# Integrating Strengths of Different Multi-Agent Workflows via Step-Aware Hybrid Topology Planning

**Jingxuan Yu**
Southeast University
Nanjing, China
yujingxuan24@seu.edu.cn

**Ju Jia***
Southeast University
Nanjing, China
jiaju@seu.edu.cn

**Yiqian Chen**
Southeast University
Nanjing, China
cyqian@seu.edu.cn

**Yuchong Chen**
Southeast University
Nanjing, China
chenyuchong@seu.edu.cn

**Cong Wu**
Wuhan University
Wuhan, China
cnacwu@whu.edu.cn

**Di Wu**
La Trobe University
Melbourne, Australia
d.wu@latrobe.edu.au

**Siqi Ma**
University of Wollongong
Wollongong, Australia
siqim@uow.edu.au

**Jie Gui**
Southeast University
Nanjing, China
guijie@seu.edu.cn

## Abstract

Large language model (LLM)-driven agents are proposed for a wide range of applications. As scenarios become increasingly composite, a graph-structured multi-agent system (MAS) offers a promising solution due to the orchestration of various skills. To architect a reasonable underlying workflow, recent advances mostly investigate two structures: sequential topology and decentralized topology. The former constrains the structure into a directed acyclic graph for multi-step executions but lacks sufficient scalability, while the latter leverages node-wise subgraphs for knowledge aggregation with synchronous parallelization but lacks long-term planning. Therefore, our key insight is that their strengths are mutually complementary. Driven by this motivation, we introduce step-aware hybrid topology for a MAS, where sequential connections support inter-step ordered planning and decentralized connections promote intra-step perspective integration. Concretely, to effectively coordinate different structural components, we clarify the corresponding topological definition and the protocols for step-wise serialization and parallelization. Subsequently, to dynamically refine desired structures, we introduce the parameterized distribution of a hybrid topology, which is initialized by LLM-driven graph planning. Eventually, to further reduce token consumption and time complexity, we supplement the topological density and length normalization for end-to-end learning. Extensive experiments demonstrate that the hybrid topology outperforms single-form topology in 90.4% of tasks, our agentic initialization improves learning in 88.8% of scenarios, proposed normalizations reduce token consumption by 20%$\sim$30% and time consumption by 7%$\sim$13% with a slight utility fluctuation. The code is available at https://github.com/rkxuan/HTAS.

*Corresponding author

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

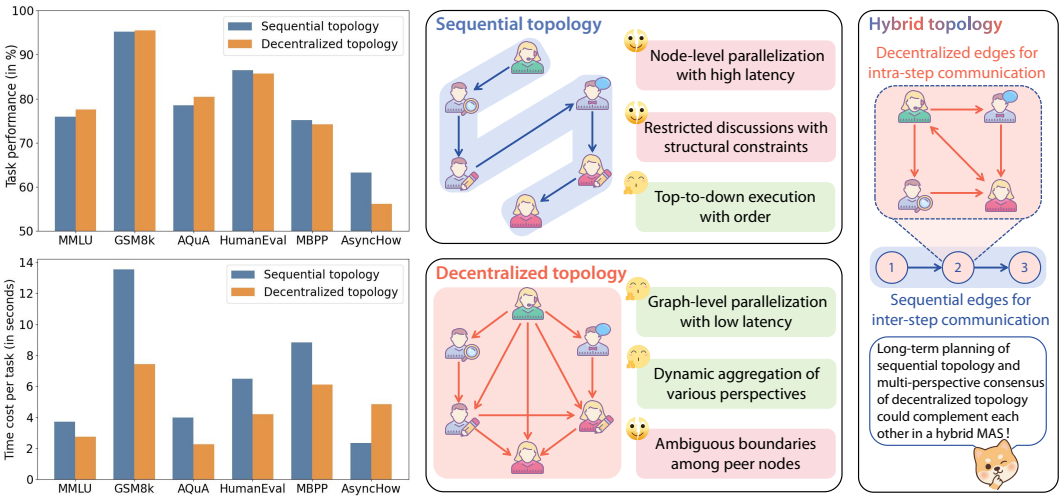



Figure 1: (Upper left) Decentralized topology are suitable for short-term tasks from MMLU to AQuA, while sequential topology are suitable for long-term tasks from HumanEval to AsyncHow. (Lower left) Decentralized topology has better time efficiency. (Right) Motivation of hybrid MAS.

## 1 Introduction

Numerous large language model (LLM)-driven agents have been designed to handle various real-world tasks, such as code writing [1, 2], domain question answering [3, 4], and fuzzy testing [5, 6]. However, in complex tasks such as autonomous driving or smart city management, a 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, the graph-structured multi-agent system (MAS) has emerged as a crucial solution, in which diverse skills could be integrated via mutual communications. Moreover, recent advanced practices [7–9] have demonstrated that, in most cases, a plurality of cross-domain knowledge outperforms individual intelligence.

The underlying topology of a graph-structured MAS, i.e., how communication flows from one agent to another, is closely related to downstream performance, task-solving efficiency, and robustness [10–14]. Early studies [15, 16] intuitively construct system topologies by adopting some well-known structures (e.g., chain, star, tree) based on prior knowledge regarding task requirements and role definitions. However, as the scale of the system increases and the task environment varies, such handcrafted designs typically require complicated testing and tuning. To architect an ideal communication topology automatically, graph-based optimization strategies [7, 8] are introduced to the agentic domain with resource savings and considerable improvements.

Previous investigations are primarily limited to one of two specific structural forms: sequential topology and decentralized topology. The former [7, 17–19] constrains the structure to a directed acyclic graph (DAG), whose isomorphic communication protocol is one-by-one execution according to the topological sort. As shown in Figure 1, it promotes long-term collaboration but also results in time latency as the channel length increases. The latter [20–22] supports any connected graphs, in which an agentic message propagation protocol is applied for communication. As a result, while graph-level synchronous parallelization is achieved, the ordered workflow has been compromised.

Driven by the above observations, our core motivation is that **the advantages of two topologies could be integrated and the disadvantages could be mitigated in a hybrid architecture**. Specifically, sequential connections naturally support orderly inter-step communication, whereas decentralized connections promote intra-step communication with unrestricted workflows. Driven by this motivation, we propose a step-aware hybrid topology planning scheme for MASs, named HTAS. Firstly, to ensure reliable system scheduling, we formalize the algorithms for execution over the underlying hybrid topology and for evaluating the validity of a given system with serialization and parallelization. Subsequently, to dynamically refine desired structures, we introduce the parameterization of the hybrid topology for optimization and alleviate the instability originating from uniform initialization via a topology-informed agent. Eventually, to further strengthen time efficiency and reduce token

costs, we introduce topological density normalization and length normalization over the distribution. In summary, our contributions are as follows:

- **Superiority identification.** We reveal that sequential topology naturally motivates step-by-step reasoning for long-term tasks with a topological order, while decentralized topology boosts the quality of multiple-role discussions with lower time latency for short-term QA tasks.
- **System discovery.** We introduce our HTAS, a hybrid topology planning scheme for step-wise workflows and gradient-based optimization. Moreover, compared with prevalent MAS topology learning strategies, our scheme leverages LLM-driven planning as a reliable alternative for initialization and discusses the topological normalizations for improved time and token efficiency.
- **Comprehensive evaluation.** We conduct comprehensive evaluations across nine tasks and prevalent LLM backbones. Results show that: (1) hybrid topology improves the accuracy from 64.0% to 68.4% on AsyncHow; (2) our optimization strategy is crucial for the implementation of desired hybrid topologies; (3) density normalization and length normalization reduce nearly 4K token consumption and 12% time per AQuA task, respectively, within 1% performance degradation.

## 2 Related Works

**LLM-Driven Multi-Agent Systems.** The underlying mechanism of the MAS lies in the decomposition of a compound task into multiple sub-algorithms, which are executed by corresponding domain-specific agents empowered by LLMs [10, 23]. For example, in a bug-fixing agent system [6], the retriever first identifies potential bugs, followed by the verifier who validates candidate patches, and ultimately, the bug fixer refines the patch until the issue is resolved or the budget is exhausted.

**Multi-Agent Topological Designs.** Generally, prevailing topologies of MAS could be divided into sequential topology [7,11,15,16] and decentralized topology [18,20,22]. The inherent communication protocol of the form is the one-by-one workflow based on the topological sort. In this way, agents are organized in a centralized or hierarchical structure. In contrast, the latter facilitates the aggregation of information in a peer-to-peer subgraph via the underlying agentic message propagation protocol [22]. Our work merges the advantages of both in the proposed hybrid topology for MASs.

**Graph-Based Agent-Planning Optimization.** The task reasoning and function calling typically require structural decomposition [15,24]. To automatically obtain the desired graph description, some studies [7, 11, 12, 19, 25] propose the parameterization of agentic graphs, and others [8, 24, 26] discuss the utilization of LLM planning. In this paper, we introduce the parameterized hybrid topology for end-to-end learning and stabilize early optimization epochs via an LLM-driven initializer.

