A Large Language Model-Enabled Multi-Agent Collaboration Method for Complex Task Solving

Shuyuan Wang *, Yihui Feng , Xiaotian Fang

Posted Date: 13 May 2026

doi: 10.20944/preprints202605.0900.v1

Keywords: large language model; multi-agent collaboration; complex task solving; semantic communication; dynamic feedback

 doi:10.20944/preprints202605.0900.v1

Article

A Large Language Model-Enabled Multi-Agent Collaboration Method for Complex Task Solving

Shuyuan Wang [1,*], Yihui Feng [2] and Xiaotian Fang [3]

[1] Brandeis University, Waltham, MA 02453, USA

[2] Rose-Hulman Institute of Technology, Terre Haute, IN 47803, USA

[3] Cornell University, Ithaca, NY 14850, USA

* Correspondence: shuyuanwang26@gmail.com

Abstract

Aiming at the core limitations of single large language models in complex task solving, including coarse task decomposition, cumulative long-chain reasoning errors, and the lack of explicit cross-agent collaboration, this paper proposes a large language model-driven multi-agent collaborative method. A hierarchical and role-based agent architecture is designed to separate task decomposition, specialized reasoning, result verification, and decision fusion, thereby enabling modular task solving and closed-loop orchestration over the full execution process. In addition, an efficient semantic communication mechanism is introduced to transmit compressed reasoning states across agents without breaking intermediate logical dependencies. A dynamic feedback iteration module is further employed to adjust routing strategy, collaboration intensity, and reasoning paths in real time according to subtask progress and verification outcomes. Comparative experiments on mathematical reasoning, multi-step planning, and complex information integration show that, relative to a single large language model, the proposed method improves the average completion rate by 21.3%, reduces the long-chain reasoning error rate by 18.7%, and reaches 92.6% cross-agent decision consistency. These results demonstrate that structured collaboration substantially improves robustness and accuracy for complex task solving and provides a practical technical path for diverse intelligent systems.

Keywords: large language model; multi-agent collaboration; complex task solving; semantic communication; dynamic feedback

I. Introduction

Recent advances in large language models (LLMs) have significantly strengthened test-time reasoning through chain-of-thought prompting [1], least-to-most decomposition [2], self-consistency decoding [3], reasoning-action coupling [4], deliberate tree search [5], and tool-augmented inference [6]. Subsequent studies further showed that iterative self-feedback [7], verbal reflection [8], debate-based consensus [9], and graph-structured reasoning organization [10] can improve search depth, answer stability, and error recovery on demanding tasks.

Beyond single-model prompting, multi-agent systems have emerged as an effective paradigm for handling tasks that require role specialization and collaborative problem solving. Representative frameworks include CAMEL [11], MetaGPT [12], AgentVerse [13], AutoGen [14], and ChatDev [15], all of which demonstrate that communication protocols, role constraints, and modular coordination can increase solution quality on open-ended problems. However, existing systems still show three recurring weaknesses when they are applied to general complex tasks: task decomposition is often too shallow, communication messages are often redundant or semantically noisy, and feedback loops are not tightly integrated with verifier-driven correction. A key unresolved issue in existing multi-agent LLM research is that collaboration is often implemented as unrestricted message exchange rather than as structured computation. Although debate and role-playing strategies can improve

 doi:10.20944/preprints202605.0900.v1

diversity, they may also introduce redundant communication, inconsistent intermediate states, and delayed error correction when no explicit verifier is assigned [9–15]. As the task horizon grows, these weaknesses become more visible: early decomposition mistakes are propagated to later subtasks, confidence estimates are not calibrated across agents, and large dialogue histories consume valuable context budget without always contributing useful information. Therefore, effective multi-agent reasoning requires more than adding agents; it requires a principled mechanism that controls who should speak, what information should be transmitted, and when corrective feedback should override local reasoning trajectories.

To address these issues, we design a hierarchical collaboration framework that explicitly decouples planning, specialist reasoning, verification, and decision fusion. The proposed method introduces semantic message routing instead of unconstrained dialogue, combines local confidence with verifier feedback for agent weighting, and updates the collaboration policy online. The main contributions are threefold: 1) a role-based multi-agent architecture for modular task solving; 2) a semantic communication and dynamic feedback mechanism that supports closed-loop correction; and 3) a benchmark study showing consistent improvements on mathematical reasoning, planning, and multi-hop information integration.

