HYPER-MACNET: Task-Aware Hypergraphs for Collective Intelligence in Multi-Agent LLMs

Anonymous ACL submission

Abstract

Multi-agent LLM systems perform well on bounded tasks, but they often break down on cross-domain, long-horizon tasks that require many interdependent constraints to be satisfied together. A key difficulty is coordination: isolated pairwise discussions can yield locally correct plans that become inconsistent when merged. To address this, we present HYPER-MACNET, a multi-agent framework that organizes collaboration with task-aware hypergraphs. Instead of using a fixed communication graph, HYPER-MACNET decomposes each problem into a subtask dependency DAG and assigns each subtask to a hyperedge, which explicitly defines the responsible collaboration unit. It further performs mode-aware collaboration, assigns roles within each unit, and records intermediate artifacts on a global blackboard for dependency-consistent propagation and aggregation. On standard benchmarks, HYPER-MACNET achieves a 6.1% relative gain over the strongest baseline and a 37.8% gain over a vanilla baseline. On the CLM complex-task evaluation, it attains the best mean ranking, indicating more globally consistent coordination under coupled constraints.

1 Introduction

Large language models (LLMs) have demonstrated remarkable proficiency in natural language understanding, code generation, and structured reasoning (Brown et al., 2020; Wei et al., 2022; Achiam et al., 2023). These advances have catalyzed the development of multi-agent systems (LLM-MAS), where specialized agents—assigned heterogeneous roles and tools—collaborate to solve complex problems (Wu et al., 2024; Shinn et al., 2023; Park et al., 2023). However, current evaluations of LLM-MAS predominantly focus on well-scoped tasks with stable reasoning trajectories, such as math and programming benchmarks (Cobbe et al., 2021; Hendrycks et al., 2020; Chen, 2021). In contrast, real-world deployments demand cross-domain, multi-stage coordination where interdependent constraints must be satisfied across long horizons and intermediate artifacts are continuously revised. In these settings, unstructured natural language interaction often leads to coordination drift, redundant deliberation, and a breakdown in global consistency (Cemri et al., 2025; Valmeekam et al., 2023; Shumailov et al., 2023).

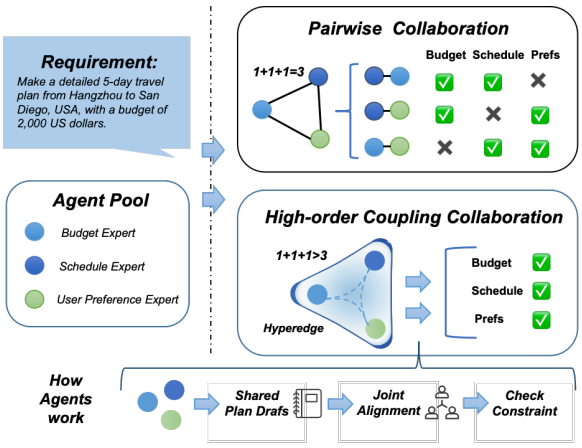

Figure 1: Pairwise exchanges may satisfy constraints only locally and can become inconsistent after post-hoc merging. In contrast, a higher-order collaboration unit co-edits a shared plan draft and performs joint constraint checking over budget, schedule, and preference constraints, producing a feasible travel plan.

We identify a fundamental bottleneck in cross-domain collaboration: *irreducibility* (Figure 1). In many tasks, feasibility depends on jointly aligning heterogeneous constraints under a shared context, and post-hoc merging of independent pairwise threads is therefore often unreliable (Battiston et al., 2021; Abella et al., 2023; Njougouo et al., 2025). This aligns with findings in complex systems that higher-order interactions can exhibit emergent effects beyond pairwise links (Battiston et al., 2020; Benson et al., 2016). Consequently, modeling collaboration purely as a standard graph of binary

edges can obscure the group-level coupling required for coherent reasoning. Operationally, irreducibility manifests as a "divide-and-reconcile" failure mode: constraints may be satisfied in isolated pairwise threads yet become mutually incompatible when merged, making aggregation underdetermined because conflicts surface only under joint constraint checking.