## 3 Preliminaries

**Multi-Agent Systems as Graphs.** A MAS can be represented as a graph $G = (N, E, F, P, I, o)$, where $N$ is the set of nodes (agents), $E \subset \{N \times N\} \cup \{N \times o\}$ is the topology formally represented as a set of edges (communication channels), $F = \{f_i | i \in N\}$ is a set of agent computational routines controlled by the prompt set $P = \{p_i | i \in N\}$, $I$ is the function that maps nodes to the index of the execution order, i.e., $I(n_i) \to \mathbb{Z}^+$, and $o$ is the node (agent) for the final decision.

**Sequential Topology.** In this scenario, $G$ is limited to a DAG (including centralized and hierarchical structures), whose topological sort determines the execution order. Concretely, node $i$ receives raw input $x$, messages from predecessors $\{z_v | v \in pre(i)\}$, system prompt $p_i$, and outputs $z_i$ in natural language, formally as:

$$z_i = f_i(x, \{z_v | v \in pre(i)\}, p_i). \tag{1}$$

**Decentralized Topology.** In this scenario, $G$ is any directed connected graph, in which the underlying protocol is the agentic message propagation mechanism. That is, node $i$ first generates a self-message $m_i = f_i(x, p_i)$, which is sent to the out-neighbors. Subsequently, the self-message and received messages from neighborhood $\mathcal{N}(i)$ are updated to an output $z_i$ across nodes, denoted as:

$$z_i = f_i(x, m_i, \{m_v | v \in \mathcal{N}(i)\}, p_i). \tag{2}$$

Notably, the message generation and message update are executed synchronously across nodes.

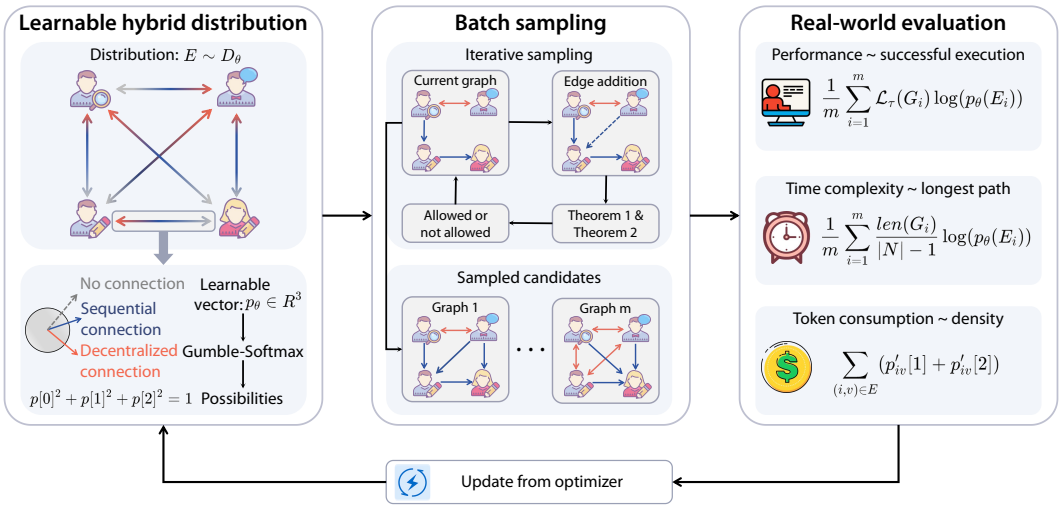



Figure 2: Our scheme is composed of three modules: (1) a parameterized hybrid distribution to be optimized; (2) an iterative sampling module to select candidates; (3) a real-world evaluation module to give gradient-based feedback w.r.t. performance, time consumption and token cost. After training, a desired topology could be sampled from the optimized distribution.

# 4 Our Proposed HTAS

**Objective.** We aim to combine the strengths of sequential topology and decentralized topology for step-informed planning and communication. In addition, we wish to provide the first insight into how to further improve the feasibility of a hybrid-structured MAS via multi-objective optimization.

**Overview.** As shown in Figure 2, the distribution of the hybrid topology is first parameterized. Subsequently, the iterative sampling is executed for candidates. Eventually, parameters are updated based on evaluations w.r.t. utility, time complexity, and token consumption.

## 4.1 Hybrid Topology with Step-Wise Execution

**Coordination of Sequential and Decentralized Topology.** Given a MAS with hybrid topology $G = (N, E, F, P, o)$, the state of $e_{iv}$ is either sequential or decentralized. To leverage strengths, the sequential topology determines a hierarchical relationship in execution, which supports inter-step communication. In contrast, the decentralized topology establishes a synchronous relationship, which becomes the backbone of intra-step aggregation. The above description is formalized as Algorithm 1:

**Structural Constraints over Hybrid Topology.** Given the inherent sequential order of Algorithm 1, we can conceptualize the architecture of a hybrid topology as a process in which edges are incrementally added to a fixed structure. To ensure the validity of $G$ (i.e., no cycle in the phase 3), the addition of a sequential edge $e^s_{iv}$ or a decentralized edge $e^d_{iv}$ should follow:

**Theorem 1.** *The validity of adding a sequential connection $e^s_{iv}$ into a hybrid graph $G$ is contingent upon two conditions: (1) if the destination $n_v$ is not in any decentralized components, the updated sequential topology does not introduce any cycles; (2) if the destination $n_v$ is in a decentralized connected component $G^d_{sub}$, virtual sequential connections (not operational in real communication) should be added from the source to other members in component $G^d_{sub}$, i.e., $G = G \cup \{e^s_{vir,ij} | n_j \in G^d_{sub}, n_j \neq n_v\}$, then the updated sequential topology does not introduce any cycles.*

*Proof.* (1) if the destination $n_v$ is not in any decentralized component, the update leads to $I(n_i) = I(n_v) - 1$. When a sequential topology cycle $c = [n_i, n_v, ..., n_i]$ exists, the execution index satisfies $I(n_i) = I(n_i) + |c| - 1$, which leads to a conflict; (2) if the destination $n_v$ is in a decentralized component $G^d_{sub}$, where $\forall n_j \in G^d_{sub}, I(n_v) = I(n_j)$. The update results in $I(n_i) = I(n_v) - 1$, thus $\forall (n_j) \in G^d_{sub}, I(n_i) = I(n_j) - 1$. From the topological perspective, equivalent virtual connections from the source to other members are added, and then cycles are checked.

**Algorithm 1** MAS execution with an underlying hybrid topology

**Require:** System $G$, UnionFind $uf$ that stores decentralized connected components, input $x$;
**Ensure:** Final answer $z_o$;

**// Phase 1: Distinguishing steps;**
initialize a node2step map { };
**for** $n$ in $N$ **do**
  **if** $uf$.contain($n$) == False **then**
    node2step[$n$] = $n$;
  **else**
    $G^d_{sub}$ = $uf$.get_subgraph($n$) and node2step[$n$] = $G^d_{sub}$;
  **end if**
**end for**
**// Phase 2: Recording sequential topologies in the step level;**
initialize a sequential set of edges among steps $E^s_{step} = [\,]$;
**for** $e^s_{iv}$ in set of sequential edges $E^s$ **do**
  $E^s_{step}$.add((node2step[$i$], node2step[$v$]));
**end for**
**// Phase 3: Executing steps according to the topological order;**
**for** step in TopologicalSort($E^s_{step}$) **do**
  **if** step without an internal decentralized topology **then**
    executing Eq.1 according to the mutual node-level sequential topology;
  **else**
    generating self-messages as $m_i = f_i(x, \{z_v | v \in pre(i)\}, p_i)$ for all members;
    executing Eq.2 according to internal node-level decentralized topology;
  **end if**
**end for**
**return** $z_o$

**Theorem 2.** *The validity of adding a decentralized connections into a hybrid graph $G$ is contingent upon the following condition: Let the decentralized component containing the source node be denoted as $G^d_{sub,i}$ and the decentralized component containing the destination node be denoted as $G^d_{sub,v}$. Virtual sequential connections are added from all predecessors of $G^d_{sub,i}$ to $G^d_{sub,v}$ and from all predecessors of $G^d_{sub,v}$ to $G^d_{sub,i}$, then the updated sequential topology does not introduce any cycles.*

Proof. The addition of a decentralized topology $e^d_{iv}$ leads to a combination of two steps, thus $I(G^d_{sub,i}) = I(G^d_{sub,v})$. In this way, their predecessors are shared in the sequential topology, i.e., $\forall n_k, I(n_k) = I(G^d_{sub,i}) - 1 \leftrightarrow I(n_k) = I(G^d_{sub,v}) - 1$, which are represented as additional virtual sequential connections. Then the updated sequential topology does not introduce any cycles.

In addition, we formalize Theorem 1 and Theorem 2 into pseudo-codes in Appendix D.