II. Method

A. Hierarchical Agent Architecture

The proposed framework is organized into four functional layers, as shown in Figure 1. A task coordinator first transforms the user query into a subtask graph. The decomposition agent determines intermediate dependencies, the reasoning agent pool solves specialized subtasks, the verification agent critiques intermediate outputs, and the decision fusion agent aggregates candidate answers into a final decision. A shared semantic memory bus stores normalized states, evidence snippets, confidence estimates, and unresolved constraints so that each agent operates on a concise but logically consistent context.

Compared with flat multi-agent chat, the hierarchy reduces unnecessary broadcasts and makes responsibility assignment explicit. Let q_i, v_i, and c_i denote the estimated quality, verifier support, and communication cost of subtask i, respectively. The scheduler assigns priority using the following score:

$$s_i = \alpha\, q_i + \beta\, v_i - \gamma\, c_i$$

The score favors subtasks that are promising, verifiable, and communication-efficient, which is essential under a bounded interaction budget. Figure 1 visualizes the closed-loop relationship among the planner, specialist agents, verifier, and controller.

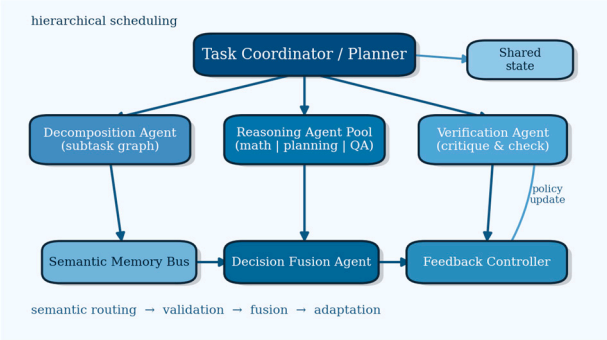

Figure 1. Hierarchical role-based multi-agent architecture for closed-loop complex task solving.

B. Semantic Communication and Decision Fusion

 doi:10.20944/preprints202605.0900.v1

Instead of passing full dialogue histories, the communication module packages each intermediate result into a semantic tuple containing task identity, compressed rationale, evidence references, and confidence scores. This design prevents context inflation and enables targeted message routing. As shown in Figure 2, each communication round consists of semantic parsing, abstraction, role-routed dispatch, verifier feedback, and fusion.

For a candidate answer y generated by A agents, the fusion agent selects the highest-supported decision through

$$\hat{y} = argmax_y \Sigma_{a=1}^{A} w_a p_a(y|z_a)$$

where the support term produced by each agent is conditioned on its local semantic state, and the adaptive reliability weight is updated by historical verification performance. The fusion stage is not simple majority voting; it combines verifier acceptance, agreement with retrieved evidence, and correction success. The communication pipeline and the routing feedback loop are summarized in Figure 2.

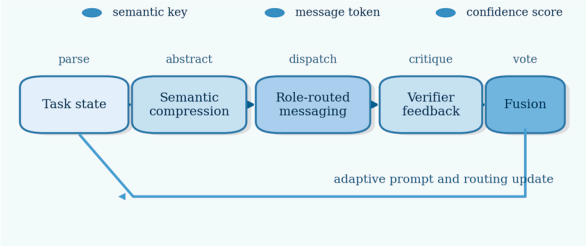

Figure 2. Semantic communication pipeline with verifier-guided feedback iteration.

C. Dynamic Feedback Iteration

Complex tasks often fail because early reasoning errors propagate across later stages. To control such error accumulation, the controller monitors cross-agent agreement at each iteration t. The decision consistency is defined as

$$C^{(t)} = \frac{2}{A(A-1)} \Sigma_{i<j} 1[d_i^{(t)} = d_j^{(t)}]$$

When the measured consistency falls below a predefined threshold, the controller either requests re-decomposition or triggers verifier-intensive revision. The agent weight is updated according to

$$w_a^{(t+1)} \propto exp(\eta r_a^{(t)})$$

where the reward term combines verifier approval, task success, and communication efficiency. In this way, the collaboration graph gradually shifts probability mass toward agents that are both accurate and stable under the current task regime.