Hypergraphs provide a principled abstraction for such higher-order interactions: a hyperedge connects an arbitrary subset of nodes, naturally representing a discrete collaboration unit (Zhou et al., 2006; Feng et al., 2019; Yadati et al., 2019). While recent work uses hypergraphs mainly as communication backbones to improve message passing efficiency (Zhang et al., 2025), the task structure and collaboration logic are often left implicit—encoded in prompts or ad-hoc scripts. Without an explicit organizational interface, it remains unclear which subproblems require collective reasoning, how they depend on each other, and which interaction protocol each group should follow (Wang et al., 2024; Guo et al., 2024).

To bridge this gap, we present HYPER-MACNET, which uses hypergraphs as an explicit organizational layer—who collaborates on which subtask and in what order—rather than only a communication topology. HYPER-MACNET induces a dependency DAG of interdependent subtasks, and binds each subtask to a hyperedge (a collaboration unit) to preserve higher-order coupling throughout execution. On top of this structure, it selects a collaboration protocol and assigns roles within each team based on subtask semantics, and propagates intermediate artifacts via a global blackboard along the DAG to ensure traceability and global consistency. Notably, HYPER-MACNET is parameter-free and operates at the system level, enabling plug-and-play transfer across LLM backbones without retraining.

We evaluate HYPER-MACNET on standard benchmarks (Cobbe et al., 2021; Hendrycks et al., 2020; Chen, 2021) and a new cross-domain multi-step benchmark, CLM. It achieves a 6.1% relative gain over the strongest baseline and a 37.8% gain over a vanilla baseline on standard tasks, while securing the best mean rank on CLM. Our main contributions are:

- **Organizational hypergraph representation.** We formalize hypergraphs to model higher-order collaboration under irreducible coupling, treating collaboration units (hyperedges) as first-class objects.
- **Structure induction and protocol routing.** We introduce mechanisms to make subtask dependencies and interaction procedures explicit, enabling task-adaptive coordination policies.
- **Empirical evaluation.** We provide comprehensive results on CLM and standard benchmarks, along with system-level analyses characterizing how structural explicitness affects coordination quality.

2 Related Work

We briefly review related research from individual reasoning paradigms to multi-agent collaboration, high-order topology, and collective intelligence.

Individual Reasoning Paradigms. LLM reasoning has evolved from linear CoT to tree/graph structures such as ToT and GoT (Wei et al., 2022; Yao et al., 2023; Besta et al., 2024). However, these paradigms often suffer from structural flattening in multi-agent settings (Park et al., 2023; Li et al., 2023): complex internal reasoning is collapsed into plain-text dialogues, losing topological priors. Autonomous agents such as AutoGPT and AutoGen (Richards, 2023; Wu et al., 2024) implement iterative self-refinement, but still emphasize individual loops rather than structured collective synergy.

Coordination and Scaling. Current LLM-based MAS largely focus on SOP-style workflows or role-based organizations or on dynamic group scheduling and collaboration infrastructures (Hong et al., 2023; Qian et al., 2024a; Li et al., 2023; Chen et al., 2024). Recent work on scaling collaboration with graph-structured topologies (Zhuge et al., 2024; Qian et al., 2024b; Zhou et al., 2025; Ye et al., 2025) reveals performance plateaus in dense dyadic networks. Graph-optimized swarms such as GPTSwarm and MacNet (Zhuge et al., 2024; Qian et al., 2024b) mitigate some routing issues but remain restricted to pairwise links. HYPER-MACNET instead uses hypergraphs to model irreducible high-order couplings that go beyond standard graphs.

