MPAS: Breaking Sequential Constraints of Multi-Agent Communication Topologies via Individual-Epistemic Message Propagation

Jingxuan Yu[1], Ju Jia[1, 2*], Simeng Qin[3], Xiaojun Jia[4], Siqi Ma[5], Yihao Huang[6], Yali Yuan[1], Guang Cheng[1]

[1]School of Cyber Science and Engineering, Southeast University, Nanjing, China
[2]Engineering Research Center of Blockchain Application, Supervision and Management (Southeast University), Ministry of Education, China
[3]School of Management, Northeastern University, China
[4] College of Computing and Data Science, Nanyang Technological University, Singapore
[5]School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, Australia
[6]School of Computing, National University of Singapore, Singapore
{yujingxuan24, jiaju}@seu.edu.cn, qinsimeng@neuq.edu.cn, jiaxiaojunqaq@gmail.com, siqim@uow.edu.au, huangyihao22@gmail.com, {yaliyuan, chengguang}@seu.edu.cn

Abstract

Large language model (LLM)-driven agents are designed to handle a wide range of tasks autonomously. As tasks become increasingly composite, the integration of multiple agents into a graph-structured system offers a promising solution. Recent advances mainly architect the communication order among agents into a specified directed acyclic graph, from which a one-by-one execution can be determined by topological sort. However, sequential architectures restrict the diversity of the information flow, hinder parallel computation, and exhibit vulnerabilities to potential backdoor threats. To overcome underlying shortcomings of sequential structures, we propose a node-wise multi-agent scheme, named **m**essage **p**assing multi-**a**gent **s**ystem (MPAS). Specifically, to parallelize the communication across agents, we extend the message propagation mechanism in graph representation learning to multi-agent scenarios and introduce our individual-epistemic message propagation. To further enhance expressiveness and robustness, we investigate three self-driven message aggregators. To achieve desired working flows, collaborative connections can be optimized without constraints. The experimental results reveal that compared to state-of-the-art sequential designs, MPAS could architect more advanced algorithms in 93.8% of the evaluations, reduce the average communication time from 84.6 seconds to 14.2 seconds per round on AQuA, and improve resilience against backdoor misinformation injection in 94.4% tests.

Code — https://github.com/rkxuan/MPAS

1 Introduction

Numerous agents (Liu et al. 2025; Wang et al. 2025a) driven by large language models (LLMs) have been proposed in many high-impact applications, ranging from code writing (Zhang et al. 2024c, 2025c), question answering (Wu et al. 2025), recommendation (Huang et al. 2025; Wang et al. 2025b) to out-of-distribution detection (Lee et al. 2025). However, in complex tasks such as autonomous driving or smart city management, the single agent will struggle to analyze the problem from multiple perspectives and adapt to the dynamic environment. Therefore, to satisfy composite requirements in real-world applications, multi-agent systems have emerged as a crucial solution, which could integrate various functions from different individuals. Even more exciting, many studies (Chen et al. 2024; Zhuge et al. 2024; Hong et al. 2024; Tianyidan et al. 2025) demonstrate that these systems often outperform individual intelligence due to a plurality of knowledge domains.

The communication topology of a system, i.e., how information flows from one agent to another, is closely related to task performance, operational efficiency, and robustness against potential attacks (Pan et al. 2025). Early approaches intuitively construct the system topologies by adopting some well-known structures (e.g., chain, star, tree) based on task requirements and agent roles. Since optimal communication topologies vary with scenarios and the composition of agents, handcrafted designs typically require exhaustive testing and tuning. To architect suitable structures automatically, recent advances (Zhuge et al. 2024; Zhang et al. 2024b, 2025a) optimize the topology as a learnable objective, which leads to significant improvement in expressiveness, task adaptation, and adversarial robustness.

However, prior investigations are mostly limited to sequential structures, i.e., the directed acyclic graphs. The underlying reason lies in the assumption that agents in a graph-structured system must be executed iteratively. Therefore, if non-sequential structures (e.g., cycles) exist, the system will be stuck in a dead loop. Concretely, there are several potential shortcomings originating from the strict sequential constraints in multi-agent systems: (1) the unidirectional execution queue restricts the node-to-graph interaction, which directly weakens the flexibility of cooperation; (2) the sys-

*Corresponding author

tem employs a one-by-one communication, where each node remains blocked until its predecessor completes the procedure, inevitably introducing latency as the number of agents grows; and (3) the contextual coherence is more likely to rely on topologically critical nodes (e.g., cut vertices between connected components). If such nodes are compromised via backdoors (Wang et al. 2024; Yang et al. 2024), attackers can easily intercept ongoing information flows and inject malicious requests (Yan et al. 2025).

In this work, to break sequential constraints of multi-agent communication topologies, we propose a node-wise multi-agent scheme, named **m**essage **p**assing multi-**a**gent **s**ystem (MPAS). Concretely, to coordinate interactions among agents, we transfer the message propagation mechanism to agentic scenarios, and implement the message propagation protocol that has three parallelized sub-operations driven by individual-epistemology of agents. To further motivate agents to distill meaningful context and filter malicious instructions, we investigate how to realize self-driven message aggregators by the corresponding prompts. To determine task-aware topologies, collaborative connections can be optimized as learnable parameters and meritorious components will not be discarded by topological sampling. In summary, our contributions are as follows:

- We investigate the advantages of the node-wise multi-agent system. To the best of our knowledge, this is the first study to thoroughly evaluate the non-sequential communication topology w.r.t. effectiveness, time-efficiency, and robustness against local misinformation injection.
- We introduce our MPAS, a node-wise multi-agent scheme that supports individual-level parallelization. In particular, the communication protocol of our scheme is isomorphic to the message propagation mechanism of MPNNs, which facilitates node-to-graph iterations, realizes step-wise parallelization, and strengthens system robustness against potential attacks.
- We conduct extensive experiments on six real-world datasets. The results reveal that our MPAS improves the success rate of task-solving over 0.4%$\sim$7.6%, shortens the running time up to 83.1% and reduces the threats of backdoor attacks by 7.4%$\sim$26.3% compared to state-of-the-art (SOTA) sequential topological designs.