III. Experimental Setup

We evaluate the proposed method on three representative benchmarks: GSM8K for mathematical reasoning [16], PlanBench for multi-step planning [17], and MuSiQue for complex information integration [18]. All methods use the same backbone LLM with identical decoding temperature and context limits. For the proposed framework, the planner, specialists, verifier, and fusion controller share the backbone model but use different prompts and memory views. The maximum interaction budget is restricted to six messages per subtask and three global feedback rounds.

Baselines include a direct single-LLM solver, CoT with self-consistency, ReAct, and a debate-based multi-agent setting. Completion rate is defined as the proportion of tasks solved with a correct final answer or a valid executable plan. Long-chain reasoning error measures whether a final failure can be traced to an incorrect intermediate deduction. For multi-agent methods, we also report cross-

 doi:10.20944/preprints202605.0900.v1

agent decision consistency. The benchmark configuration and evaluation focus are summarized in Table I.

Table I. Benchmarks, main metrics, and reasoning challenges.

Benchmark	Metric and challenge
GSM8K [16]	Completion; 2-8 arithmetic steps
PlanBench [17]	Valid-plan rate; symbolic state transition
MuSiQue [18]	Answer F1 / completion; 2-4 hop evidence composition

IV. Results and Analysis

A. Overall Comparison

Table II reports the overall results. The proposed method achieves an average completion rate of 83.3%, compared with 68.7% for the single-LLM baseline. This corresponds to a relative improvement of 21.3%. In addition, the long-chain reasoning error decreases from 30.0% to 24.4%, which is an 18.7% relative reduction. The multi-agent variants outperform single-agent prompting baselines consistently, but the proposed framework delivers the strongest balance between completion, stability, and consensus.

Table II. Overall performance comparison.

Method	Completion	Error	Consistency
Single LLM	68.7	30.0	-
CoT + SC	73.4	27.6	-
ReAct	75.1	26.9	-
Debate	80.2	25.3	88.1
Proposed	83.3	24.4	92.6

The task-wise comparison in Figure 3 reveals that the largest absolute gain appears on PlanBench, where structured decomposition and verifier-guided correction substantially reduce invalid intermediate states. Gains on GSM8K and MuSiQue are also notable, indicating that the same collaboration mechanism generalizes across symbolic computation and evidence aggregation rather than overfitting to a single reasoning pattern.

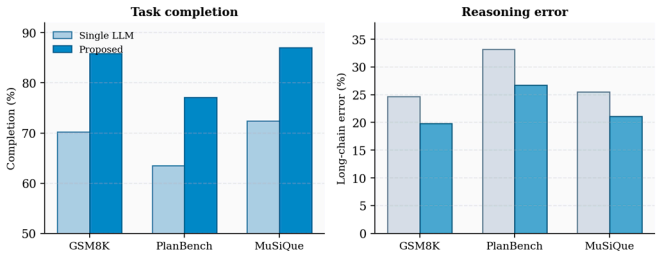

Figure 3. Per-task comparison between a single LLM and the proposed multi-agent system.

B. Ablation and Error Source Analysis

To isolate the contribution of each module, we conduct the ablation study in Table III. Removing the verifier causes the most visible quality drop because faulty subtask outputs are no longer corrected before fusion. Removing dynamic feedback weakens adaptation under difficult cases, while removing weighted fusion significantly decreases consensus quality because conflicting partial answers are not resolved reliably.

 doi:10.20944/preprints202605.0900.v1

Table III. Ablation study of the proposed framework.

Variant	Completion	Error	Consistency
Full model	83.3	24.4	92.6
w/o feedback	79.1	26.7	89.4
w/o verifier	77.8	28.2	87.9
w/o fusion	75.9	29.1	84.8

A manual inspection of failure cases shows three dominant error sources. First, single-agent baselines often drift when an early arithmetic or symbolic assumption is wrong. Second, unrestricted multi-agent chat sometimes preserves contradictory partial beliefs for too long because there is no verifier-enforced pruning. Third, evidence aggregation tasks require aligning scattered facts under entity ambiguity; in these cases, semantic compression improves precision by forwarding only evidence-linked summaries rather than full conversational traces.