### 4.2 Topology Planning with LLM-Initialized Optimization

**Parameterization of Distribution.** To employ gradient-based learning techniques to architect the desired topology, the underlying idea is to define a parameterized probabilistic distribution over potential topologies as $E \sim D_\theta$ [7, 22]. Technologically, for our hybrid topology, a learnable vector $p_\theta \in R^3$ is assigned to $e_{iv}$ and is normalized to $p'_{iv}$ as the possibility of [no, sequential, decentralized] via Gumbel-Softmax [27]. Therefore, the approximated joint possibility of $D_\theta$ is given by:

$$E \sim D_\theta = \prod_{i \neq v} \begin{cases} p'_{iv}[j], \text{if the } j\text{-th state is sampled and allowed,} \\ 0, \text{otherwise.} \end{cases} \tag{3}$$

To avoid invalid sampling (i.e., a possibility of 0), we utilize the iterative sampling strategy [7] following the guidance of Theorem 1 and Theorem 2. Firstly, $E$ is initialized as $\{\}$, then potential $e_{iv}$ is sampled as one of three states. If the addition of $e_{iv}$ violates topological constraints, it will not be included. Otherwise, add $e_{iv}$ to $E$ with the corresponding possibility $p'_{iv}[j]$.

**Optimization Strategy.** After the parameterization, we aim to minimize the expectation of sampled graphs w.r.t. pre-defined utility loss $\mathcal{L}_\tau$ via the gradient-based optimization:

$$\text{argmin}_\theta \, \mathbb{E}(\mathcal{L}_\tau(G)) \; s.t. \; G \sim (N, D_\theta, F, P, I, o). \tag{4}$$

To estimate Eq.4, mini-batches of hybrid topologies are sampled from the distribution, i.e., $\{G_1, G_2, \ldots, G_m\} \sim D_\theta$. Eventually, the utility loss of a batch is calculated in the log space as:

$$\mathcal{L}_\tau(G_\theta) \log(P(G_\theta)) \approx \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_\tau(G_i) \log(p_\theta(E_i)). \tag{5}$$

**LLM as an Initializer.** In some previous implementations [7, 22], probability of each connection state is initialized uniformly (e.g., $[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ for a hybrid connection). But given $O(2^{|E|})$ complexity of the search space, a reliable initialization of the distribution is crucial to enhance the stability of subsequent optimization. Therefore, before learning, we leverage the reasoning capabilities of LLMs to investigate the potential candidates with planning constrained by Theorem 1 and Theorem 2:

$$f_{LLM}(N, o, p_{init}) \rightarrow \{G_1, G_1, ..., G_k\}, \text{ where } p_{init} \text{ is the prompt for initializer.} \tag{6}$$

Subsequently, we evaluate these candidates and aggregate them based on performance:

$$\hat{E} = \sum_{j=1}^{k} w_j E_j, \text{where } w = \text{softmax}(u) \text{ and } u_j = \frac{\mathcal{L}_\tau(G_j) - min(\mathcal{L}_\tau)}{max(\mathcal{L}_\tau) - min(\mathcal{L}_\tau)}. \tag{7}$$

Eventually, the pre-training objective is to minimize the discrepancy between the $E_\theta$ and $\hat{E}$:

$$\text{argmin}_\theta \, \mathcal{L}_{init} = \sum_{i \neq v} ||p'_{iv} - \hat{e}_{iv}||_2^2. \tag{8}$$

The detailed design of the LLM-driven initializer can be found in Appendix C.

### 4.3 Topological Normalization with Efficiency Improvement

**Topological Density Normalization for Token Efficiency.** As shown in Figure 3, there is a positive correlation between the number of edges and the token cost of a system. Therefore, we employ the penalty for possibilities of connections, which is formulated as a specific $l_1$ normalization:

$$\mathcal{L}_d(G_\theta) = \sum_{i \neq v} (p'_{iv}[1] + p'_{iv}[2]). \tag{9}$$

Notably, Eq.9 is directly implemented over the distribution, regardless of the batch sampling. Consequently, $l_1$ normalization encourages the sparsity of an optimized topology.

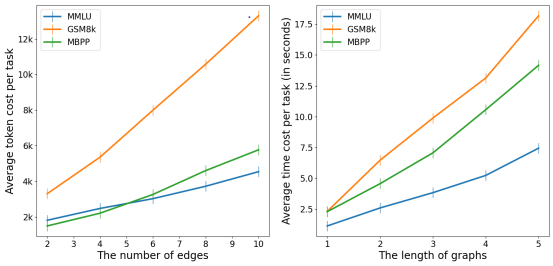



Figure 3: (Left) Relationship between token cost and the number of edges. (Right) Relationship between time cost and the length of agentic graphs in a 6-agent system with hybrid topology.

**Topological Length Normalization for Time Efficiency.** Under asynchronous environment, the execution time has a positive correlation with the longest path of $G$ at the step level, where a sequential connection is assigned a weight of 1, and a source step with two-stage decentralized communication leads to 1 additional weight. Thus, the topological length normalization $\mathcal{L}_{len}$ over $D_\theta$ is estimated in the log space as:

$$\mathcal{L}_{len}(G_\theta) \log(P(G_\theta)) \approx \frac{1}{m} \sum_{i=1}^{m} \frac{\text{LongestPath}(G_i)}{|N| - 1} \log(p_\theta(E_i)), \tag{10}$$

where $\frac{1}{|N|-1}$ is used for normalization on the system scale, given the length ranges in [0, $|N| - 1$]. Eventually, the completed objective of our optimization is denoted as:

$$\mathcal{L}_{total} = \mathcal{L}_\tau + \lambda_1 \mathcal{L}_d + \lambda_2 \mathcal{L}_{len}, \tag{11}$$

where $\lambda_1$ and $\lambda_2$ are weights for two normalizations. After training, a desired topology can be iteratively sampled from the optimized distribution for downstream tasks.

Table 1: Performance comparison (in %) among sequential topologies (S), decentralized topologies (D) and hybrid topologies (H). We vary the number of agents in [5, 7, 9]. The "w" refers to "with LLM-driven initialization" and "w/o" refers to without LLM-driven initialization.

| System | Strategy | MMLU | | | AQuA | | | MBPP | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 5 | 7 | 9 | 5 | 7 | 9 | 5 | 7 | 9 |
| S | w | 75.9↑ | 77.0↑ | 77.6↑ | 78.5↑ | 79.2↑ | 79.0↑ | 75.2↓ | 76.7↑ | 77.6↑ |
| | w/o | 75.2 | 76.8 | 76.0 | 78.0 | 77.7 | 77.5 | 75.5 | 75.9 | 76.7 |
| D | w | 77.6↑ | 77.5↓ | 78.4↑ | 80.4↑ | 79.5↑ | 80.1↑ | 74.2↑ | 76.0↑ | 75.5↓ |
| | w/o | 76.6 | 78.2 | 78.0 | 78.8 | 79.0 | 78.3 | 73.5 | 74.3 | 75.8 |
| H | w | 77.2↑ | 79.1↑ | 79.3↑ | 81.3↑ | 81.6↑ | 82.0↑ | 77.5↑ | 78.9↑ | 80.0↑ |
| | w/o | 76.0 | 77.5 | 77.1 | 79.3 | 80.2 | 79.6 | 75.7 | 77.4 | 78.3 |

# 5 Experimental Results

In this section, we conduct experimental evaluations to address the following research questions: (**RQ1**) Could the introduction of LLM-driven initializer improve the reliability of topology optimization? (**RQ2**) Could our hybrid topology effectively integrate advantages from various topologies? (**RQ3**) Does our topological density constraint further improve token efficiency? (**RQ4**) Does our topological length constraint further improve time efficiency?

## 5.1 Experimental Settings

**Datasets.** (1) short-term reasoning scenarios, including a general QA benchmark MMLU [28], mathematical reasoning benchmarks GSM8k [29] and AQuA [30] ; (2) long-term planning scenarios, including code planning and generation tasks, HumanEval [31], MBPP [32], and an asynchronous graph planning task, AsyncHow [33]. The details of datasets are presented in Appendix A.

**MASs**: In the main paper, the LLM backbone is GPT-4o-mini [34]. The achievement of MASs w.r.t. role definitions, tool callings, memory building is consistent with previous studies [7, 22]. Systems built on three topologies will benefit from asynchronous parallelization. All evaluations will be repeated five times.

**Baselines.** For the sequential topology, we select handcrafted topologies: chain, tree, star, random graphs, and optimized topologies from recent influential works: GPTSwarm [7], GDesigner [25], MaAS [18], AgentPrune [11]. From the decentralized topology, we choose handcrafted topologies: chain, tree, star, cycle, full connection, random graphs, and optimized topologies from state-of-the-art (SOTA) work, MPAS [22]. Notably, GPTSwarm and MPAS serve as the basic distribution optimizers for sequential topologies and decentralized topologies, respectively. The details of implementation could be found in Appendix B.

## 5.2 Evaluation of LLM-Driven Initializer (RQ1)

To illustrate the performance improvement originating from the LLM-driven initialization, we conduct an ablation study on this module. The number of LLM-generated topologies is 8 and each is evaluated over 64 tasks. From the data shown in Table 1, the following insights are confirmed:

**In 24 / 27 evaluations, the introduction of LLM-driven initialization improves the task performance of optimized systems.** This improvement arises from the fact that topological optimization is inherently a parameterized exploration, where reasonable pre-training suppresses misleading search directions and accelerates the gradient-based reinforcement learning.

