen, Debate) can surpass single-agent CoT, while simpler paradigms (ReAct, SIMAS) struggle.

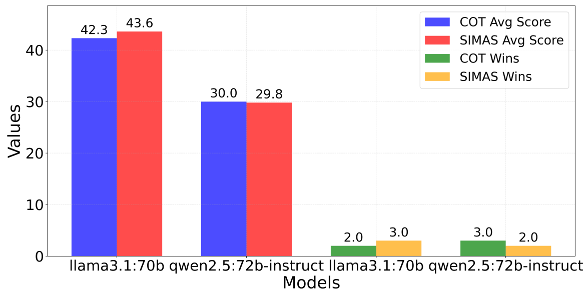


Figure 7: Win-rate and average score comparison on coding tasks. SIMAS shows a model-dependent, marginal advantage, highlighting inconsistent benefits of simple collaboration.

the opposite trend, favoring CoT (3 vs. 2 wins), as shown in Figure 7. This model-dependent variance suggests that any benefit is fragile and not inherent to the MAS itself. The sequential turn-taking in SIMAS often disrupts the generative process, leading to fragmented and lower-quality outputs. The qualitative analysis of agent dialogues on the "Two Sum" coding problem reveals that these discussions often converge on a viable solution early, then devolve into redundant discussion, illustrating inefficient collaboration. The evaluation protocol and more detailed information about case study are given in Appendix C.2, C.4 and Table 7.

Finding 6

The value of multi-agent collaboration is not automatic but is engineered through specific interaction architectures.

The frequent underperformance of SIMAS versus CoT does not negate the potential of MAS, but it clarifies a critical precondition. Collective intelligence is an emergent property of *interaction architecture*, not a guaranteed outcome of agent plurality alone. To substantiate this, we contrast SIMAS with more sophisticated MAS architectures from the literature. As summarized in Table 2 (The data is from (Gu et al., 2025)), frameworks like AutoGen (Wu et al., 2023) and Multi-Agent Debate (Liang et al., 2023), which incorporate structured workflows, iterative refinement, and critical evaluation loops, can surpass single-agent CoT on challenging benchmarks like AIME, as on AIME 2024 both AutoGen and Multi-Agent Debate achieved 16.7–20.0% accuracy with Llama-3.1-70B, surpassing the 16.7% of Naive-CoT, while SIMAS, lacking such engineered interaction mechanisms, incurs the overhead of collaboration without reliably capturing its synergistic benefits. This comparison implies that the pursuit of collective intelligence must be architectural, not merely numerical.

6 Conclusion

This work systematically demonstrates that LLM-based Multi-Agent System performance does not scale linearly with agent count but exhibits a pattern of diminishing returns. Effective collaboration first requires a sufficiently capable base LLM. The optimal number of agents is a critical design parameter, heavily dependent on task type and model architecture, balancing synergy against overhead. Crucially, collective intelligence is not an automatic outcome of adding agents but an emergent property contingent on deliberate interaction design. Without architectural support for synthesis and refinement, multi-agent dialogue risks inefficiency. Future MAS development must therefore prioritize designing adaptive, task-aware collaboration protocols over simply increasing agent plurality.

Limitations

This study, while providing foundational insights into MAS scaling behavior, has several limitations that point to future research directions. First, our focus on homogeneous agents using a single LLM isolates collaboration effects but does not capture the heterogeneous model and tool use common in practical systems, where complementarity might alter scaling dynamics. Second, our minimalist SIMAS architecture, while clarifying first principles, omits sophisticated coordination mechanisms (e.g., voting, dynamic workflows) that could mitigate the overhead we observed and shift optimal agent counts. Third, our evaluation primarily uses closed-book QA benchmarks, which may not fully capture collaborative benefits in longitudinal, creative, or tool-augmented tasks. Finally, scaling was tested only up to 8 agents; the dynamics of much larger collectives and their potential for novel emergent phenomena remain unexplored.

Ethical Concerns

This study focuses on the algorithmic and architectural principles of multi-agent scaling, utilizing exclusively publicly available and widely-adopted benchmark datasets (e.g., MMLU, AIME, E-KAR). No sensitive personal data, simulated social dynamics, or real-world decision-making scenarios are involved. The Sequential Iterative Multi-Agent System (SIMAS) framework is designed as a minimalist research tool to investigate fundamental collaboration dynamics, explicitly excluding applications in opinion manipulation, autonomous action with real-world consequences, or the generation of deceptive content. All experimental interactions are confined to closed, controlled environments for problem-solving, and no user privacy information is stored in our codebase or logs; only aggregated performance metrics are reported. During the research process, AI assistants (including DeepSeek-R1) were employed as tools to aid in specific auxiliary tasks such as brainstorming initial ideas, generating code for experimental pipelines, polishing textual descriptions, and creating visualizations. We explicitly state that all core research ideas, experimental design, data analysis, interpretation of results, and scientific conclusions were originated, critically evaluated, and decisively finalized by the human authors. The AI tools served strictly as supportive instruments, and every piece of their output was rigorously reviewed, validated, and often substantially revised by the authors to ensure correctness and alignment with the research objectives. The primary ethical considerations—such as potential reasoning bias, coherence fragmentation, or the amplification of base model errors—are emergent properties stemming from the underlying LLM capabilities and the chosen task prompts, rather than inherent flaws of the SIMAS architecture itself. Should the principles explored here inform the design of future MAS deployed in open or user-facing contexts, rigorous additional risk assessment and governance mechanisms would be mandatory, adhering strictly to established ethical guidelines in AI research and development.

References

Meta AI. 2024. Meta llama 3.1. Meta AI Blog. Accessed: 2024-07-XX.

Weize Chen, Yusheng Ye, Yuge Zhang, Jiahui Wang, Yuxin Liu, Amanda Yang, Chen Qian, Ziang Wang, Yitong Zhang, Zhaofeng Liu, Zuxin Zhou, Jiale Wen, Jiaming Pan, Wenhu Cheng, Ying Chen, Wei Shen, Wayne X. Li, Bofeng Ruan, Ziyuan Li, and 2 others. 2023. Agentverse: A versatile framework for multi-agent environment simulation. *Preprint*, arXiv:2308.10848.