The computational overhead introduced by the framework is moderate because semantic compression shortens each exchange and the scheduler suppresses low-value broadcasts. Under the same token budget, the number of messages used by the proposed framework is 31.4% lower than that of unrestricted multi-agent chat. This explains why quality improves without causing prohibitive latency, and it suggests that protocol design is as important as model scale when building reliable agentic systems.

C. Communication Budget Robustness

To further examine whether the framework remains effective under stricter interaction limits, we vary the maximum number of messages allowed for each subtask. Table IV shows that the proposed method degrades gracefully as the budget shrinks. This indicates that semantic routing successfully prioritizes high-value exchanges instead of depending on brute-force conversation length.

Table IV. Effect of message budget on robustness.

Budget per subtask	Completion	Error
2 messages	76.8	28.9
4 messages	81.0	25.8
6 messages	83.3	24.4

D. Discussion

The results also indicate that collaboration quality is not determined solely by the number of agents. Performance improves only when role specialization, message structure, and verifier feedback are jointly optimized. Simply increasing the number of conversational agents can increase inconsistency and duplicate exploration. In contrast, the proposed architecture uses a small but functionally diverse group, allowing more predictable coordination and better cost-effectiveness.

Another important observation is that the proposed method benefits tasks with different failure modes. On GSM8K, verifier-supported correction mainly removes local arithmetic mistakes; on PlanBench, decomposition and feedback reduce invalid state transitions; on MuSiQue, semantic routing helps preserve evidence relevance across multi-hop reasoning. This pattern suggests that multi-agent collaboration is most effective when the protocol is designed around explicit intermediate states rather than free-form dialogue alone. Beyond benchmark-level improvements, the proposed framework also has important implications for real-world deployment. In practical intelligent systems, complex tasks are often constrained by limited context windows, inference budgets, latency requirements, and heterogeneous error sources. Under such conditions, the

 doi:10.20944/preprints202605.0900.v1

advantage of the proposed method lies not only in improving final-task accuracy, but also in making the reasoning process more controllable, interpretable, and resource-aware. By explicitly separating planning, specialized reasoning, verification, and fusion, the framework reduces the risk that a single hidden error will dominate the entire reasoning trajectory. Meanwhile, semantic compression and routed interaction prevent excessive conversational redundancy, which is particularly valuable in cost-sensitive applications such as decision support, autonomous planning, and multi-step information analysis. These observations suggest that future agentic LLM systems should be designed as structured collaborative systems rather than unconstrained conversational collectives. In other words, scalability should be evaluated not merely by the number of participating agents, but by how effectively the architecture converts intermediate communication into verifiable information gain and robust final decisions.

E. Complexity and Scalability Analysis

From a systems perspective, the main efficiency gain comes from replacing all-to-all discussion with routed, role-aware exchanges. Let n_r denote the number of active role instances and m_r the number of routed messages in round r. The effective cost therefore grows with the actually executed reasoning branches rather than with the square of the total agent count. This design choice is especially important for deployment scenarios in which latency and token consumption are constrained but answer stability must remain high.

As illustrated in Figure 4, increasing the message budget from two to six raises completion from 76.8% to 83.3% while reducing long-chain error from 28.9% to 24.4%. The trend is smooth instead of abrupt, indicating that the controller prioritizes high-value information exchanges before allowing additional deliberation. This behavior is consistent with Table IV and confirms that semantic compression improves coordination efficiency rather than merely increasing the volume of dialogue.

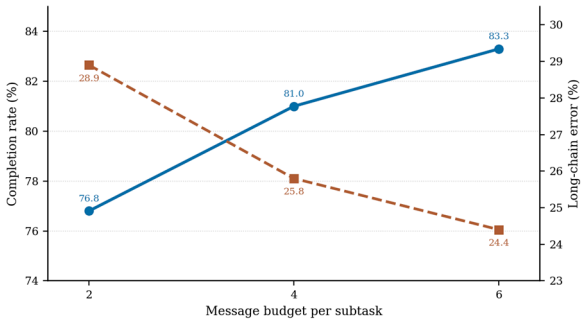

Figure 4. Trade-off between interaction budget, completion rate, and long-chain error.