2 Related Work

Message Passing Neural Networks. The message passing neural network (MPNN) (Kipf and Welling 2017; Hamilton, Ying, and Leskovec 2017; Veličković et al. 2018; Zheng et al. 2025; Liang et al. 2025; Jia et al. 2025b,a), is one of the most widely used graph neural network frameworks, which can learn powerful representations on given graphs at scale due to its message propagation mechanism. Specifically, in an MPNN layer, each node encapsulates the representation as a message to its in-neighbors, subsequently aggregates received messages, and eventually updates the representation. In this way, each MPNN layer progressively expands the propagation range of information by one hop (Zhang et al. 2024a). Our work extends the concept of message propagation mechanism to the multi-agent collaboration scenarios and proposes the message propagation protocol, in which agents spontaneously generate and aggregate messages based on their individual epistemology.

Task-Solving Multi-Agent Systems. The underlying mechanism of the multi-agent system lies in the decomposition of a compound task into multiple sub-algorithms, which are executed by corresponding domain-specific agents (Tran et al. 2025). For example, software-related agents, such as debugger, architect, and engineer, are predefined for code generation by advanced frameworks (Hong et al. 2024; Holt, Luyten, and van der Schaar 2024). For graph data synthesis, four specialized agents (manager, perception, enhancement, and evaluation) are orchestrated by GraphMaster (Du et al. 2025). In addition to role assignment, a well-structured communication topology between agents could significantly improve system performance and robustness (Pan et al. 2025).

Multi-Agent Topological Designs. Handcrafted topologies are usually based on some widely used structures, such as chain (Qian et al. 2023), tree (Yao et al. 2023), star (Qian et al. 2024). In contrast, optimized topologies are constructed via automatic learning techniques. For example, GPTSwarm (Zhuge et al. 2024) reformulates the discrete search over feasible topologies to the continuous learning for an expected graph distribution. G-Designer (Zhang et al. 2024b) utilizes the graph auto-encoder to encode task-specific nodes (i.e., agents) and then decode a scenario-adaptive communication topology. Agent-Prune (Zhang et al. 2025a) introduces a sparse structural optimization to prune redundant connections. Nevertheless, prevalent designs are mainly limited to the sequential structure, which is a narrow subset of node-wise topologies. In addition, NetSafe (Yu et al. 2024) explores the node-wise topology to some extent, but it focuses primarily on simple communication and security aspects.

3 Preliminaries

Message Propagation Mechanism. In general, an MPNN layer can be formed as (Fey and Lenssen 2019):

$$h_i^t = \gamma_\theta(h_i^{t-1}, \bigoplus_{v \in \mathcal{N}(i)} \phi_\theta(h_i^{t-1}, h_v^{t-1}, e_{iv})) \tag{1}$$

where h_i is the representation of the node and h_i^0 is the original feature of the node, e_{iv} is the edge feature, $\bigoplus$ denotes a differentiable permutation-invariant function (e.g., sum, mean, max or min), γ_θ and ϕ_θ denote learnable functions (e.g., MLP). Based on Eq.(1), the message propagation mechanism can be decomposed into three sub-operations:

$$m_i^t = M_\theta(h_i^{t-1}) \tag{2}$$

$$a_i^t = A_\theta(m_i^t, \{m_v^t, e_{iv} | v \in \mathcal{N}(i)\}) \tag{3}$$

$$h_i^t = U_\theta(h_i^{t-1}, a_i^t) \tag{4}$$

where M_θ, A_θ, U_θ are the message generation, aggregation, and update function, respectively. From the perspective of the graph, each step can be performed in parallel across nodes without underlying topological constraints, i.e.,

Algorithm 1: Sequential graph execution

Input: Computation graph $G = (N, E, F, P, o)$, input x, empty context z^0 for each node
Parameter: Communication rounds T
Output: Final answer z_o

for $t = 1, 2, ..., T$ **do**
  for i in TopologicalSort(N, E) **do**
    $z_i^t = f_i(x, z_i^{t-1}, \{z_v^t | v \in \text{pre}(i)\}, p_i)$
  end for
end for
return $z_o = f_o(x, \{\}, \{z_i^T | i \in N\}, p_o)$

Algorithm 2: Message propagation protocol

Input: Computation graph $G = (N, E, F, P, o)$, input x, empty context h^0 for each node
Parameter: Communication rounds T
Output: Final answer h_o

for $t = 1, 2, ..., T$ **do**
  parallelize message generation to generate $\{m_i^t\}$
  parallelize node aggregation to generate $\{a_i^t\}$
  parallelize response update to generate $\{h_i^t\}$
end for
return $h_o = f_o(x, \{\}, \{h_i^T | i \in N\}, p_o)$

node-wise operations. Therefore, message propagation operations of MPNN inspire us to design message propagation protocol for multi-agent communications beyond sequential structures. After T layers, a readout operation P can be implemented to obtain the graph representation h_g:

$$h_g = P(\{h_i^T | i \in N\}) \tag{5}$$

Multi-Agent Systems as Graphs. LLM-driven agents can be collaborated as a directed graph G as a tuple (N, E, F, P, o), where N is the set of nodes (agents), $E \subset N \times N$ is the topology formally represented as the set of edges (directed inter-agent links), $F = \{f_i | i \in N\}$ is a set of agent computational routines controlled by the prompt set $P = \{p_i | i \in N\}$, and o is the final output operation. In the t-th round of communication, node i receives raw input x, previous self-message z_i^{t-1}, messages from in-neighbors $\{z_v^t | v \in \mathcal{N}(i)\}$, system prompt p_i, and outputs string z_i^t in natural language, i.e., $z_i^t = f_i(x, z_i^{t-1}, \{z_v^t | v \in \mathcal{N}(i)\}, p_i)$. After T rounds, the final output operation maps all messages to a system answer, i.e., $z_o = f_o(x, \{\}, \{z_i^T | i \in N\}, p_o)$. Prevalent topological designs focus mainly on directed acyclic graphs (Tran et al. 2025), in which the nodes are executed in an iterable manner according to the topological sort as summarized in Algorithm 1.