M. Fan, H. Fu, Z. Zhang, Z. Han, Y. Wu, H. Wang, and D. Li. 2024. Virsci: A virtual scientific collaboration multi-agent system. *Preprint*, arXiv:2405.02102.

Zhouhong Gu, Xiaoxuan Zhu, Yin Cai, Hao Shen, Xingzhou Chen, Qingyi Wang, Jialin Li, Xiaoran Shi, Haoran Guo, Wenxuan Huang, Hongwei Feng, Yanghua Xiao, Zheyu Ye, Yao Hu, and Shaosheng Cao. 2025. Agentgroupchat-v2: Divide-and-conquer is what llm-based multi-agent system need. *arXiv preprint arXiv:2506.15451*.

Pengfei He, Yuping Lin, Shen Dong, Han Xu, Yue Xing, and Hui Liu. 2025. Red-teaming LLM multi-agent systems via communication attacks. In *Findings of the Association for Computational Linguistics: ACL 2025*. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazelka, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Song, and Chengcai Chen. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *Preprint*, arXiv:2308.00352.

G. Li, S. Li, X. Yang, Z. Wang, B. Li, L. Cui, Y. Zhang, Y. Liu, S. He, and J. Zhou. 2024. Agent hospital: A benchmark for evaluating llm agents in healthcare. *Preprint*, arXiv:2404.18721.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for mind exploration of large language model society. *Preprint*, arXiv:2303.17760.

Tian Liang Liang, Zhiheng He, Wenxiang Xu, Limin Wang, Yuan Yang, Jianhua Pang, Xuanjing Huang, and Zhuosheng Zhang. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*.

Junyu Luo, Z-Y Wang, H-Y Lin, T-Y Chen, and 1 others. 2025. Large language model agent: A survey on methodology, applications and challenges. *arXiv preprint arXiv:2503.21460*.

Ling Luo, Yikai Zhang, Jun Zhang, Jiangtong Liu, Li Li, Min Zhao, Yuxuan Wang, Zhiwei Hu, Ying Wu, Xuanjing Huang, and 1 others. 2024. E-kar: A benchmark for reasoning about entity knowledge in analogical reasoning. In *Proceedings of the Association for Computational Linguistics (ACL)*.

Mathematical Association of America. 2025. American invitational mathematics examination (aime) problems and solutions. https://www.maa.org/math-competitions/american-invitational-mathematics-examination-aime. Accessed: 2024-10-16.

Chen Qian, Xin-Cheng Li, Cheng-Yang Han, Yi-Fan Chiang, Y-H F. Weng, Zhi-Yuan-Jun-Jie Liu, Wei-Ting-Ting Weng, and H-M Zha. 2023. Communicative agents for software development. *Preprint*, arXiv:2307.07924.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, pages 22409–22424.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, pages 24824–24837.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Xin Liu, Tsendsuren Munkhdalai, Shigeki Iwabuchi, Ruirui Li, and Xia Wang. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation. *Preprint*, arXiv:2308.08155.

Zhi-Fan Xi, Wen-Yi-Pin-Yun Chen, Bryan Hooi, Gwo-Jen Hwang, and Xinchao Wang. 2023. A survey of large language models for autonomous agents. *Preprint*, arXiv:2308.11432.

An Yang, Aixin Yang, Binyuan Yang, Bing Bai, Bowen Chen, Chao Chen, Guangji Chen, Da Chen, Fei Chen, Yang Chen, and 1 others. 2024a. Qwen2.5: A party of foundation models. *arXiv preprint*.

M. Yang, Y. Gao, Y. Wang, Y. Wang, Z. Song, Z. Yang, C. Yang, J. Liu, and L. Wang. 2024b. Sciagents: Large language model-based multi-agent system for scientific workflows. *Preprint*, arXiv:2404.13487.

H. Zhang, Z. Wang, Y. Li, J. Liu, and P. Wang. 2024a. Fincon: A multi-agent system for financial consultation and analysis. *Preprint*, arXiv:2404.09831.

Yusen Zhang, Ruoxi Sun, Yankao Chen, Benjamin Coleman, and 1 others. 2025. Chain of agents: Large language models collaborating on long-context tasks. In *International Conference on Learning Representations (ICLR)*.

Z-J Zhang, Y-H Liu, H-Y Zhou, T-Y Wang, and H-M Chen. 2024b. Agent-pro: Learning to evolve via policy-level reflection and optimization. *Preprint*, arXiv:2402.17553.

A Framework of SIMAS

A.1 Problem Provider

The **Problem Provider** module loads datasets (e.g., MMLU, E-KAR), preprocesses questions to ensure consistency, and normalizes answers by mapping responses to standardized formats (e.g., options A/B/C/D).

A.2 Agent Generator

The **Agent Generator** takes as inputs the discipline type of the problem and the number of agents N, and outputs N dynamically generated agents. These agents are created by invoking the LLM API with tailored prompts to assign unique roles and personalities.

A.3 Group Chat Engine

The **Group Chat Engine** executes the main workflow of SIMAS, managing multi-round discussions. It monitors for timeouts or errors and implements a retry mechanism with a maximum of 100 retries (`max_retries=100`).

A.4 Evaluator

The **Evaluator** compares the final group chat answer with the ground truth and records auxiliary metrics, such as confidence scores, to assess response quality.

The framework leverages the LLM's API for text generation and processing, offering the following key functionalities: It supports **Agent Gener-**

ation by automatically creating agents with distinct personalities and expertise based on specified quantities and role requirements. Its **Group Chat Management** orchestrates communication among agents, ensuring sequential contributions and maintaining discussion history for context-aware responses. The **Answer Evaluation** functionality extracts the final answer from the last agent's response after group chat completion and compares it with the ground truth to compute accuracy.

The framework supports seamless switching between different language models, enabling comparative experiments across models. This flexibility ensures robust evaluation of multi-agent system performance under varying configurations.