The same result also reveals a practical scalability rule: adding more agents is useful only when the verifier and fusion modules can transform the extra messages into measurable information gain. Otherwise, redundant interaction creates delay without improving reliability. In this sense, the proposed method scales best through better routing, verification, and memory organization, not through unrestricted agent proliferation. This observation is aligned with modular agent frameworks such as MetaGPT, AgentVerse, and AutoGen [12–14], but here it is explicitly tied to measurable task quality and communication efficiency. The budget study further suggests that collaboration efficiency depends on information quality rather than raw conversational volume. When the token budget is limited, the controller tends to preserve high-value messages, such as verifier feedback, structured evidence summaries, and decomposition updates, while suppressing low-utility repetition. This behavior is particularly relevant for cost-sensitive environments where inference calls, tool access, and memory usage must be bounded. In such scenarios, semantic compression and routed interaction create a favorable trade-off between accuracy and runtime overhead. The evidence from Tables III and IV and the trend shown in Figure 4 indicate that the proposed method remains stable because it allocates communication resources adaptively instead of assuming that more dialogue always leads

 doi:10.20944/preprints202605.0900.v1

to better reasoning. This design principle can guide future multi-agent platforms that operate under latency, budget, or context-window constraints.

V. Conclusion

This paper presented a large language model-enabled multi-agent collaboration method for complex task solving. By integrating hierarchical role assignment, semantic message routing, verifier-guided correction, and dynamic feedback iteration, the proposed framework improves both task completion and reasoning robustness over strong single-agent baselines. Experimental results on mathematical reasoning, planning, and multi-hop information integration confirm that structured collaboration is an effective path for making LLM systems more reliable on complex tasks. Future work will focus on tighter memory control, adaptive topology search, and cost-aware collaboration policies for long-horizon real-world applications.

References

1. J. Wei et al., “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” in Proc. NeurIPS, 2022.
2. D. Zhou et al., “Least-to-Most Prompting Enables Complex Reasoning in Large Language Models,” in Proc. ICLR, 2023.
3. X. Wang et al., “Self-Consistency Improves Chain of Thought Reasoning in Language Models,” in Proc. ICLR, 2023.
4. S. Yao et al., “ReAct: Synergizing Reasoning and Acting in Language Models,” in Proc. ICLR, 2023.
5. S. Yao et al., “Tree of Thoughts: Deliberate Problem Solving with Large Language Models,” in Proc. NeurIPS, 2023.
6. T. Schick et al., “Toolformer: Language Models Can Teach Themselves to Use Tools,” in Proc. NeurIPS, 2023.
7. A. Madaan et al., “Self-Refine: Iterative Refinement with Self-Feedback,” arXiv preprint arXiv:2303.17651, 2023.
8. N. Shinn et al., “Reflexion: Language Agents with Verbal Reinforcement Learning,” in Proc. NeurIPS, 2023.
9. Y. Du et al., “Improving Factuality and Reasoning in Language Models through Multiagent Debate,” arXiv preprint arXiv:2305.14325, 2023.
10. M. Besta et al., “Graph of Thoughts: Solving Elaborate Problems with Large Language Models,” in Proc. AAAI, 2024.
11. G. Li et al., “CAMEL: Communicative Agents for ‘Mind’ Exploration of Large Language Model Society,” in Proc. NeurIPS, 2023.
12. S. Hong et al., “MetaGPT: Meta Programming for a Multi-Agent Collaborative Framework,” in Proc. ICLR, 2024.
13. W. Chen et al., “AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors,” in Proc. ICLR, 2024.
14. Q. Wu et al., “AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation,” in Proc. COLM, 2024.
15. C. Qian et al., “ChatDev: Communicative Agents for Software Development,” in Proc. ACL, 2024.
16. K. Cobbe et al., “Training Verifiers to Solve Math Word Problems,” arXiv preprint arXiv:2110.14168, 2021.
17. K. Valmeekam et al., “PlanBench: An Extensible Benchmark for Evaluating Large Language Models on Planning and Reasoning about Change,” in Proc. NeurIPS Datasets and Benchmarks, 2023.
18. H. Trivedi et al., “MuSiQue: Multihop Questions via Single-hop Question Composition,” Trans. Assoc. Comput. Linguistics, vol. 10, pp. 539–554, 2022.