**Among three topologies, hybrid topology relies most heavily on the warm up of LLM-generated searches.** Compared with single-form topologies, the distribution of hybrid topology is more complex. Consequently, during the optimization, more feasible directions will be sampled and evaluated. In this way, our initialization further reduces instability in the early stages of topology learning.

Table 2: Performance (in $\mu \pm \sigma$ %) comparison in 5-agent systems.

| System | Structure | Single-step task | | | Multi-step task | | |
|---|---|---|---|---|---|---|---|
| | | MMLU | GSM8k | AQuA | HumanEval | MBPP | AsyncHow |
| S | Chain | 73.4±1.2 | 93.0±1.2 | 77.4±1.5 | 84.4±1.7 | 73.6±0.7 | 59.0±2.4 |
| | Tree | 73.5±2.4 | 92.6±1.8 | 75.8±2.3 | 82.5±1.5 | 72.3±0.5 | 56.4±1.8 |
| | Star | 73.8±2.7 | 93.4±2.0 | 75.4±1.9 | 83.6±1.9 | 72.8±1.1 | 54.2±1.4 |
| | Random | 73.3±3.9 | 91.9±1.8 | 76.3±1.7 | 82.5±2.5 | 71.8±1.4 | 57.3±3.6 |
| | Swarm | 75.9±1.4 | 95.2±1.1 | 78.5±1.1 | 86.5±1.0 | 75.2±1.0 | 63.3±1.2 |
| | GDesigner | 76.2±2.0 | 94.6±1.6 | 79.2±1.3 | 86.0±0.8 | 75.0±1.5 | 64.0±1.4 |
| | MaAS | 75.4±2.2 | 94.1±2.2 | 77.4±1.4 | 86.4±0.8 | 75.9±0.8 | 62.1±1.9 |
| | AgentPrune | 76.0±1.3 | 94.8±2.0 | 78.3±1.2 | 85.6±1.5 | 74.8±1.0 | 63.6±1.7 |
| D | Chain | 73.6±2.0 | 93.6±2.2 | 75.9±2.2 | 82.4±1.4 | 69.2±0.6 | 50.7±1.6 |
| | Tree | 74.1±1.8 | 93.7±1.0 | 76.4±1.6 | 83.5±1.1 | 69.3±1.0 | 52.3±2.7 |
| | Star | 74.7±1.6 | 94.6±1.6 | 77.3±1.3 | 82.1±0.6 | 72.3±0.8 | 53.5±2.0 |
| | Cycle | 75.6±0.4 | 95.0±0.4 | 78.5±1.1 | 84.2±0.7 | 73.8±1.2 | 54.6±0.8 |
| | Complete | 76.0±1.2 | 93.6±1.8 | 77.8±1.6 | 83.0±0.7 | 69.8±0.5 | 51.2±1.0 |
| | Random | 74.9±2.3 | 93.2±1.9 | 77.9±2.4 | 83.1±0.9 | 72.5±1.4 | 54.9±2.5 |
| | MPAS | **77.6±1.5** | 95.5±1.1 | 80.4±0.9 | 85.7±1.6 | 74.2±1.8 | 56.2±2.1 |
| H | Random | 73.8±1.8 | 92.3±2.3 | 76.2±1.8 | 82.8±3.3 | 72.0±2.5 | 57.8±4.6 |
| | HTAS | 77.0±1.5 | **96.2±0.8** | **81.3±1.4** | **87.6±2.1** | **77.5±0.7** | **68.4±1.6** |

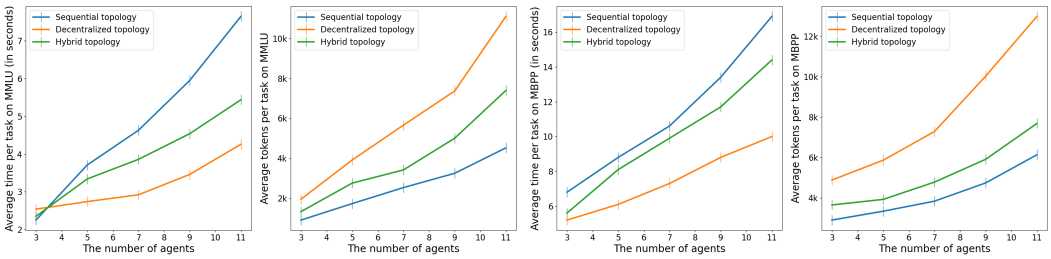



Figure 4: Comparisons of time consumption (in seconds) and token consumption (in tokens) among three topologies after the distribution optimization on MMLU (left) and MBPP (right).

### 5.3 Evaluation of Various Topologies (RQ2)

To demonstrate a better adaptation of our scheme, we compare the optimized hybrid graphs with single-form topologies. In this section, $\lambda_1$ and $\lambda_2$ are set to 0 because system performance is the primary goal in practice. For fairness, the LLM-driven initializer is utilized for all distributions. Experimental results w.r.t. performance, time consumption, and token cost are shown in Table 2 and Figure 4, respectively. We observe that:

**The hybrid topology has better adaptation across various scenarios with comprehensive algorithms.** Specifically, the sequential topology and the decentralized topology have their own preferred domains: the former stimulates multi-step planning, while the latter boosts the aggregation of ideas within a step. In contrast, optimized hybrid topologies achieve the best performance in 5/6 scenarios with 0.7%∼4.4%. In MMLU, we reckon that the reason why decentralized topology outperforms others is that daily QA requires the information sharing of various perspectives rather than reasoning. For instance, as presented in Figure 5, agents tend to adopt mutual decentralized topologies.

**The hybrid topology balances time consumption and token consumption.** Generally, the step-by-step execution of sequential topologies consumes more time, while graph-level communication of decentralized topologies requires more mutual communications. As a result, our scheme offers a compromise solution with reasonable costs. Moreover, for specific tasks, the consumption of a hybrid topology depends primarily on the dominant topology, such as the sequential topology in MBPP.

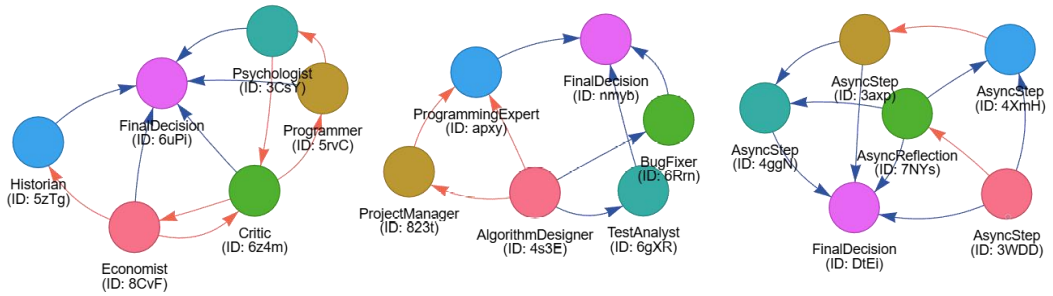



Figure 5: The optimized topologies on MMLU, MBPP and AsyncHow (from left to right). Blue arrows represent sequential connections, orange arrows represent decentralized connections.

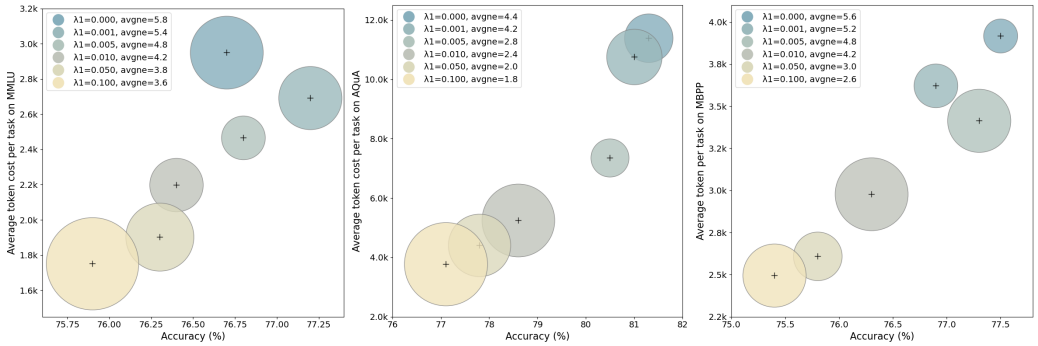



Figure 6: Influence of weight $\lambda_1$ on performance (in %), token consumption (in tokens), the average number of edges (denoted as "avgne") on MMLU, AQuA and MBPP (from left to right).