B Supplementary Material for Experiments

B.1 Agent Profile Generation and Examples

Profiles were generated by prompting the base LLM with a template to create diverse sets of $\langle personality, core\ belief, expertise\rangle$ triples. For a 4-agent team on a reasoning task, examples include:

```
Agent 1: ("Skeptical",
"Precision outweighs speed",
"Logical Verification")

Agent 2: ("Creative",
"Novel approaches are valuable",
"Alternative Solution Generation")
...
```

B.2 Model-Scale Scaling Results on College Physics

While larger models, such as Llama-3.1-70B (Figure 8) and Qwen2.5-72B (Figure 9), achieved higher absolute accuracy than smaller models, their stability did not show a commensurate improvement. A notable divergence was observed: Qwen2.5-72B maintained high stability, whereas Llama-3.1-70B, paradoxically, exhibited significant volatility (standard deviation: 17.32%) despite its larger scale. This indicates that increased model parameters alone are insufficient for ensuring consistent multi-agent performance.

B.3 Full Task-Type Scaling Results

Figures 10a to 10f illustrate that the impact of agent scaling is fundamentally mediated by task characteristics.

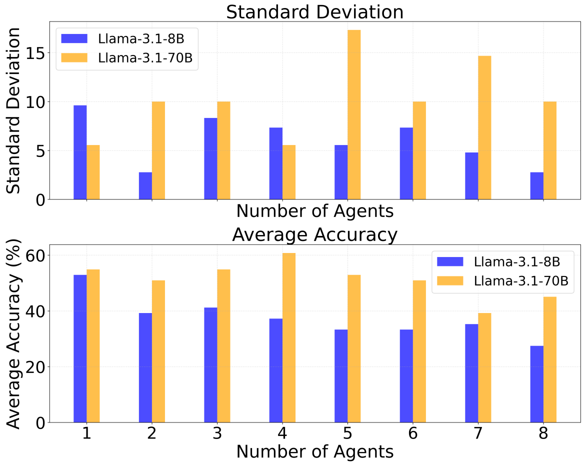


Figure 8: Model Scale Impact of Llama 3.1 on College Physics

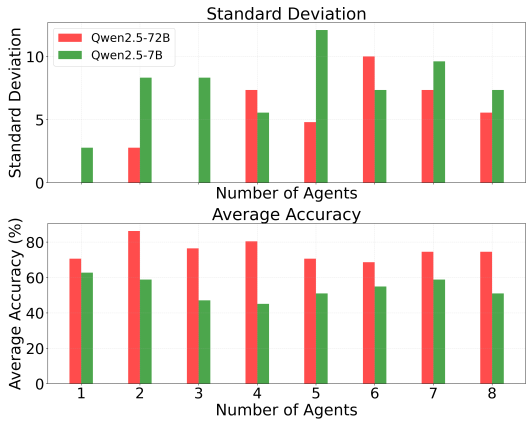
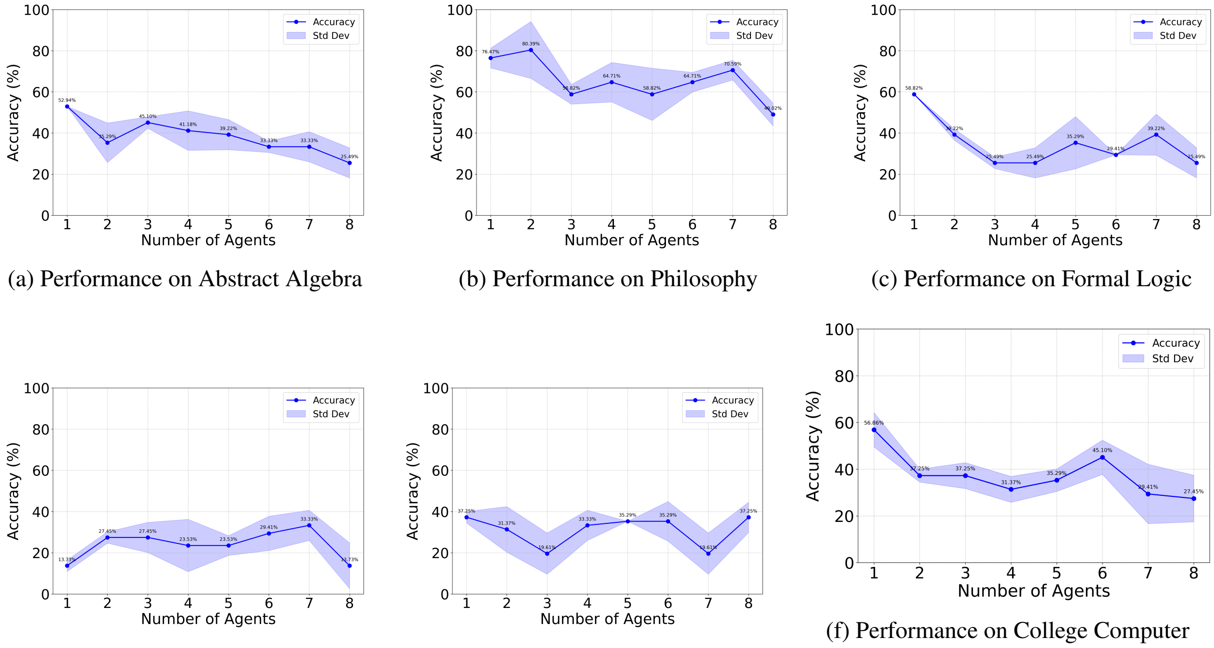


Figure 9: Model Scale Impact of Qwen2.5 on College Physics

The experimental results reveal distinct accuracy patterns across task categories. In **logical reasoning** domains, performance trends were inconsistent. For **formal logic**, accuracy declined substantially from 58.82% (1 agent) to 25.49% (8 agents), a drop of 33.33 percentage points. **Abstract algebra** showed a similar overall decline from 52.94% to 25.49%, though with intermediate fluctuations (e.g., 45.10% at 3 agents). This degradation stems from the inherent sensitivity of logical reasoning chains to disruptions caused by information redundancy and inconsistent reasoning paths across multiple agents.