4 Our Proposed MPAS

4.1 Overview of Our Scheme

Objective. In this paper, we aim to establish a node-wise multi-agent scheme that can be organized with an arbitrary topology, through which the message propagation mechanism is extended to LLM multi-agent scenarios. In addition, we wise to give the first insight on how to further enhance the feasibility of node-wise systems in the real-world tasks.
Overview. Firstly, we propose our message propagation protocol, which is formally similar to the MPNN layer, as shown in Figure 1. After a communication round, each node could extend the interaction range by one hop. Secondly, we investigate self-driven message aggregators, which are essential for the diversity and validity of information exchange. Eventually, we reveal that meritorious connections will not be pruned after topology optimization in our scheme, unlike in sequential frameworks. In general, the execution on the MPAS is summarized as Algorithm 2.

4.2 Individual-Epistemic Message Propagation

Parallelization is performed across nodes and edges in the MPNN due to the message propagation mechanism. Building upon this insight, we propose the message propagation protocol for parallel communication on node-wise topologies, which is also composed of three sub-operations, i.e., message generation, node aggregation, and response update. However, unlike in graph learning scenarios, these sub-operations are driven by individual-epistemology of agents rather than the universal template (e.g., the deep layer). In this way, local interactions effectively broaden the scope of thinking while averting homogenization among nodes.

Message Generation. At the beginning of the t-th round, each agent independently generates an initial insight on the raw input x as the message to share domain-specific knowledge:

$$m_i^t = f_i(x, h_i^{t-1}, \{\}, p_{i,M}) \tag{6}$$

where h_i^{t-1} represents the textual output from the previous round and $h_0 = \{\}$, $p_{i,M}$ is the message generation prompt, such as "Generate a brief analysis on the given question $< x >$ based on your role and the previous output $< h_i^{t-1} >$." The message m_i^t is then sent to out-neighbors.

Node Aggregation. Subsequently, each agent independently serves as the message aggregator to consolidate received messages into an external knowledge supplement:

$$a_i^t = f_i(x, m_i^t, \{m_v^t | v \in \mathcal{N}(i)\}, p_{i,A}) \tag{7}$$

where a_i^t represents an aggregated message that contains t-ego information, and $p_{i,A}$ is the aggregation prompt to achieve various message aggregators, which will be discussed further in Section 4.3.

Response Update. After node aggregation, each agent updates their previous response h_i^{t-1} to an enriched perspective h_i^t based on the raw input x and the aggregated message a_i^t:

$$h_i^t = f_i(x, h_i^{t-1}, \{a_i^t\}, p_{i,U}) \tag{8}$$

where $p_{i,U}$ is the update prompt, such as "Update your output $< h_i^{t-1} >$ on the given question $< x >$ based on your role and messages from other experts $< a_i^t >$."

After T rounds, the final decision operation generates a textual output as the system answer, which can be treated

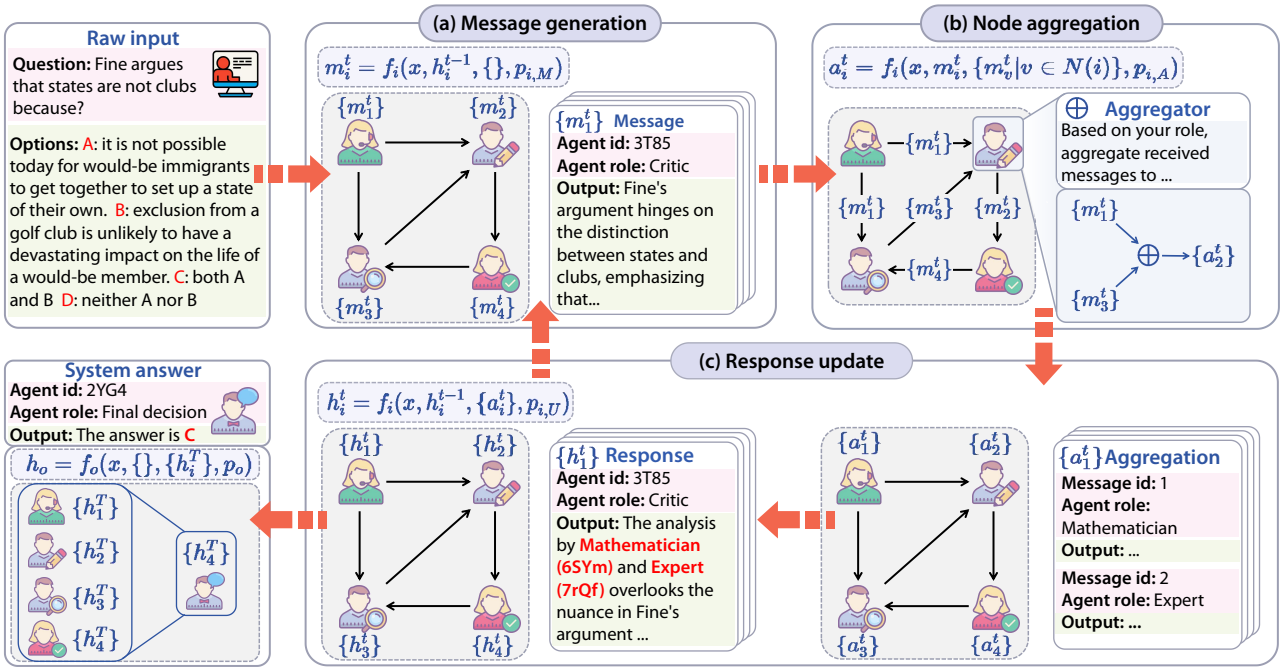

Figure 1: Our message propagation protocol has three modules: Message generation, node aggregation, and response update.

as a virtual node (Gilmer et al. 2017) to readout all node representations (responses):

$$h_o = f_i(x, \{\}, \{h_i^T | i \in N\}, p_o) \tag{9}$$

where p_o is the system prompt to achieve thought alignment and guide decision making.