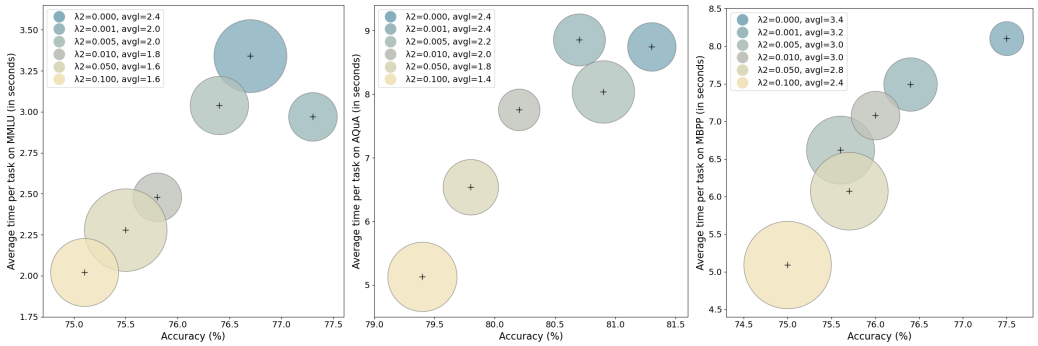



Figure 7: Influence of weight $\lambda_2$ on performance (in %), time consumption (in seconds), the average length of longest path (denoted as "avgl") on MMLU, AQuA and MBPP (from left to right).

**Topology optimization is crucial for coordinating the interaction between the two topologies.** In random sampling, the availability of hybrid topologies is significantly limited. In contrast, as shown in Figure 5, high-performance algorithms are discovered after learning, such as 5 agents engaging in a hierarchical discussion and conveying the historical data to the final decision operation in AsyncHow.

### 5.4 Analysis of Topological Density Normalization (RQ3)

We conduct an evaluation of the token efficiency offered by the topological density constraint. Specifically, we will vary the value of $\lambda_1$ in [0.001, 0.005, 0.01, 0.05, 0.1] compared with $\lambda_1 = 0$ and report task performance and token consumption. From Figure 6, we observe following key points:

Generally, our topological density normalization could **effectively suppress the addition of communication connections.** Across 3 evaluated scenarios, the number of pruned edges ranges from 1 to 3 as the value of $\lambda_1$ increases. Furthermore, in most tasks, the removal of a few redundant edges leads

to only **a marginal decrease in performance while significantly reducing token cost.** For instance, in MMLU, when the number of edges is decreased from 5 to 3, task performance remains virtually unchanged, while token consumption is reduced by nearly 20% to 30%. Therefore, we recommend setting $\lambda_1$ to a small value (e.g., $\lambda_1 = 0.01$) to enhance token efficiency in practice.

### 5.5 Analysis of Topological Length Normalization (RQ4)

We analyze how to improve time efficiency via our topological length constraint. Concretely, we will vary the value of $\lambda_2$ in [0.001, 0.005, 0.01, 0.05, 0.1] with $\lambda_2 = 0$ and report task performance and time consumption. Figure 7 presents the experimental results, and we find:

Generally, our topological length normalization **effectively reduces the length of optimized hybrid graphs.** Specifically, when $\lambda_2 = 0.1$, time consumption is cut by nearly 40%$\sim$50%. However, we observe that **over-constraining the topology length will significantly reduce performance in long-term planning tasks.** We reckon the underlying reason is that incremental communication is essential for multi-round interaction scenarios. Therefore, we recommend that developers start with a small value of $\lambda_2$ (e.g., 0.001) and gradually increase it based on real-world requirements.

## 6 Conclusion

In this study, we reveal that the sequential and decentralized topology serve as complementary frameworks within the MAS domain: the former structures multiple steps into a coherent workflow, while the latter promotes open dialogues without any constraints. Motivated by this observation, we introduce HTAS, a hybrid topology planning scheme that integrates the strengths of both structures for improved utility, reduced token cost and enhanced time efficiency. Extensive experiments across real-world tasks lead to the following conclusions: (1) compared with single-form topologies, hybrid-structured systems demonstrate improved adaptability across both short-term and long-term scenarios, with balanced time complexity and token consumption; (2) a desired hybrid topology involves complex inter-step and intra-step communication, which requires implementation of autonomous learning strategies; (3) the introduction of an LLM-driven initializer mitigates ineffective topology searches, particularly for hybrid structures; and (4) the implementation of appropriate topological normalizations can greatly reduce both time and token consumption. In the future, we plan to explore testing-time training within the hybrid topology to enhance adaptability in dynamic environments.

## 7 Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62402106 and Grant 62676088, in part by the Natural Science Foundation of Jiangsu Province of China under Grant BK20241272, in part by the "Zhishan" Scholars Programs of Southeast University under Grant 2242026RCB0031, and in part by the Start-Up Research Fund of Southeast University under Grant RF1028623129.

## References

[1] M. Seo, J. Baek, S. Lee, and S. J. Hwang, "Paper2code: Automating code generation from scientific papers in machine learning," in *NeurIPS 2025 Fourth Workshop on Deep Learning for Code*.

[2] Z. Zhang, R. Liu, A. Liu, X. Liu, X. Gao, and H. Sun, "Code2bench: Scaling source and rigor for dynamic benchmark construction," in *The Fourteenth International Conference on Learning Representations*.

[3] J. Jiang, K. Zhou, W. X. Zhao, Y. Song, C. Zhu, H. Zhu, and J.-R. Wen, "Kg-agent: An efficient autonomous agent framework for complex reasoning over knowledge graph," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 9505–9523.

[4] X. Wu, J. Yang, L. Chai, G. Zhang, J. Liu, X. Du, D. Liang, D. Shu, X. Cheng, T. Sun *et al.*, "Tablebench: A comprehensive and complex benchmark for table question answering,"

in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 24, 2025, pp. 25 497–25 506.

[5] F. Liu, Y. Zhang, J. Luo, J. Dai, T. Chen, L. Yuan, Z. Yu, Y. Shi, K. Li, C. Zhou *et al.*, "Make agent defeat agent: Automatic detection of taint-style vulnerabilities in llm-based agents," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 3767–3786.

[6] Z. Yu, Z. Guo, Y. Wu, J. Yu, M. Xu, D. Mu, Y. Chen, and X. Xing, "Patchagent: A practical program repair agent mimicking human expertise," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 4381–4400.

[7] M. Zhuge, W. Wang, L. Kirsch, F. Faccio, D. Khizbullin, and J. Schmidhuber, "Gptswarm: language agents as optimizable graphs," in *Proceedings of the 41st International Conference on Machine Learning*, 2024, pp. 62 743–62 767.

[8] J. Zhang, J. Xiang, Z. Yu, F. Teng, X.-H. Chen, J. Chen, M. Zhuge, X. Cheng, S. Hong, J. Wang *et al.*, "Aflow: Automating agentic workflow generation," in *The Thirteenth International Conference on Learning Representations*.

[9] X. Han, C. Zhu, H. Zhu, and X. Zhao, "Swarm intelligence in geo-localization: A multi-agent large vision-language model collaborative framework," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, 2025, pp. 814–825.

[10] K.-T. Tran, D. Dao, M.-D. Nguyen, Q.-V. Pham, B. O'Sullivan, and H. D. Nguyen, "Multi-agent collaboration mechanisms: A survey of llms," *arXiv preprint arXiv:2501.06322*, 2025.

[11] G. Zhang, Y. Yue, Z. Li, S. Yun, G. Wan, K. Wang, D. Cheng, J. X. Yu, and T. Chen, "Cut the crap: An economical communication pipeline for llm-based multi-agent systems," in *The Thirteenth International Conference on Learning Representations*.

[12] S. Wang, G. Zhang, M. Yu, G. Wan, F. Meng, C. Guo, K. Wang, and Y. Wang, "G-safeguard: A topology-guided security lens and treatment on llm-based multi-agent systems," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 7261–7276.

[13] H. Li, X. Liu, C. H. Chun, D. Li, N. Zhang, and C. Xiao, "Drift: Dynamic rule-based defense with injection isolation for securing llm agents," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[14] A. Pei, Z. Yang, S. Zhu, R. Cheng, and J. Jia, "Selfprompt: Autonomously evaluating llm robustness via domain-constrained knowledge guidelines and refined adversarial prompts," in *Proceedings of the 31st International Conference on Computational Linguistics*, 2025, pp. 6840–6854.

[15] S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng, J. Wang, C. Zhang, Z. Wang, S. K. S. Yau, Z. Lin *et al.*, "Metagpt: Meta programming for a multi-agent collaborative framework," in *The Twelfth International Conference on Learning Representations*, 2024.

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

[17] G. Zhang, M. Fu, K. Wang, G. Wan, M. Yu, and S. YAN, "G-memory: Tracing hierarchical memory for multi-agent systems," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[18] G. Zhang, L. Niu, J. Fang, K. Wang, L. Bai, and X. Wang, "Multi-agent architecture search via agentic supernet," in *International Conference on Machine Learning*. PMLR, 2025, pp. 75 834–75 852.

[19] Y. Yue, G. Zhang, B. Liu, G. Wan, K. Wang, D. Cheng, and Y. Qi, "Masrouter: Learning to route llms for multi-agent systems," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 15 549–15 572.

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

[21] J. Wan, T. Yu, K. Jiang, Y. Fu, W. Jiang, and J. Zhu, "Digest the knowledge: Large language models empowered message passing for knowledge graph question answering," in *Proceedings*

*of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 15 426–15 442.