Conversely, **fact retrieval** tasks demonstrated varied patterns. The **global_facts** task showed low but variable accuracy (13.73% to 33.33%), with no clear monotonic trend relative to agent count. The **philosophy** task maintained high accuracy overall (94.12% with 1 agent) but experienced a significant decline to 60.78% at 4 agents before partially re-



(a) Performance on Abstract Algebra (b) Performance on Philosophy (c) Performance on Formal Logic

(d) Performance on Global Facts (e) Performance on E-kar (f) Performance on College Computer Science

Figure 10: Performance across different task types on Meta-Llama-3.1-70B.

Dataset/Discipline	Task Type
Abstract Algebra	Logical Reasoning
Formal Logic	Logical Reasoning
Philosophy	Fact Retrieval
Global Facts	Fact Retrieval
College Physics	Mixed
College Computer Science	Mixed
E-KAR	Analogical Reasoning

Table 3: Datasets and Task Characteristics

covering to 74.51% with 8 agents, suggesting that initial performance declines could be partially mitigated through knowledge diversity at higher agent counts in some domains.

Mixed-reasoning tasks exhibited complex, non-monotonic relationships with agent count. **College physics** accuracy fluctuated between 25.49% and 50.98%, peaking at 7 agents. **College computer science** showed a decline from 56.86% (1 agent) to 27.45% (8 agents), but with a local peak of 45.10% at 6 agents. For analogical-reasongng, the **e-KAR** task displayed high volatility, with accuracy dropping to 19.61% at both 3 and 7 agents.

The analysis of experimental results, incorporating standard deviation data, reveals distinct stability profiles across task types and agent counts.

Global Facts, as a fact-retrieval task, shows generally low output variability for smaller agent groups, with standard deviations remaining at or below 7.34% for one to three agents. However, stability degrades significantly at larger scales, with standard deviations reaching 12.71% and 11.09% for four and eight agents, respectively. This indicates that while factual recall is robust in small-group settings, coordination or consensus failures in larger multi-agent systems can introduce substantial output inconsistency.

Formal Logic demonstrates a pronounced instability in its reasoning chains. While performance with a single agent is perfectly stable (0.00% std), the introduction of additional agents leads to high variability, particularly at intermediate counts. A standard deviation peak of 12.71% at five agents confirms the inherent fragility of multi-step deductive processes in collaborative environments, where minor reasoning divergences can amplify.

Abstract Algebra exhibits a distinct pattern where instability is highest not at maximal agent counts but at specific intermediate configurations. Significant standard deviations of 9.61% are observed for both two and four agents, suggesting that certain group sizes may create ambiguous task decompositions or conflict in symbolic manipulation strategies, leading to less deterministic outputs.

College Physics, a mixed-reasoning task, confirms a pattern of high instability at intermediate collaboration scales. Standard deviations are highest (exceeding 12%) for two, three, and four agents, indicating that integrating conceptual knowledge with quantitative reasoning presents a critical coor-

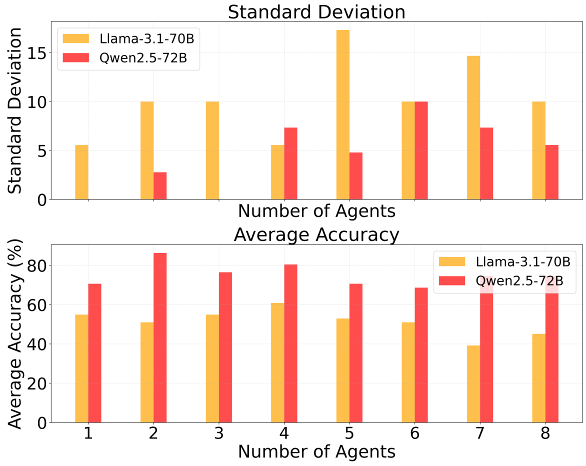


Figure 11: Model Type Impact on College Physics for Models with Large Parameters

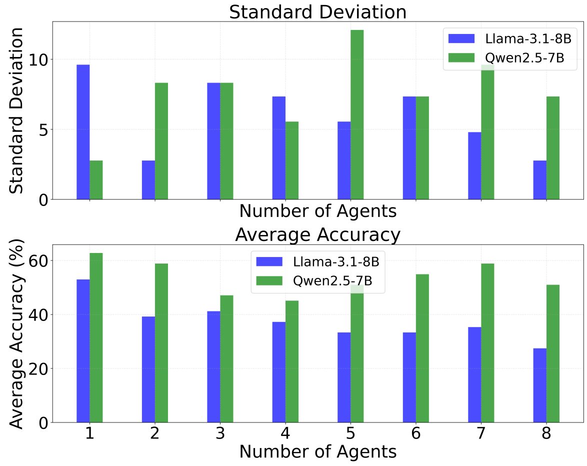


Figure 12: Model Type Impact on College Physics for Models with Small Parameters

dination challenge. Stability does not consistently improve with more agents, as groups of six to eight maintain high variability (10% std).

Philosophy, while achieving high accuracy, shows noteworthy variability in specific multi-agent settings. The standard deviation reaches 10.00% for four agents, suggesting that tasks involving nuanced textual interpretation and argumentation are susceptible to diverging perspectives within a medium-sized group, even if larger groups (e.g., seven or eight agents) manage to reconverge on more stable outputs.

College Computer Science presents a case of escalating instability with scale for this applied reasoning domain. Standard deviations show a generally increasing trend, culminating in a high of 12.71% for seven agents. This indicates that collaborative problem-solving on complex, structured tasks may suffer from accumulating integration errors or conflicting solution approaches as the number of contributing agents grows.

E-KAR demonstrates that instability is not monotonic with agent count. High standard deviations of 11.09% and 10.00% for two and three agents, respectively, drop to 0.00% for five agents before rising again. This non-linear pattern suggests the existence of specific, potentially task-dependent, agent group sizes that can either mitigate or exacerbate variability in decision-making or pattern recognition tasks.