4.3 Self-Driven Message Aggregators

The vanilla message aggregator is a simple concatenation:

$$a_i^t \leftarrow \text{Concat}(m_i^t, \{m_v^t | v \in \mathcal{N}(i)\}) \tag{10}$$

which lacks flexibility in information integration and fails to stimulate the environmental perception of agents. Recall that in MPNN, the message propagation relies on deep layers, while in our scheme, the message propagation is guided by individual-epistemology. Following this insight, we investigate three self-driven message aggregators.

- **Selective Aggregator**: This aggregator defines universal edge prompts (UEPs) for evaluating the reliability of in-neighbors, and induces agents to selectively retain helpful messages, such as "Evaluate received messages $<\{m_v^t\}>$ from following perspectives:$<$ UEPs $>$, and tell me which messages will help you further reason on the given question $< x >$." As a result, edge information (e_{iv}) is supplemented and irrelevant communication edges will be spontaneously pruned, which potentially achieves dynamic topologies and improves system robustness.
- **Attention Aggregator**: This aggregator is inspired by the shared attention mechanism in GAT (Veličković et al. 2018), which learns the importance of a received message to self-message. Similarly, the attention aggregation prompt guides agents to concentrate on the most relevant information of the received messages to self-messages based on the role and raw input, such as "Extract information of received messages $<\{m_v^t\}>$ that supplements or differs from your message $< m_i^t >$ based on the given question $< x >$ and your role." In this way, individual epistemology is effectively leveraged to promote the efficiency of knowledge extraction.
- **Selective Attention Aggregator**: This aggregation is the combination of the selective aggregator and the attention aggregator. The aggregation prompt instructs agents to distill the valuable insights from the selected messages.

4.4 Optimizing MPAS

Given the inherent complexity of topology discovery, it is advisable to employ automated learning techniques to architect the desired topology. The underlying idea is to define a parameterized probabilistic distribution over potential topologies as $E' \sim D_\theta$ (Zhuge et al. 2024). Subsequently, the utility loss is set to optimize the distribution to expected direction via gradient descent methods (Ruder 2016):

$$\min \mathcal{L}_u = -u_\tau(G') \text{ s.t. } G' \sim (N, D_\theta, F, P, o) \tag{11}$$

where u_τ is the utility function for tasks τ. Technologically, a learnable parameter $\theta_{iv} \in \mathcal{R}$ can be assigned to each edge and then normalized to $\theta'_{iv} \in [0, 1]$ as its probability. Therefore, the joint distribution for a node-wise topology is:

$$D_\theta = \prod_{i \neq v} (\theta_{iv})^{e_{iv}} (1 - \theta_{iv})^{1 - e_{iv}} \tag{12}$$

Then mini-batches can be sampled from the distribution where each mini-batch contains $\{E_1, E_2, \ldots, E_m\} \sim D_\theta$.

Eventually, the batch gradient of the Eq.(11) is estimated as:

$$\nabla_\theta \mathcal{L}_u \approx -\frac{1}{m}\sum_{i=1}^{m} u_\tau(G_i)\nabla_\theta \log(p_\theta(E_i)) \tag{13}$$

After training, a desired topology can be sampled from the optimized distribution:

$$E = \{e_{iv} = 1 | \theta'_{iv} > \delta\} \tag{14}$$

where δ is set to 0.5 in our scheme. However, if the communication topology is limited to the directed acyclic graph, an iterable sampling will be exploited after optimization, during which the learned connections could be lost. For example, even though a connection has a high probability (i.e., $\theta' = 0.98$), it will still be discarded if its addition leads to a cycle. In contrast, meritorious edges on the node-wise structure can be reserved for real-world applications.

5 Experimental Results

In this section, we conduct experimental evaluations to address the following research questions:

- **RQ1:** Whether MPAS could outperform SOTA sequential topological designs across various real-world scenarios?
- **RQ2:** Can our parallelized message propagation protocol alleviate communication congestion as scale increases?
- **RQ3:** Does our scheme architect node-wise communication topologies that improve system robustness against local misinformation injection?
- **RQ4:** Can our self-driven message aggregators further enhance system performance and security?
- **RQ5:** How to illustrate that each round of communication in MPAS could expand the range of message propagation?

5.1 Experimental Settings

Datasets. We evaluate systems on three categories of widely-used dataset: (1) a general reasoning dataset, MMLU (Hendrycks et al. 2021); (2) mathematical reasoning datasets, including GSM8k (Cobbe et al. 2021), MultiArith (Roy and Roth 2015), SVAMP (Patel, Bhattamishra, and Goyal 2021), AQuA (Ling et al. 2017); and (3) a code generation dataset, HumanEval (Chen et al. 2021).

Multi-Agent Systems. The agents are driven by GPT-4o-mini (Achiam et al. 2023), whose achievement is consistent with previous research (Zhang et al. 2024b, 2025a). We will randomly select agents from the list and subsequently organize them into a system that facilitates two rounds of communication. All evaluations will be repeated five times.

Baselines. For individual intelligence, we randomly select an agent as the task-solver. For handcraft topologies, we select chain, star, tree, complete graph, and random graph, which are defined in (Qian et al. 2024). For sequential optimized topologies, we choose three SOTA works, including, GPTSwarm (Zhuge et al. 2024) , G-Designer (Zhang et al. 2024b), and Agent-Prune (Zhang et al. 2025a).