[22] J. Yu, J. Jia, S. Qin, X. Jia, S. Ma, Y. Huang, Y. Yuan, and G. Cheng, "Mpas: Breaking sequential constraints of multi-agent communication topologies via individual-epistemic message propagation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 35, 2026, pp. 29 847–29 855.

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

[24] J. Wu, Q. Zhao, Z. Chen, K. Qin, Y. Zhao, X. Wang, and Y. Yao, "Gap: Graph-based agent planning with parallel tool use and reinforcement learning," in *The First Workshop of NeurIPS on Knowledge Graphs & Agentic Systems Interplay*.

[25] G. Zhang, Y. Yue, X. Sun, G. Wan, M. Yu, J. Fang, K. Wang, T. Chen, and D. Cheng, "G-designer: Architecting multi-agent communication topologies via graph neural networks," in *International Conference on Machine Learning*. PMLR, 2025, pp. 76 678–76 692.

[26] B. Jin, C. Xie, J. Zhang, K. K. Roy, Y. Zhang, Z. Li, R. Li, X. Tang, S. Wang, Y. Meng *et al.*, "Graph chain-of-thought: Augmenting large language models by reasoning on graphs," in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024, pp. 163–184.

[27] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *International Conference on Learning Representations*, 2017.

[28] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," in *International Conference on Learning Representations*, 2021.

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

[30] W. Ling, D. Yogatama, C. Dyer, and P. Blunsom, "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)*, 2017, pp. 158–167.

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

[32] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le *et al.*, "Program synthesis with large language models," *arXiv preprint arXiv:2108.07732*, 2021.

[33] F. Lin, E. La Malfa, V. Hofmann, E. M. Yang, A. Cohn, and J. B. Pierrehumbert, "Graph-enhanced large language models in asynchronous plan reasoning," *arXiv preprint arXiv:2402.02805*, 2024.

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

[35] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," in *International Conference on Learning Representations*, 2023.

[36] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," in *The eleventh international conference on learning representations*, 2022.

[37] A. Liu, B. Feng, B. Xue, B. Wang, B. Wu, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan *et al.*, "Deepseek-v3 technical report," *arXiv preprint arXiv:2412.19437*, 2024.

[38] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican *et al.*, "Gemini: a family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2023.

[39] S. Xue, J. Li, J. Jia, J. Yu, and X. Jia, "Let prompts bridge defense knowledge: Transferable graph purification via vulnerability-aware gpl," *arXiv preprint arXiv:2608.29054*, 2026.

[40] J. Jia, J. Yu, D. Wu, C. Wu, H. Zhu, and L. Wang, "Prompt as a double-edged sword: A dynamic equilibrium gradient-assigned attack against graph prompt learning," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, 2025, pp. 1049–1060.

[41] J. Jia, J. Song, J. Yu, J. Guo, X. Jia, D. Wu, Y. Yuan, and G. Cheng, "Pagpl: Privacy-aware graph prompt learning scheme via adaptive perturbation-estimated topology recovery," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 27, 2026, pp. 22 247–22 255.

[42] J. Jia, R. Li, C. Wu, Y. Feng, S. Ma, L. Wang, and R. H. Deng, "Environment-adaptive representation interaction for privacy-perturbed graphs against deceptive ood attacks," *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 12 429–12 444, 2025.

# A LLM-Driven Agentic Datasets

The descriptions and statistics of the benchmark datasets are presented in Table 3.

Table 3: Dataset descriptions and statistics.

| Scenario | Dataset | Task Type | Metric | Questions |
|---|---|---|---|---|
| General reasoning | MMLU [28] | 4-choice general question answering | Accuracy | 1816 |
| Math reasoning | GSM8K [29] | Mathematical problem solving | Accuracy | 1319 |
| | AQuA [30] | 5-choice mathematical problem answering | Accuracy | 254 |
| Code reasoning | HumanEval [31] | Python code generation | Passing rate | 161 |
| | MBPP [32] | Python code generation | Passing rate | 974 |
| Graph planning | AsyncHow [33] | Shortest path generation | Accuracy | 1598 |

# B Implementation Details

For a fair comparison among three topologies, the following experimental settings remain consistent throughout evaluations.

**Hyperparameters of LLMs.** The temperature is set to 0.0, the top_p is set to 1.0, the frequency_penalty is set to 0.0, the presence_penalty is set to 0.0, and the max_tokens is set to 8192. The implementation of LLMs is achieved by calling official APIs with stability.

**Train Test Split.** MMLU [28] is officially divided into a train set and a test set. For other datasets, they will be split into train sets with 60% samples and test sets with the remaining 40% samples.

**Optimized Topologies.** The temperature of Gumbel-Softmax is set to 1.0, The learning rate is set to 0.1 with Adam as the optimizer, the batch size is set to 16, the number of learning epochs is set to 100. Within a batch, a task will be executed by a system, whose topology is sampled from the distribution.

**Task-Solving Agents.** For MMLU, GSM8k and AQuA, we refer to the implementations of MPAS [22]. For HumanEval and MBPP, agents are designed to support the long-term loop of code generation, testing and feedback. For Asynchow, we refer to the provided agent designs of benchmarks but adapt them to multi-agent communication. The specific implementations of agents can be found in our open code.

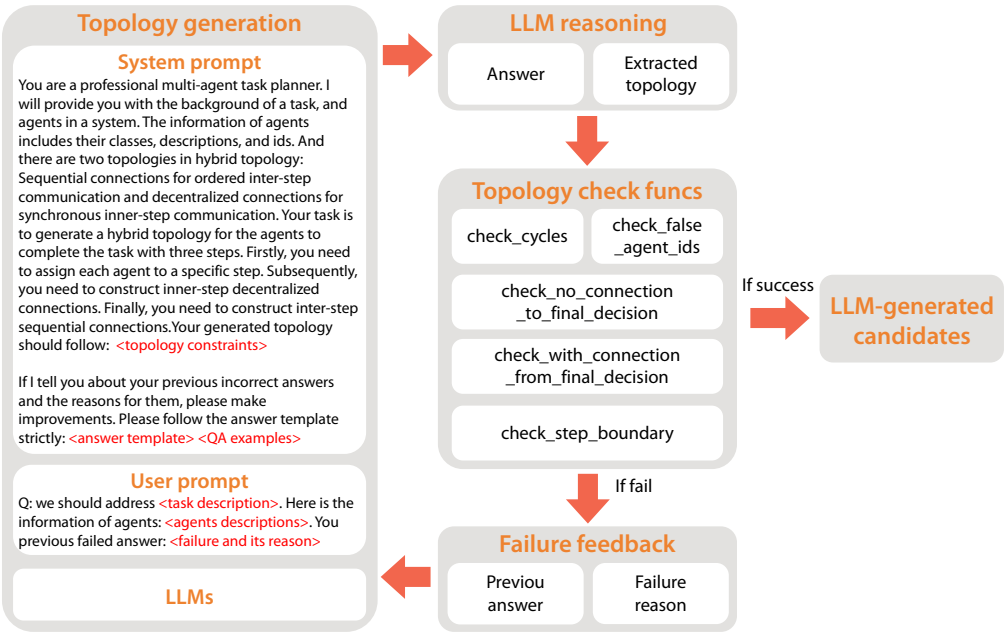



Figure 8: The design of our LLM-driven initializer.

**Final Decision Agent.** In MMLU, we utilize Self-Consistency [35] as the final decision operation, which aligns with GPTSwarm [7]. In other scenarios, we follow GDesigner [25] to employ senior decision makers to summarize operational information and answer queries.

**Communication Context.** We insert the outputs of predecessors into the user prompt as follows:

```
Q: < task description>.
Answer from other agents:
(1) <agent_role, agent_id, agent_output>
(2) <agent_role, agent_id, agent_output>
...
Your previous thought: <self-message>
```

**Asynchronism.** As stated in the main paper, all systems will benefit from asynchronous parallelization with sufficient APIs. For example, given a sequential topology with [a->b, a->c], nodes with the same topological index (i.e., b and c) will be executed asynchronously.

**Evaluation.** The evaluation is repeated five times across experiments to reduce variability w.r.t. LLM answers, time costs, token costs. The experimental results are reported as the average value, and in a few cases, the standard deviation is also included.

## C Design of LLM-Driven Initializer

We take the agent design for the initialization of hybrid distribution as an example because LLM-driven hybrid topology generation is much more complex than single sequential topology or decentralized topology. Generally, our proposed agent follows React [36]-style for topology initialization. As shown in Figure 8, firstly, an LLM maps (role description, topology generation requirement, answer examples, system description, failure feedback) to a textual topology. Subsequently, the topology is evaluated by various topology constraints formed as external tools. If the topology is not qualified, tools will return feedback to the LLM for the next turn. Otherwise, the qualified topology is sent to the buffer of LLM-driven candidates.