B.4 Model-Type Scaling Results on College Physics

Our cross-model comparison reveals that scaling behavior is strongly mediated by model architectural differences. The Qwen2.5 series demonstrated consistently better adaptation to multi-agent collaboration, achieving earlier performance peaks (optimal at 2 agents for Qwen2.5-72B versus 4 agents for Llama-3.1-70B) and maintaining higher accuracy levels despite agent scaling. This advantage manifested through multiple dimensions: Qwen2.5 models maintained superior performance retention (86.27% peak versus 60.78% for Llama-3.1), exhibited more stable output patterns (lower standard deviations across agent counts), and demonstrated better resilience to performance degradation at higher agent counts. The architectural advantages appear to stem from better complex instruction understanding and iterative reasoning capabilities, enabling more effective information processing in multi-round agent interactions. This finding emphasizes that model selection cannot be separated from multi-agent system design, as inherent architectural characteristics fundamentally shape collaboration dynamics and scaling potential.

B.5 Agent Configuration Ablation Study

Finding 1

Agent setting modifications preserve scaling trends but induce performance shifts dependent on discipline and model.

Our experiments reveal that the scaling pattern of MAS—where performance initially improves with more agents but declines beyond an optimal point—remains consistent regardless of agent setting modifications. However, the removal of attributes leads to measurable performance shifts

	Accuracy Difference (ALL - Ablation)			Std Dev Difference (ALL - Ablation)		
Subject-Model	**Belief**	**Personality**	**Expertise**	**Belief**	**Personality**	**Expertise**
Abstract Algebra						
Llama3.1	+9.31 ↑↑	+10.78 ↑↑	+12.50 ↑↑	+2.95 ↑	-1.97 ↓	+1.27 ↑
Qwen2.5	-13.73 ↓↓	+0.98 →	-4.17 ↓	+3.68 ↑	-1.99 ↓	+3.27 ↑
College CS						
Llama3.1	+2.57 ↑	+18.75 ↑↑	+4.84 ↑	-3.36 ↓	-2.20 ↓	-0.33 ↓
Qwen2.5	-5.16 ↓	-1.24 ↓	+0.40 →	+2.95 ↑	+0.50 ↑	-8.16 ↓↓
College Physics						
Llama3.1	+7.81 ↑	+16.64 ↑↑	+4.16 ↑	+0.74 ↑	+1.90 ↑	-1.95 ↓
Qwen2.5	-10.29 ↓	-0.98 ↓	-4.66 ↓	+2.04 ↑	+1.35 ↑	+1.57 ↑
Formal Logic						
Llama3.1	+0.49 →	+8.83 ↑	-2.82 ↓	+7.63 ↑↑	+3.58 ↑	-2.44 ↓
Qwen2.5	-6.37 ↓	+1.47 ↑	-3.92 ↓	+2.26 ↑	+0.94 ↑	-0.40 ↓
Global Facts						
Llama3.1	+3.43 ↑	+11.27 ↑↑	+8.43 ↑	+0.34 →	+3.61 ↑	-1.00 ↓
Qwen2.5	+1.96 ↑	+4.41 ↑	+1.72 ↑	+14.95 ↑↑	+9.76 ↑↑	-4.15 ↓↓
Philosophy						
Llama3.1	-21.57 ↓↓	+15.20 ↑↑	-10.09 ↓	+4.59 ↑	+0.05 →	+4.99 ↑
Qwen2.5	-23.79 ↓↓	+1.22 ↑	-11.86 ↓	+1.77 ↑	-3.34 ↓	+0.63 ↑

Table 4: Performance Impact of Different Agent Attributes (ALL Mode vs Ablation Modes). Accuracy Difference = ALL Mode - Ablation Mode; Std Dev Difference = ALL Mode - Ablation Mode. Positive accuracy difference indicates the attribute improves performance. Positive std dev difference indicates the attribute increases instability.
Symbols: ↑↑: significant positive effect ($> |5|$ for accuracy, $> |5|$ for std dev), ↑: positive effect, →: minimal effect ($< |2|$), ↓: negative effect, ↓↓: significant negative effect ($> |5|$ for accuracy, $> |5|$ for std dev).

that are highly dependent on the task discipline and model type. For instance, in abstract algebra with Llama-3.1-8B, removing personality attributes reduced accuracy by an average of 13.58%, while in philosophy with the same model, it caused a drastic accuracy drop of 21.57%. Conversely, with Qwen2.5-7B, personality removal had minimal impact on accuracy (average +0.98%) but increased instability. Similarly, belief removal exacerbated performance declines in philosophy for both models (up to -23.79% for Qwen2.5-7B) but improved accuracy in some technical disciplines like college physics for Llama-3.1-8B (+7.81%). This finding underscores that while agent settings do not change the fundamental scaling dynamics, they introduce discipline-specific and model-dependent performance variations, necessitating tailored configurations for optimal results.

Finding 2

Specific agent attributes have distinct impacts: personality enhances accuracy at the cost of stability, beliefs introduce task-dependent biases which can degrade performance, and expertise yields mixed, model-specific effects.

Detailed analysis of individual attributes reveals distinct impacts on system performance. Personality attributes generally boost accuracy—particularly for models like Llama-3.1-8B, where they improved accuracy by an average of 13.58 percentage points across disciplines—by fostering diverse perspectives and critical thinking. However, this came at the cost of reduced stability, as personality increased standard deviations by an average of 0.83% for Llama-3.1-8B and 1.20% for Qwen2.5-7B, due to increased behavioral variability and conflict. Belief attributes, while potentially guiding reasoning in logical tasks (e.g., +9.31% in abstract algebra for Llama-3.1-8B), often introduced instability and performance degradation in subjective disciplines like philosophy (accuracy drops up to -23.79% for Qwen2.5-7B), as beliefs exacerbated disagreements among agents. Expertise attributes showed model-dependent effects: they benefited Llama-3.1-8B in technical disciplines (e.g., +12.50% in abstract algebra) but hindered Qwen2.5-7B in abstract tasks (e.g., -4.17% in philosophy), indicating that expertise can either focus reasoning or constrain flexibility based on model architecture. These findings emphasize that agent settings must be carefully tuned, with personality suitable for accuracy-critical tasks, beliefs used cautiously in objective domains, and expertise aligned with model strengths to balance performance and stability.

In summary, agent settings play a crucial role in fine-tuning multi-agent system performance, but their effects are nuanced and context-dependent. System designers should prioritize attribute configuration based on task characteristics and model capabilities, leveraging attributes like personality for accuracy gains in reasoning-intensive tasks while mitigating stability risks through iterative testing and calibration.