5.2 Evaluation of Task Performance (RQ1)

To uncover the limitations of sequential communication, we compare our MPAS scheme with mentioned sequential multi-agent systems, involving 5 agents. To ensure fairness, both the MPAS and the sequential framework concatenate messages from in-neighbors without further distillation or thinking in this experiment. From the results shown in Table 1, we can draw the following conclusions:

The MPAS scheme is more effective than sequential communication frameworks. On the one hand, when topological structures are handcrafted, MPAS outperforms sequential systems in all evaluated cases and significantly improves the accuracy by 0.9%$\sim$6.4%. On the other hand, when topological structures are optimized, MPAS exhibits a marked advantage over SOTA sequential topological designs, with 0.2%$\sim$2.3% improvement in six scenarios.

The topology optimization on node-wise topologies automatically discovers high-performance algorithms. To illustrate this, optimized structures are presented in Figure 2, which confirms that there are task-aware communication topologies beyond the sequential components.

5.3 Evaluation of Running Time (RQ2)

We report the running time of the MPAS as the number of agents increases and compare our scheme with G-Designer and Agent-Prune, both of which employ a more lightweight structure to enhance efficiency. Table 2 presents the relationships between the running time and the agent number in MMLU and AQuA. The results illustrate that:

As the number of agents increases, the execution time of **the parallel execution rises gradually**, whereas the execution time of **the sequential execution escalates sharply**. Furthermore, when the scale of the system is relatively large (e.g., with 9 or 11 agents), parallel algorithms demonstrate a substantial time-efficiency compared to sequential algorithms. Therefore, our proposed scheme is particularly applicable for the large-scale integration of tool-intensive agents.

5.4 Evaluation of Robustness (RQ3)

We assess the robustness of the optimized MPAS against local misinformation injection (Fang, Easwaran, and Genest 2025; Zhang et al. 2025b). To understand worst-case scenarios, we assume that adversarial attackers can directly control several agents to mislead the system during training, while backdoor attackers can do so during testing. Notably, in this setting, GPTSwarm and G-Designer have demonstrated a certain level of adversarial robustness, serving as the baselines. The system involves 3 trustful agents and 3 agents controlled by the attacker, following the terminology of (Zhuge et al. 2024). The results on MMLU and AQuA are shown in Table 3, which confirms the following insights:

In adversarial scenarios, any structure-wise topology optimizations safeguard the collaboration. During training, harmful connections will be pruned and adversarial agents will be isolated. Therefore, the workflow will not be altered in the middle of any structure. **In backdoor scenarios, optimized MPAS provides a more trustworthy application.** After training, the topology is frozen and then ap-

System	Topological design	MMLU	GSM8K	MultiArith	SVAMP	AQuA	HumanEval
Individual	Single node	68.5±0.8	89.9±0.9	89.1±1.7	87.0±1.0	76.6±0.9	79.1±1.1
Sequential system	Chain	69.4±1.2	92.4±0.8	91.8±1.5	88.6±1.2	77.6±1.0	81.1±1.1
	Star	71.4±1.8	93.8±0.9	90.3±1.4	89.6±1.9	76.1±1.7	82.7±1.2
	Tree	71.8±2.7	92.5±2.1	93.2±1.0	89.4±1.4	75.6±1.9	83.6±1.4
	Random graph	71.2±3.9	90.4±2.7	91.5±4.3	89.6±2.4	76.7±2.6	82.4±2.3
	Swarm	73.9±1.3	94.5±0.9	94.5±1.3	91.8±1.6	77.6±0.8	84.5±1.0
	G-Designer	76.9±2.2	93.9±1.5	94.2±1.6	90.8±1.2	78.7±0.9	83.9±1.2
	Agent-Prune	73.2±2.0	93.7±2.4	93.7±1.3	91.2±1.2	76.5±1.1	84.9±1.4
MPAS	Chain	75.8±2.0	94.5±1.7	94.7±1.1	90.2±1.2	78.9±1.4	83.5±1.0
	Star	75.0±1.4	94.7±1.5	93.8±1.2	91.4±0.5	77.9±1.0	85.0±0.9
	Tree	75.0±2.1	93.7±1.0	94.8±1.8	91.4±1.4	76.4±1.6	84.5±1.4
	Complete graph	76.5±1.2	94.0±1.4	93.7±1.3	91.0±1.5	77.8±1.6	83.1±0.9
	Random graph	75.7±3.4	93.0±2.4	95.3±3.0	91.0±2.3	77.5±2.3	85.8±2.1
	Optimized topology	**77.3±1.3**	**95.9±1.1**	**96.7±1.7**	**92.0±1.1**	**80.0±0.8**	**87.2±1.2**

Table 1: Performance (in %) comparison among three type of systems, including individual intelligence, sequential multi-agent systems and our MPASs. We report the mean and standard deviation across 5 runs.