# D Algorithms

## D.1 Check for Addition of a Sequential Connection

**Algorithm 2** Check for addition of a sequential connection

**Require:** Current hybrid topology $G = (N, E, o)$, a UnionFind $uf$ that stores decentralized connected components, a sequential connection $e^s_{iv}$ to be added;
**Ensure:** CycleFlag and updated topology $G'$;

$G' =$ deepcopy$(G)$;
$G^d_{sub}$=$uf$.find_subgraph($n_v$);
**if** $G^d_{sub}$=={} **then**
    $G' = G' \cup e^s_{iv}$;
**else**
    Initialize a set of virtual sequential edges $E^s_{vir} = \{\ \}$;
    **for** $n_j$ in $G^d_{sub}$ **do**
        **if** $n_j$!=$n_v$ **then**
            $E^s_{vir}$=$E^s_{vir} \cup e^s_{vir,ij}$;
        **end if**
    **end for**
    $G' = G' \cup e^s_{iv} \cup E^s_{vir}$;
**end if**
**return** CheckCycle($G'$), $G'$

## D.2 Check for Addition of a Decentralized connection

**Algorithm 3** Check for addition of decentralized connection

**Require:** Current hybrid topology $G = (N, E, o)$, a UnionFind $uf$ that stores decentralized connected components, decentralized connection $e^d_{iv}$ to be added;
**Ensure:** CycleFlag and updated topology $G'$;

$G' =$ deepcopy$(G)$;
$G^d_{sub,i}$=$uf$.find_subgraph($n_i$), $G^d_{sub,v}$=$uf$.find_subgraph($n_v$);
Initialize a set of virtual sequential edges $E^s_{vir} = \{\ \}$;
**for** $n_k$ in $G^d_{sub,i}$.predecessors **do**
    **for** $n_j$ in $G^d_{sub,v}$ **do**
        $E^s_{vir} = E^s_{vir} \cup e^s_{vir,kj}$;
    **end for**
**end for**
**for** $n_k$ in $G^d_{sub,v}$.predecessors **do**
    **for** $n_j$ in $G^d_{sub,i}$ **do**
        $E^s_{vir} = E^s_{vir} \cup e^s_{vir,kj}$;
    **end for**
**end for**
$G' = G' \cup E^s_{vir} \cup e^d_{iv}$;
**return** CheckCycle($G'$), $G'$

### D.3 Optimization over Distribution of a Hybrid Topology

**Algorithm 4** Optimization over a hybrid topology

**Require:** Potential connections $E_p$ with corresponding learnable distribution matrix $E_\theta$, batch size $b$, training epoch $t$;
**Ensure:** Optimized distribution $E_\theta$;

**for** $i$ in 1,...,$t$ **do**
  batch = []
  **for** $k$ in 1,...,$b$ **do**
    log_possibilities = [];
    edge_possibilities = [];
    $G = (N, \{\}, F, P, I, o)$;
    $uf$ = UnionFind();
    **for** $(n_i, n_v, p_\theta)$ in $E_p$ **do**
      $p'_\theta$ = Gumbel-Softmax$(p_\theta)$;
      $j$ = Argmax$(p'_\theta)$;
      **if** $j$==0 **then** // no connection
        $G' = G$;
      **else if** $j$==1 **then** // a sequential connection
        CycleFlag, $G'$ = Algorithm 2$(G, uf, e^s_{iv})$;
        **if** CycleFlag==False **then**
          continue;
        **end if**
      **else** // a decentralized connection
        CycleFlag, $G'$ = Algorithm 3$(G, uf, e^d_{iv})$;
        **if** CycleFlag==False **then**
          continue;
        **end if**
        $uf$.merge$(n_i, n_v)$; // only for two decentralized connected components
      **end if**
      log_possibilities.append$(log(p'_\theta[j]))$;
      edge_possibilities.append$(1 - p'_\theta[0])$;
      $G = G'$;
    **end for**
    length = LongestPath($G'$);
    batch.append([$G'$, log_possibilities, edge_possibilities, length]);
  **end for**
  Calculate loss according to Eq.11 and update the parameterized distribution;
**end for**
**return** $E_\theta$

## E Optimization on Single-Form Topologies

In this section, we introduce the vanilla optimization strategies for the sequential distribution and the decentralized distribution.

**Optimization on Sequential Topology.** The strategy is first proposed by GPTSwarm [7]. A learnable weight $\theta_{iv}$ is assigned to an edge, then normalized to $\theta'_{iv} \in [0, 1]$ as the existence possibility. In this way, a parameterized probabilistic distribution over DAGs with fixed nodes $N$ and required edges $E$ is defined as:

$$D_\theta = \prod_{i \neq v} \begin{cases} \theta'_{iv}, \text{ if } E \cup e_{iv} \text{ is a DAG,} \\ \text{1-}\theta'_{iv}, \text{ if } e_{iv} \text{ is not sampled,} \\ 0, \text{ otherwise.} \end{cases} \tag{12}$$

A sampling method that realizes this distribution is first to initialize a graph $G \leftarrow (N, E)$. Then, iteratively sample whether to include edge $e_{iv}$ in $G$ for all edges. If including $e_{iv}$ causes a cycle in

Table 4: Performance (in $\mu \pm \sigma$ %) comparison among sequential topologies, decentralized topologies and hybrid topologies in 5-agent systems driven by Deepseek-V3.

| System | Structure | Single-step task | | | Multi-step task | | |
|---|---|---|---|---|---|---|---|
| | | MMLU | GSM8k | AQuA | HumanEval | MBPP | AsyncHow |
| S | Chain | 75.5±0.4 | 93.1±1.0 | 84.1±1.3 | 69.5±1.6 | 52.5±1.2 | 85.4±2.3 |
| | Tree | 77.1±2.4 | 92.5±2.1 | 82.8±1.3 | 66.3±0.8 | 53.5±1.2 | 84.8±2.0 |
| | Star | 76.2±0.7 | 93.0±1.7 | 84.8±1.3 | 68.2±1.1 | 55.7±1.3 | 79.4±1.4 |
| | Random | 75.4±3.1 | 92.1±2.6 | 84.1±2.4 | 67.0±1.7 | 59.8±1.8 | 80.4±2.8 |
| | Swarm | 80.5±1.7 | 95.8±2.0 | 85.7±1.1 | 75.0±2.0 | 66.1±1.3 | 87.4±1.2 |
| | GDesigner | 79.7±1.3 | 95.4±1.6 | 87.4±0.8 | 74.5±1.6 | 65.7±1.6 | 88.2±1.5 |
| | MaAS | 81.4±2.5 | 94.8±2.2 | 86.8±0.9 | 73.7±0.9 | 62.5±1.5 | 87.6±1.5 |
| | AgentPrune | 78.5±1.8 | 94.5±1.3 | 86.5±1.2 | 74.0±1.0 | 66.5±2.1 | 86.3±2.2 |
| D | Chain | 82.9±1.4 | 94.9±0.5 | 87.9±0.8 | 60.6±1.8 | 58.3±1.3 | 78.8±1.0 |
| | Tree | 82.6±1.7 | 94.8±1.9 | 86.8±1.2 | 58.6±2.3 | 58.9±1.7 | 76.3±1.3 |
| | Star | 80.0±1.1 | 94.6±2.4 | 88.0±2.0 | 60.0±1.6 | 56.6±1.4 | 79.1±1.9 |
| | Cycle | 82.9±2.0 | 95.0±1.1 | 87.6±1.0 | 65.0±2.6 | 60.6±1.1 | 83.0±1.7 |
| | Complete | 83.1±2.6 | 94.2±2.0 | 86.6±1.3 | 64.0±3.0 | 50.2±1.4 | 80.2±0.8 |
| | Random | 82.1±3.5 | 93.4±2.7 | 86.4±2.6 | 62.5±4.4 | 59.3±1.4 | 77.8±2.6 |
| | MPAS | **84.9±1.6** | 96.8±1.6 | 88.7±1.5 | 67.5±0.9 | 62.9±1.9 | 83.0±0.8 |
| H | Random | 78.0±2.8 | 94.0±2.0 | 86.0±2.3 | 69.0±3.7 | 60.7±3.0 | 83.7±4.1 |
| | HTAS | 84.0±2.1 | **97.4±1.0** | **90.2±1.6** | **76.0±1.8** | **68.7±1.6** | **92.1±0.7** |

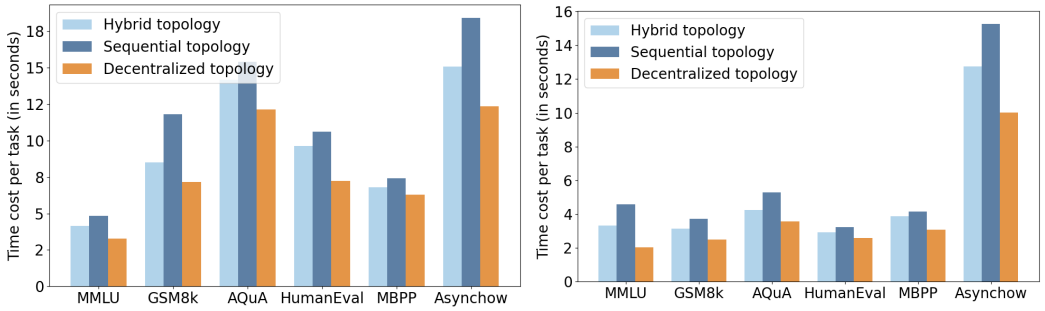



Figure 9: Time cost (in seconds) among three topologies driven by Gemini-3.1-flash driven by Deepseek-V3 (left) and Gemini-3.1-flash (right).

current $G$, then the edge would not be included. Otherwise, add the edge to $G$ with probability $\theta'_{iv}$. After learning, edges will be sampled iteratively as the $\theta'_{iv} > 0.5$ and it will not lead to cycles.