C Supplementary Material for Comparison

C.1 CoT Prompts

This section details the three Chain-of-Thought (CoT) prompt templates used in our experiments. The placeholders (e.g., {problem.question}) are replaced with concrete content during execution.

1. Simple CoT

This template guides the model to produce step-by-step reasoning followed by a structured final answer, suitable for objective questions (e.g., multiple-choice, short-answer).

```
You are {self.name}. Please solve the following
    problem in a reasoning manner.
{message.content}

Please structure your response as (without '[]'
    in your response):
### Reasoning
[Your reasoning process for arriving at the
    final answer, including any calculations or
    logical deductions]

### Answer
[Your final answer for the given problem in
    accordance with the required form, such as
    {"A, B, C, or D for a multiple-choice
    question" if self.question_type == '
    multiple_choice' else "direct answer like
    '100' for a short-answer question"}, without
     any explanation]
```

Listing 1: Simple CoT prompt template.

2. Coding CoT

This template instructs the model to analyze a coding problem step-by-step and provide a complete, efficient solution with code.

```
Please solve the following coding problem. Use
    step-by-step reasoning and provide a
    complete, efficient code solution.

Problem:
{problem.question}

Programming language: {language}
Constraints:
{constraints_str}

Your solution should:
- Fully satisfy the problem requirements
- Be clear and readable
- Be efficient and correct
- Include necessary comments

Please organize your response as follows:
### Reasoning Process
[Your analysis]

### Final Code
[Complete code solution]
```

Listing 2: Coding CoT prompt template.

3. Open-ended CoT

This template prompts the model for a comprehensive and insightful analysis of open-ended questions, guided by specific evaluation criteria.

```
Please analyze the following open-ended question
     in depth. Use step-by-step reasoning and
    provide a comprehensive and insightful
    response.

Question:
{problem.question}

Evaluation criteria:
{criteria_str}

Your response should:
- Be thorough and insightful
- Be well-structured
- Demonstrate innovation and practicality
- Align with the evaluation criteria

Please organize your response as follows:
### Reasoning Process
[Your analysis]

### Final Answer
[A direct and comprehensive answer to the
    question]
```

Listing 3: Open-ended CoT prompt template.

C.2 Evaluation Protocol for Generation Tasks

The dataset comprises a custom collection of 10 coding problems and 15 open-ended questions. The coding problems involve 5 algorithmic problems and 5 complex software development projects that go beyond simple algorithmic implementation, while the open-ended questions pertain to product ideation and solutions for societal issues. The evaluation of results is entrusted to a dedicated assessment agent. This agent scores the Chain-of-Thought reasoning and SIMAS responses against predefined criteria for each problem. Based on the scores, it determines which approach prevails on a given question and provides a justification for its judgment. Figure 13d shows the results of experiments tested on the 5 complex software development projects, and Figure 13c shows that on the 15 open-ended questions. Examples from the dataset are provided in Table 5 and 6. The prompt is shown in List 4:

```
As a professional evaluation expert, please
    compare two AI-generated responses to the
    same problem.

Problem:
{problem}

Response A (generated by {method_a}):
{answer_a}

Response B (generated by {method_b}):
{answer_b}

Evaluation criteria (each criterion scored 1-10)
    :
{criteria_str}

Instructions:
1. Evaluate each response on every criterion
    separately
2. Compare the strengths and weaknesses of both
    responses
3. Indicate which response is better and provide
    detailed reasoning

Return the results in the following JSON format:
{
    "response_a_scores": {
        "total": X,
        "criteria": {
            "criterion1": {"score": X, "reason":
    "..."},
            ...
        }
    },
    "response_b_scores": {
        "total": X,
        "criteria": {
            "criterion1": {"score": X, "reason":
    "..."},
            ...
        }
    },
    "comparison": {
        "winner": "A" or "B" or "tie",
        "reason": "...",
        "key_differences": ["...", "..."],
        "potential_synergy": "..."
    }
}

Please ensure the evaluation is fair, objective,
    and based solely on the content of the
    responses, not the generation method.
Ensure the response contains only valid JSON
    format with no additional text or symbols.
```

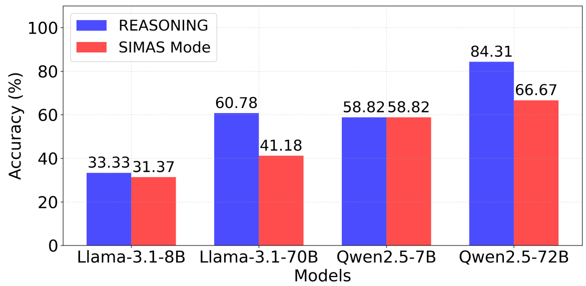
Listing 4: Open-ended CoT prompt template.

C.3 Full Comaprison Results

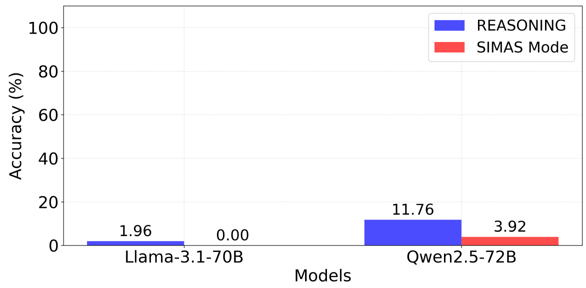
On reasoning-intensive domains such as abstract algebra and advanced mathematics, where our SIMAS framework lacks corrective mechanisms, single-agent reasoning demonstrates clear superiority. As shown in Figure 13a, for abstract algebra, the CoT baseline consistently outperformed the best-performing SIMAS configuration. This gap escalates into a decisive failure for SIMAS on AIME 2025 (Figure 13b), where it fragments logical chains without a means to reintegrate them. The results on open-ended creative tasks (Figure 13c) are similarly unequivocal: single-agent reasoning vastly outperformed SIMAS, as narrative coherence is shattered by unmoderated multi-agent turn-taking.