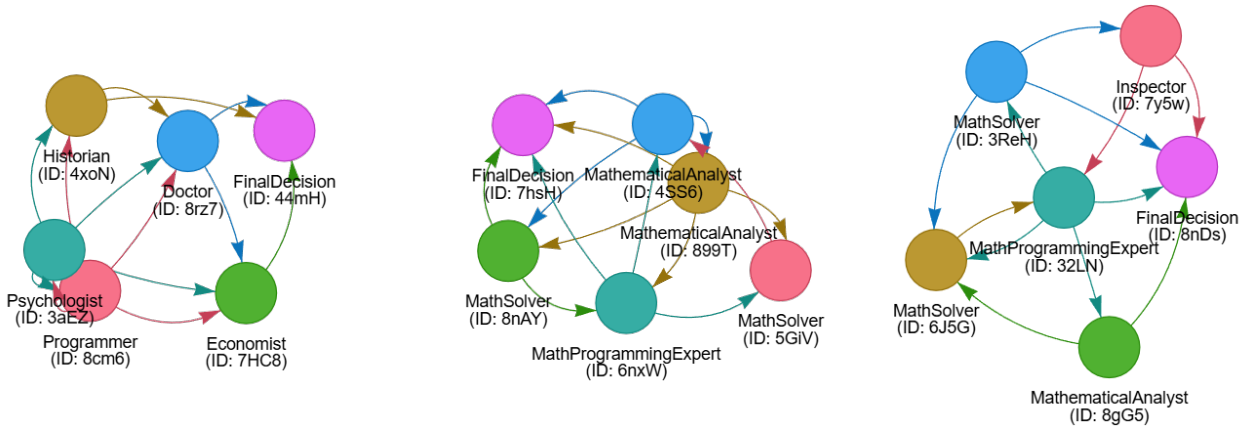

Figure 2: The optimized topologies of MPAS on MMLU, GSM8K, AQuA (from left to right).

Tasks	Systems	Number of Agents				
		3	5	7	9	11
MMLU	MPAS	**1.9**	**2.1**	**2.4**	**2.7**	**3.1**
	G-Designer	3.4	4.1	5.2	6.5	8.1
	Agent-Prune	3.3	3.9	4.9	5.3	7.0
AQuA	MPAS	7.8	**8.2**	**9.0**	**11.3**	**14.2**
	G-Designer	**7.6**	10.7	19.6	47.4	84.6
	Agent-Prune	8.2	10.5	17.5	32.5	60.5

Table 2: Average running time (in seconds) per communication round on MMLU and AQuA.

Tasks	Systems	Attacks		
		Clean	Poison	Backdoor
MMLU	MPAS	**76.4**	**75.1**	**72.1**
	G-Designer	75.2	73.9	62.1
	GPTSwarm	73.8	71.2	45.8
AQuA	MPAS	**78.3**	**77.6**	**74.3**
	G-Designer	77.9	77.0	72.5
	GPTSwarm	77.4	76.0	58.5

Table 3: Performance (in %) of robustness test on MMLU and AQuA.

plied to tasks. As a result, the static topology cannot dynamically filter out abnormal relationships. However, our scheme facilitates a more resilient solution: Although part of the system is compromised, the remaining components can still exchange reliable information without relying on successors.

5.5 Study on the Message Aggregators (RQ4)

We conduct an evaluation of the advantages offered by three advanced message aggregators compared to the vanilla textual concatenation. From Table 4, we observe the following key points: (1) **Attention aggregator enhances expressive-**

Scenario	Message aggregator	MMLU	GSM8K	MultiArith	SVAMP	AQuA	HumanEval
Clean	Vanilla	77.3±1.3	95.9±1.1	**96.7±1.7**	92.0±1.1	80.0±0.8	87.2±1.2
	Selective	76.6±0.6↓	96.6±0.4↑	95.0±0.7↓	92.8±0.4↑	81.1±1.0↑	85.7±0.8↓
	Attention	78.7±0.3↑	**97.3±0.6**↑	95.8±0.3↓	**93.2±0.7**↑	**82.8±1.8**↑	**89.1±0.6**↑
	Selective Attention	**79.1±0.7**↑	96.9±1.0↑	95.2±1.0↓	92.8±0.4↑	82.0±1.4↑	86.1±1.2↓
Poison	Vanilla	74.8±0.9	93.0±1.0	94.2±1.4	90.0±1.1	78.3±1.2	83.4±1.4
	Selective	**78.3±0.6**↑	**95.1±0.6**↑	**94.8±0.3**↑	**91.8±0.7**↑	**79.8±1.2**↑	84.3±1.3↑
	Attention	76.7±1.2↑	91.7±0.9↓	93.3±2.0↓	90.8±1.3↑	74.8±1.5↓	83.6±1.3↑
	Selective Attention	75.4±0.5↑	93.9±1.2↑	93.2±1.0↓	91.4±1.0↑	77.7±1.2↓	**84.6±1.5**↑
Backdoor	Vanilla	72.5±2.0	91.9±2.0	91.5±2.8	88.0±1.9	74.3±2.6	81.4±1.8
	Selective	**77.1±0.8**↑	**94.5±1.3**↑	**93.5±1.7**↑	**90.0±1.8**↑	**75.6±2.3**↑	**82.0±1.2**↑
	Attention	67.2±1.6↓	89.3±2.6↓	85.8±3.4↓	85.2±2.0↓	72.3±1.5↓	80.7±2.4↓
	Selective Attention	71.0±1.1↓	92.5±1.8↑	87.7±1.8↓	86.4±2.7↓	73.9±2.1↓	81.7±1.7↑

Table 4: The results (in %) of self-driven message aggregators on real-world tasks compared to vanilla message concatenation.

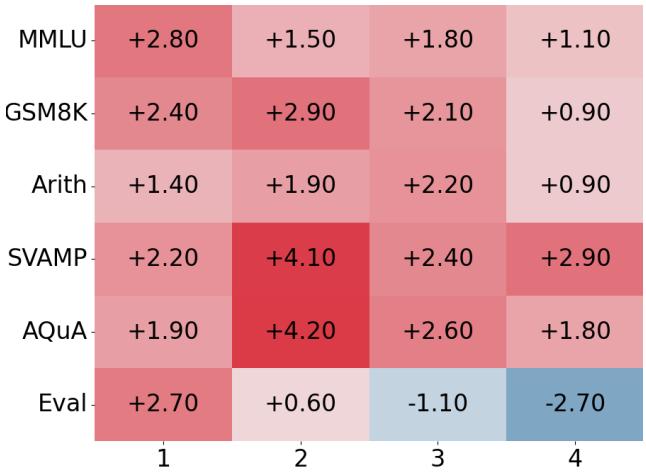

Figure 3: The performance improvement (in %) with increasing communication rounds compared to the vanilla multi-agent integration (i.e., communication rounds $T = 0$).

ness in trustworthy scenarios. It further improves the performance over 1.2%∼2.8% across 5 scenarios. With regard to the accuracy drop on MultiArith, one of the possible reasons is that introducing complexity to such a simple task is redundant. However, it is more vulnerable to malicious manipulation due to its concentration on task-related information; (2) **Selective aggregator improves robustness in noise-injected environments.** Although its message filtering mechanism is redundant in safe environments, it significantly boosts robustness by 0.6%∼3.5% under the poison scenarios and by 0.6%∼4.6% under the backdoor scenarios. (3) **Selective attention aggregator inherits the disadvantages of both.** In a trustworthy environment, it will filter out valid messages; while in a noise-injected environment, it will concentrate on malicious messages.