**Optimization on Decentralized Topology.** The strategy is first investigated by MPAS [22]. Similarly, a learnable weight $\theta_{iv}$ is assigned to an edge $e_{iv}$, then normalized to $\theta'_{iv} \in [0, 1]$ as the possibility of its existence. Given that a decentralized system supports any connected structure, the distribution can be directly defined as:

$$D_\theta = \prod_{i \neq v} (\theta'_{iv})^{e_{iv}} (1 - \theta'_{iv})^{1 - e_{iv}}. \tag{13}$$

After learning, all edges with $\theta'_{iv} > 0.5$ are sampled without any constraints.

## F Evaluation on Other LLM Backbones

In the main paper, we evaluate the performance when GPT-4o-mini is utilized as the backbone. In this section, we further conduct evaluations on systems driven by DeepSeek-V3 [37] and Gemini-3.1-flash [38]. Moreover, our LLM-driven initialization is utilized and driven by corresponding LLMs.

Table 5: Performance (in $\mu \pm \sigma$ %) comparison among sequential topologies, decentralized topologies and hybrid topologies in 5-agent systems driven by Gemini-3.1-flash.

| System | Structure | Single-step task | | | Multi-step task | | |
|---|---|---|---|---|---|---|---|
| | | MMLU | GSM8k | AQuA | HumanEval | MBPP | AsyncHow |
| S | Chain | 85.5±1.9 | 93.5±1.4 | 80.2±1.8 | 91.4±0.6 | 87.3±1.9 | 80.7±1.8 |
| | Tree | 85.2±1.4 | 93.3±1.3 | 84.3±1.7 | 90.7±0.8 | 86.2±1.3 | 83.1±0.8 |
| | Star | 86.4±1.6 | 92.4±1.5 | 81.2±1.3 | 89.3±1.2 | 85.4±1.4 | 83.8±1.7 |
| | Random | 84.8±2.8 | 92.0±1.7 | 80.3±1.8 | 90.0±2.0 | 85.8±2.9 | 81.0±2.1 |
| | Swarm | 89.8±1.4 | 96.1±1.8 | 88.9±0.6 | 95.0±0.8 | 91.6±1.0 | 84.4±1.6 |
| | GDesigner | 87.4±1.8 | 96.5±1.3 | 89.4±1.2 | 94.1±0.8 | 92.2±1.2 | 85.6±1.5 |
| | MaAS | 87.1±2.1 | 95.5±1.9 | 88.2±0.8 | 90.7±0.8 | 90.5±1.3 | 86.1±1.6 |
| | AgentPrune | 88.8±1.4 | 95.0±1.2 | 89.2±1.1 | 93.6±1.4 | 91.1±0.5 | 85.0±0.9 |
| D | Chain | 87.7±2.3 | 92.2±1.8 | 86.5±1.7 | 88.6±1.4 | 83.4±1.2 | 77.9±1.7 |
| | Tree | 85.5±1.4 | 92.4±1.4 | 86.7±1.0 | 88.5±1.1 | 85.5±1.2 | 78.8±0.5 |
| | Star | 91.4±1.8 | 93.0±1.8 | 84.3±1.4 | 87.0±0.9 | 83.4±0.6 | 78.0±2.4 |
| | Cycle | 89.5±1.1 | 93.8±1.2 | 88.8±1.5 | 89.5±1.1 | 88.7±1.8 | 76.6±0.7 |
| | Complete | 82.6±1.3 | 90.5±1.9 | 85.6±0.9 | 86.4±1.5 | 77.8±1.6 | 77.4±2.0 |
| | Random | 86.5±3.3 | 92.2±2.9 | 86.8±2.5 | 87.4±3.0 | 82.8±2.5 | 76.2±2.9 |
| | MPAS | 91.5±0.9 | 95.5±0.5 | 90.5±0.6 | 93.6±0.8 | 90.0±1.7 | 79.7±1.6 |
| H | Random | 85.5±3.7 | 93.5±4.1 | 85.8±3.3 | 89.4±2.7 | 85.3±2.7 | 82.9±4.2 |
| | HTAS | **92.3±0.6** | **97.0±1.1** | **92.0±0.8** | **95.5±1.0** | **93.5±0.6** | **89.2±2.0** |

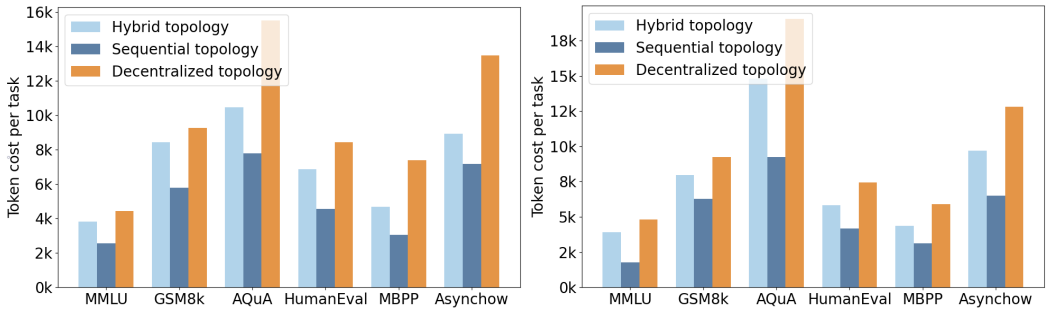



Figure 10: Token cost (in tokens) comparison among three topologies driven by Gemini-3.1-flash driven by Deepseek-V3 (left) and Gemini-3.1-flash (right).

The results of task performance are presented in Table 4 and Table 5. The results of time consumption are shown in Figure 9. The results of token consumption are shown in Figure 10. We observe that:

Our scheme realizes the best utility in 11 / 12 scenarios and has a considerable superiority on MASs driven by Gemini-3.1-flash. Notably, our scheme further improves the performance from 86.1% to 89.2% on AsyncHow. Moreover, our scheme strikes a balance between time efficiency and token efficiency compared with the time consuming sequential topology and the token-laborious decentralized topology. In conclusion, the experimental results obtained with alternative LLM backbones are consistent with those presented in the main paper.

## G Discussion on Large-Scale Topology Optimization

As the number of agents increases [39–42], the $O(n^2)$ complexity of search space increases sharply. We speculate two technological modules could help stabilize the training, i.e., our LLM-driven initialization and density normalization. To demonstrate our assumption, we conduct an experiment with 20 agent in MMLU, AQUA, MBPP. And there are four settings for optimization: optimization only (denoted as OO), optimization with LLM-driven initialization (denoted as OL), optimization with density normalization (denoted as OD), and optimization with initialization and normalization (denoted as OLD). The experimental results are shown in Table 6. We find that density normalization

Table 6: Performance comparison (in $\mu \pm \sigma$ %) among optimization strategies for hybrid topology on MMLU AQUA and MBPP.

| Optimization strategy | MMLU | AQuA | MBPP |
|---|---|---|---|
| OO | 75.8±1.0 | 81.1±2.0 | 67.5±1.2 |
| OL | **77.3±1.4** | **82.1±1.4** | **69.0±1.5** |
| OD | 74.4±2.3 | 79.6±1.7 | 65.0±1.8 |
| OLD | 75.2±1.8 | 80.3±1.0 | 66.2±0.6 |

can lead to a certain degree of decline in task performance, but LLM-driven initialization is crucial for optimization stability.

## NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction reflects our core motivation: integrating advantages of the sequential topology and the decentralized topology via our hybrid structure planning.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The limitations are discussed in the conclusion.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.

- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: For our proposed theorems, we provides the corresponding proofs.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide the detailed experimental settings in Section 5.1 and Appendix B. And our code link is provided.

   Guidelines:

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

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: Our code link is provided in the abstract.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide the detailed experimental settings in Section 5.1 and Appendix B.

   Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: We report the mean and variance of our results across 5 runs.

Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: In the beginning of each experiment, we present the corresponding settings in detail.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Our research follows it.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [N/A]

    Justification: Our scheme aims to improve the adaptation of multi-agent systems. In this way, we mainly discuss positive impacts, but will not bring negative societal impacts.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: We do not directly release models and datasets. We only discuss how to organize multiple LLM-driven agents. It is the user's responsibility to design their agents.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cite the sources of the datasets and the versions of used LLMs.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: We provide the code of our scheme and experiments, which can be found in an anonymized URL.

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

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

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

Answer: [N/A]

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

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

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