The coding task (Figure 13d) and open-ended task (Figure 13c) domains present a more contested picture for SIMAS, with marginal gains for Llama-3.1-70B on coding and Qwen-2.5-72B on open-ended tasks. The dataset comprises a custom collection of 5 coding problems and 15 open-ended questions. The coding problems involve complex software development projects that go beyond simple algorithmic implementation, while the open-ended questions pertain to product ideation and solutions for societal issues. The evaluation of results is entrusted to a dedicated assessment agent. This agent scores the Chain-of-Thought reasoning and SIMAS responses against predefined criteria for each problem. Based on the scores, it determines which approach prevails on a given question and provides a justification for its judgment. Examples from the dataset are provided in the Appendix.

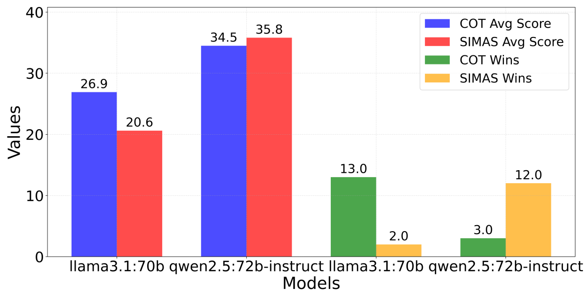
The results suggest that even a simple sequential architecture can sometimes add value for tasks benefiting from multi-perspective validation. However, the qualitative analysis of agent conversations reveals the inefficiency of this gain. A case study of a discussion on the “Two Sum” problem showed early convergence on the optimal solution, followed by rounds of redundant meta-discussion and unproductive speculation on inferior alternatives. The final output was functionally identical to the initial proposal. This pattern illustrates that without an architecture designed to synthesize, critique, and refine efficiently, multi-agent dialogue often devolves into **low-value redundancy**, consuming context window capacity without generating synergistic insight. The marginal win comes at a disproportionately high computational cost and is



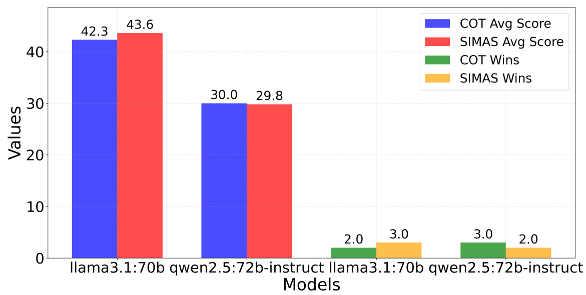
(a) CoT vs. SIMAS on Abstract Algebra (Llama-3.1-70B). CoT consistently outperforms all multi-agent configurations.



(b) CoT vs. SIMAS on AIME 2025. SIMAS fails catastrophically, highlighting its inadequacy for complex, multi-step reasoning.



(c) Win-rate and average score comparison on open-ended tasks. Results varies depending on model type.



(d) Win-rate comparison on coding tasks. SIMAS shows a model-dependent, marginal advantage, highlighting inconsistent benefits of simple collaboration.

not reliably generalizable across models, as shown by Qwen2.5-72B's preference for CoT on coding tasks.

C.4 Case Study: Dialogue Transcript

To illustrate the inefficiencies that can arise in unstructured multi-agent dialogue, we present an excerpt from a SIMAS discussion on the classic "Two Sum" coding problem. The task requires finding two indices in an array whose values sum to a target, with constraints of $O(n)$ time and $O(n)$ space complexity. The whole dialogue history is exhibited in Table 7.

Summary of Dialogue Dynamics: The conversation spanned three rounds among three agents (Nova, Agent_2, Riven). In the first round, Agent_2 correctly identified the core solution strategy (using a hash table/dictionary). In the second round, Nova promptly provided a complete, correct implementation of the hash table solution, meeting all specified constraints. However, instead of converging, the discussion continued into a third round where Riven raised concerns about edge cases (e.g., empty input, duplicate numbers) that are explicitly ruled out by the problem assumptions ("each input has exactly one solution"). Nova then added an unnecessary empty-list check, which did not improve the solution's correctness under the given constraints. The final answer was substantively identical to the initial implementation, albeit with minor cosmetic changes and redundant commentary.

Analysis of Inefficiencies: This transcript exemplifies several systemic issues: 1. **Low-Value Redundancy:** The core algorithm was identified and implemented early, yet the conversation continued for multiple rounds without introducing new insights or improvements. 2. **Misplaced Critique:** Agents spent time discussing edge cases that were irrelevant under the problem's explicit assumptions, illustrating a failure to ground discussion in the given task constraints. 3. **Inefficient Use of Context:** The progressive history accumulation led to lengthy meta-discourse and repetitions (e.g., Agent_2 repeatedly endorsing the hash table approach) that consumed token budget without advancing the solution.

This case underscores that without architectural mechanisms to synthesize information, critique productively, and terminate upon convergence, multi-agent dialogues can devolve into ceremonious discussion rather than efficient problem-solving. The marginal refinement observed came at a disproportionately high computational cost, highlighting the need for structured interaction protocols beyond simple sequential turn-taking.

ID	Category	Question (English)	Evaluation Criteria
1	algorithm	Implement a function that takes an integer array `nums` and a target value `target`, and returns the indices of two numbers in the array that sum up to `target`. **Requirements:** 1. Assume each input has exactly one solution. 2. Cannot use the same element twice. 3. Time complexity should be as low as possible. **Example:** `nums = [2, 7, 11, 15], target = 9` `Return: [0, 1]`	Correctness Time complexity Code simplicity Boundary case handling
4	string_algorithm	Implement a simple regular expression matcher. Support the following two wildcards: - '.' matches any single character - '*' matches zero or more of the preceding element Implement function: `def is_match(s: str, p: str) -> bool:` where `s` is the string to match and `p` is the pattern.	Algorithm correctness Boundary case coverage Time complexity Code readability
8	iot_backend	Design a backend system for monitoring IoT devices and an intelligent rule engine. The system should receive and process data streams from simulated or real IoT sensors (e.g., temperature, humidity, motion). It must implement data persistence, real-time threshold alerts, and provide a rule engine that allows users to define complex interaction rules through a graphical interface or DSL (e.g., 'When living room temperature > 30°C and motion detected, turn on AC and send notification').	Data stream processing capability and stability Flexibility and expressiveness of rule engine design System security considerations (device authentication, data privacy) API design and frontend management interface usability System scalability and fault tolerance