5.6 Study on the Message Propagation (RQ5)

We further present how an additional communication round extends the one-hop scope of message propagation. Specifically, we establish a system composed of five agents, whose topology is a bidirectional cycle.

Theoretically, under this setup, the following results are expected: (1) when the number of communication rounds is zero, each agent is isolated to express its initial thought, and then a final decision operation is directly implemented, which serves as the baseline; (2) when the number of communication rounds is one or two, the range of message propagation increases, which leads to more expressive organizations; and (3) when the number of communication rounds exceeds two, performance will not improve significantly for the reason that the propagation range of the five-agent system is limited to two-hop graph structure.

Figure 3 illustrates the performance improvement as the number of rounds increases, which mostly aligns with our theoretical analysis. In addition, excessive information exchange can lead to a dramatic decline for small-scale graphs.

6 Conclusion

In this study, we propose a node-wise multi-agent scheme inspired by the message propagation mechanism, named MPAS, to overcome inherent limitations of prevalent sequential structures and support parallelized communication within any topologies. The following conclusions can be drawn from our extensive experiments across real-world datasets: (1) more advanced intelligence can be represented as the combination of node-wise components beyond sequential structures; (2) as a system involves more agents and sub-systems, MPAS could be a more time-efficient alternative than the one-by-one execution; (3) the workflow of MPAS is resilient to local misinformation injection because node-to-graph iteration is not restricted to the unidirectional structure; (4) the robustness and flexibility of our scheme can be further improved by more sophisticated message aggregators. In addition, we recommend that developers investigate more communication rounds in larger-scale systems to expand the range of message propagation. In the near future, we will investigate multi-agent systems that take advantage of MPAS to address pressing real-world challenges.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62402106, U22B2025, 62172093), the Natural Science Foundation of Jiangsu Province of China (Grant No. BK20241272), the Fundamental Research Funds for the Central Universities (Grant No. 2242025K30025), and the Start-Up Research Fund of Southeast University (Grant No. RF1028623129).

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Chen, G.; Dong, S.; Shu, Y.; Zhang, G.; Sesay, J.; Karlsson, B.; Fu, J.; and Shi, Y. 2024. AutoAgents: a framework for automatic agent generation. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, 22–30.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. D. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Du, E.; Li, X.; Jin, T.; Zhang, Z.; Li, R.-H.; and Wang, G. 2025. Graphmaster: Automated graph synthesis via llm agents in data-limited environments. *arXiv preprint arXiv:2504.00711*.

Fang, X.; Easwaran, A.; and Genest, B. 2025. Adaptive Multi-prompt Contrastive Network for Few-shot Out-of-distribution Detection. In *International Conference on Machine Learning*.

Fey, M.; and Lenssen, J. E. 2019. Fast graph representation learning with PyTorch Geometric. *arXiv preprint:1903.02428*.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*, 1263–1272. PMLR.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*.

Holt, S.; Luyten, M. R.; and van der Schaar, M. 2024. L2MAC: Large Language Model Automatic Computer for Extensive Code Generation. In *The Twelfth International Conference on Learning Representations*.

Hong, S.; Zhuge, M.; Chen, J.; Zheng, X.; Cheng, Y.; Wang, J.; Zhang, C.; Wang, Z.; Yau, S. K. S.; Lin, Z.; et al. 2024. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. In *The Twelfth International Conference on Learning Representations*.

Huang, X.; Lian, J.; Lei, Y.; Yao, J.; Lian, D.; and Xie, X. 2025. Recommender ai agent: Integrating large language models for interactive recommendations. *ACM Transactions on Information Systems*, 43(4): 1–33.

Jia, J.; Li, R.; Wu, C.; Feng, Y.; Ma, S.; Wang, L.; and Deng, R. H. 2025a. Environment-Adaptive Representation Interaction for Privacy-Perceptual GNNs against Deceptive OOD Attacks. *IEEE Transactions on Information Forensics and Security*.

Jia, J.; Li, R.; Wu, C.; Ma, S.; Wang, L.; and Deng, R. H. 2025b. SIGFinger: A Subtle and Interactive GNN Fingerprinting Scheme Via Spatial Structure Inference Perturbation. *IEEE Trans. Dependable Secur. Comput.*, 22(4): 3629–3646.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

Lee, Y.; Cao, X.; Guo, J.; Ye, W.; Guo, Q.; and Chang, Y. 2025. Concept Matching with Agent for Out-of-Distribution Detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 4562–4570.

Liang, Y.; Zhang, W.; Sheng, Z.; Yang, L.; Xu, Q.; Jiang, J.; Tong, Y.; and Cui, B. 2025. Towards scalable and deep graph neural networks via noise masking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 18693–18701.

Ling, W.; Yogatama, D.; Dyer, C.; and Blunsom, P. 2017. Program Induction by Rationale Generation: Learning to Solve and Explain Algebraic Word Problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 158–167.

Liu, B.; Li, X.; Zhang, J.; Wang, J.; He, T.; Hong, S.; Liu, H.; Zhang, S.; Song, K.; Zhu, K.; et al. 2025. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*.