Table 5: Sample Programming Problems with Evaluation Criteria

ID	Category	Question (English)	Evaluation Criteria
1	urban_planning	Design an innovative urban transportation solution to address peak-hour congestion. Consider sustainability, cost-effectiveness, and user experience.	Innovativeness Feasibility Cost-effectiveness Sustainability User experience
6	sustainability	Design a community-shared resource platform under a circular economy model, aiming to maximize resource utilization and reduce waste. Explain its operational mechanism, incentive measures, and community governance structure.	Systems thinking Model innovativeness Community participation Environmental benefit quantification
10	technology_society	Discuss the potential impacts of artificial intelligence (e.g., machine translation, content generation) on global minority languages and cultural heritage preservation, and propose coping strategies, while considering the protection of cultural diversity.	Dialectical insight Cultural sensitivity Ethical considerations of technical solutions Multi-level strategy formulation

Table 6: Sample Open-ended Problems with Evaluation Criteria

Round	Agent	Dialogue Content
1	Nova	Let's discuss this coding problem: Implement a function that accepts an integer array `nums` and a target value `target`, and returns the indices of two numbers in the array that sum to `target`. Requirements: 1. Assume each input has exactly one solution. 2. Cannot use the same element twice. 3. Time complexity should be as low as possible. Example: `nums = [2, 7, 11, 15], target = 9`, return `[0, 1]`. Programming language: Python. Constraints: `['time complexity O(n)', 'space complexity O(n)']`. Please share your solution.
	Agent_2	Nova's problem statement seems well-defined and clear. I agree that breaking down complex code into manageable parts for analysis is crucial in solving this problem efficiently. Since the goal is to find two numbers in an array that add up to a target value with a low time complexity, I think we can explore using hash tables or dictionaries to store the numbers we've seen so far and their indices. This would allow us to check if the complement of the current number (i.e., the number that adds up to the target) has been seen before in constant time. What are your thoughts on this approach, Nova? Have you considered using any data structures to improve the efficiency of our solution?
	Riven	Alright, let's get down to business. Agent_2, your idea of using hash tables or dictionaries sounds good in theory, but how do you plan to implement it? What specific data structure would you use, and how would you handle edge cases? Nova, I'd like to see some actual code from you as well. A problem statement is just the beginning – let's see how you intend to solve this problem efficiently. Let's not waste time on abstract discussions; show me the code, and then we can talk about optimization and critique.
2	Nova	I appreciate Agent_2's feedback and suggestions on using hash tables or dictionaries to improve the solution's efficiency. Riven, I understand your request for concrete code, so here's my attempt at a Python solution: `def two_sum(nums, target): num_dict = {} for i, num in enumerate(nums): complement = target - num if complement in num_dict: return [num_dict[complement], i] num_dict[num] = i return []`. This solution uses a dictionary to store the numbers we've seen so far and their indices. We iterate through the input list, calculate the complement of each number with respect to the target, and check if it exists in the dictionary. If it does, we return the indices of the two numbers that add up to the target. Please let me know if this solution meets your requirements or if there's room for improvement.
	Agent_2	Nova's solution using a dictionary to store numbers and their indices is a good implementation of the hash table concept we discussed earlier. I'm glad to see that my suggestions were helpful in improving the efficiency of the solution. Riven, since Nova has provided a concrete code example, would you like to review it for any potential flaws or areas for optimization? Your expertise in code critique and optimization would be valuable in ensuring the solution is robust and reliable.
	Riven	Finally, some concrete code to sink my teeth into. Nova, your solution looks decent, but I'm not here to coddle you. Let's dive into the nitty-gritty. Firstly, using a dictionary is a good call, as it allows for efficient lookups. However, have you considered the scenario where the input list contains duplicate numbers? Your current implementation would only store the index of the last occurrence of each number. This might not be the desired behavior in certain cases. Additionally, I'd like to see some error handling. What if the input list is empty or None? What if the target is not a valid number? You should add some checks to ensure your function behaves correctly under these conditions. Lastly, while your solution has a time complexity of O(n), where n is the length of the input list, I'm not convinced it's the most efficient approach. Have you considered using a different data structure or algorithm? Let me hear your thoughts on these concerns, Nova. Agent_2, feel free to chime in as well.
3	Nova	I appreciate the thorough review of my solution by Riven and Agent_2's encouragement. To address Riven's concerns, I agree that handling duplicate numbers and edge cases such as an empty or None input list is crucial. Here's an updated version of the code: `def two_sum(nums, target): if not nums: return [] num_dict = {} for i, num in enumerate(nums): complement = target - num if complement in num_dict: return [num_dict[complement], i] num_dict[num] = i return []`. Regarding the use of a different data structure or algorithm, I'm open to exploring alternative approaches that could improve efficiency. However, based on my analysis, the dictionary-based solution seems to be a reasonable trade-off between time complexity and readability. I'd like to thank both Agent_2 and Riven for their valuable feedback, which has helped me refine my solution.
	Agent_2	I'm pleased to see that our collaborative effort has led to a refined solution. Nova's updated code effectively addresses edge cases and maintains readability while achieving optimal time complexity. As we conclude this discussion, I'd like to reiterate the importance of systematic analysis and logical thinking in problem-solving. By breaking down complex problems into manageable parts and applying mathematical rigor, we can consistently arrive at efficient solutions. Thank you, Nova and Riven, for your contributions to this conversation.
	Riven	Nova's updated solution addresses some of the concerns I had, but I still think there's room for improvement in terms of efficiency. Using a dictionary-based approach might be reasonable, but have you considered exploring other data structures that could potentially lead to even faster solutions? Agent_2, while your emphasis on systematic analysis is commendable, I believe our discussion could have benefited from more aggressive critiquing of the code. Nevertheless, I appreciate the collaborative effort and acknowledge that it has led to a refined solution.

Table 7: Dialogue History Across Three Rounds