Pan, M. Z.; Cemri, M.; Agrawal, L. A.; Yang, S.; Chopra, B.; Tiwari, R.; Keutzer, K.; Parameswaran, A.; Ramchandran, K.; Klein, D.; et al. 2025. Why do multiagent systems fail? In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*.

Patel, A.; Bhattamishra, S.; and Goyal, N. 2021. Are NLP Models really able to Solve Simple Math Word Problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2080–2094.

Qian, C.; Cong, X.; Yang, C.; Chen, W.; Su, Y.; Xu, J.; Liu, Z.; and Sun, M. 2023. Communicative Agents for Software Development. *CoRR*.

Qian, C.; Xie, Z.; Wang, Y.; Liu, W.; Dang, Y.; Du, Z.; Chen, W.; Yang, C.; Liu, Z.; and Sun, M. 2024. Scaling Large-Language-Model-based Multi-Agent Collaboration. *CoRR*.

Roy, S.; and Roth, D. 2015. Solving general arithmetic word problems. In *Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, 1743–1752. Association for Computational Linguistics (ACL).

Ruder, S. 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Tianyidan, X.; Ma, R.; Wang, Q.; Ye, X.; Liu, F.; Tai, Y.; Zhang, Z.; Wang, L.; and Yi, Z. 2025. Anywhere: A Multi-Agent Framework for User-Guided, Reliable, and Diverse Foreground-Conditioned Image Generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 7410–7418.

Tran, K.-T.; Dao, D.; Nguyen, M.-D.; Pham, Q.-V.; O'Sullivan, B.; and Nguyen, H. D. 2025. Multi-Agent Collaboration Mechanisms: A Survey of LLMs. *arXiv preprint arXiv:2501.06322*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Wang, K.; Zhang, G.; Zhou, Z.; Wu, J.; Yu, M.; Zhao, S.; Yin, C.; Fu, J.; Yan, Y.; Luo, H.; et al. 2025a. A comprehensive survey in llm (-agent) full stack safety: Data, training and deployment. *arXiv preprint arXiv:2504.15585*.

Wang, L.; Zhang, J.; Yang, H.; Chen, Z.-Y.; Tang, J.; Zhang, Z.; Chen, X.; Lin, Y.; Sun, H.; Song, R.; et al. 2025b. User behavior simulation with large language model-based agents. *ACM Transactions on Information Systems*, 43(2): 1–37.

Wang, Y.; Xue, D.; Zhang, S.; and Qian, S. 2024. BadAgent: Inserting and Activating Backdoor Attacks in LLM Agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 9811–9827.

Wu, X.; Yang, J.; Chai, L.; Zhang, G.; Liu, J.; Du, X.; Liang, D.; Shu, D.; Cheng, X.; Sun, T.; et al. 2025. Tablebench: A comprehensive and complex benchmark for table question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 25497–25506.

Yan, H.; Ma, H.; Cai, X.; Liu, D.; Yuan, Z.; Qu, X.; Dong, J.; Guan, R.; Fang, X.; He, H.; Xie, Y.; and Zhou, P. 2025. Fit the Distribution: Cross-Image/Prompt Adversarial Attacks on Multimodal Large Language Models. In *Advances in Neural Information Processing Systems*.

Yang, W.; Bi, X.; Lin, Y.; Chen, S.; Zhou, J.; and Sun, X. 2024. Watch out for your agents! investigating backdoor threats to llm-based agents. *Advances in Neural Information Processing Systems*, 37: 100938–100964.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36: 11809–11822.

Yu, M.; Wang, S.; Zhang, G.; Mao, J.; Yin, C.; Liu, Q.; Wen, Q.; Wang, K.; and Wang, Y. 2024. Netsafe: Exploring the topological safety of multi-agent networks. *arXiv preprint arXiv:2410.15686*.

Zhang, B.; Gai, J.; Du, Y.; Ye, Q.; He, D.; and Wang, L. 2024a. Beyond Weisfeiler-Lehman: A Quantitative Framework for GNN Expressiveness. In *The Twelfth International Conference on Learning Representations*.

Zhang, G.; Yue, Y.; Li, Z.; Yun, S.; Wan, G.; Wang, K.; Cheng, D.; Yu, J. X.; and Chen, T. 2025a. Cut the crap: An economical communication pipeline for llm-based multi-agent systems. In *International Conference on Learning Representations*.

Zhang, G.; Yue, Y.; Sun, X.; Wan, G.; Yu, M.; Fang, J.; Wang, K.; and Cheng, D. 2024b. G-Designer: Architecting Multi-agent Communication Topologies via Graph Neural Networks. *CoRR*.

Zhang, K.; Li, J.; Li, G.; Shi, X.; and Jin, Z. 2024c. CodeAgent: Enhancing Code Generation with Tool-Integrated Agent Systems for Real-World Repo-level Coding Challenges. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 13643–13658.

Zhang, W.; Jia, J.; Jia, X.; Huang, Y.; Li, X.; Wu, C.; and Wang, L. 2025b. PATFinger: Prompt-Adapted Transferable Fingerprinting against Unauthorized Multimodal Dataset Usage. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 403–413.

Zhang, Y.; Jin, Z.; Xing, Y.; Li, G.; Liu, F.; Zhu, J.; Dou, W.; and Wei, J. 2025c. PATCH: Empowering Large Language Model with Programmer-Intent Guidance and Collaborative-Behavior Simulation for Automatic Bug Fixing. *ACM Transactions on Software Engineering and Methodology*.

Zheng, Y.; Li, X.; Luan, S.; Peng, X.; and Chen, L. 2025. Let your features tell the differences: Understanding graph convolution by feature splitting. In *The Thirteenth International Conference on Learning Representations*.

Zhuge, M.; Wang, W.; Kirsch, L.; Faccio, F.; Khizbullin, D.; and Schmidhuber, J. 2024. GPTSwarm: language agents as optimizable graphs. In *Proceedings of the 41st International Conference on Machine Learning*, 62743–62767.