Hypergraphs and Collective Intelligence. Hypergraphs offer a mathematical foundation for group interactions beyond pairwise links (Battiston

et al., 2020). Hypergraph-based communication and forecasting frameworks demonstrate advantages for multi-agent coordination (Zhu et al., 2024; Tian et al., 2022). Existing LLM- or agent-based hypergraph approaches (Zhang et al., 2025) typically treat hypergraphs as static communication backbones. In contrast, we treat hypergraphs as an explicit organizational representation coupled with protocol routing, moving from unconstrained semantic interactions such as debate or voting (Du et al., 2023; Chan et al., 2023) toward a structurally grounded paradigm aimed at globally coherent, super-additive collective intelligence (Woolley et al., 2010).

3 Hypergraph Construction and Structural Alignment

To make higher-order collaboration priors explicit and executable in an LLM-based multi-agent system, we model the organizational structure induced by a task as a weighted hypergraph together with a DAG-dependency hyperedge. In the main text, we focus on the minimal structural definitions required by our protocol routing and cross-hyperedge aggregation modules; a detailed treatment is deferred to the appendix B.

3.1 Formal Definition

We define a weighted hypergraph

$$\mathcal{H} = (\mathcal{V}, \mathcal{E}, w),$$

where $\mathcal{V} = \{v_i\}_{i=1}^{N}$ is the node set, $\mathcal{E} = \{e_k\}_{k=1}^{M}$ is the hyperedge set, and each hyperedge $e_k \subseteq \mathcal{V}$ satisfies $|e_k| \geq 2$. The weight function $w : \mathcal{E} \to \mathbb{R}_{>0}$ assigns a positive weight $w_k = w(e_k)$ to each hyperedge, capturing structural priors such as importance, uncertainty, or difficulty. The node–hyperedge incidence relation can be encoded by a binary incidence matrix indicating whether a node participates in a hyperedge.

3.2 Node–Agent Alignment

Let $\mathcal{A} = \{a_i\}_{i=1}^{N}$ denote the set of LLM-based agents. We establish a one-to-one correspondence

$$v_i \longleftrightarrow a_i, \qquad i \in \{1, \ldots, N\},$$

so that $\mathcal{V}$ serves as the structural backbone of the system. All agents share the same base language model, but differ by role prompts and private memory states.

3.3 Hyperedge–Subtask Mapping and Dependency DAG

Let $\mathcal{T}_{\text{sub}} = \{\tau_k\}_{k=1}^{M}$ be a set of structured subtasks. We define a mapping

$$\psi : \mathcal{E} \to \mathcal{T}_{\text{sub}}, \qquad \psi(e_k) = \tau_k,$$

so that each hyperedge simultaneously specifies who collaborates (its incident agents) and what to solve(the associated subtask and interface constraints). To represent precedence constraints among subtasks, we construct a directed acyclic graph (DAG) over hyperedges,

$$\mathcal{G}_E = (\mathcal{E}, \mathcal{F}),$$

where $(e_k, e_\ell) \in \mathcal{F}$ indicates that τ_k must be completed before τ_ℓ. Since $\mathcal{G}_E$ is acyclic, it admits a topological ordering σ, which constrains macro-level scheduling and aggregation. Dependencies can be compiled from logical/information/verification constraints; in structured settings (e.g., region-, module-, or phase-based decompositions), external structural relations may further induce edges in $\mathcal{G}_E$.

Overall, Module 1 in Section 4 realizes the mapping

$$(q, \mathcal{A}) \mapsto \big(\mathcal{H},\, \mathcal{G}_E,\, \psi\big),$$

providing an executable organizational interface for downstream hyperedge-level collaboration and globally coherent aggregation.

4 Hyper-MACNET Architecture

Building on the task-driven hypergraph formulation in Section 3.1 and the dependency DAG in Section 3.3, we introduce HYPER-MACNET, a higher-order collaboration architecture for LLM-based multi-agent systems. HYPER-MACNET treats each hyperedge as an irreducible collaboration unit (a *team*) and uses the hyperedge dependency DAG to schedule execution and aggregate intermediate results into a globally coherent solution.

HYPER-MACNET consists of three coupled modules: (i) task decomposition and hypergraph induction; (ii) expert collaboration on each hyperedge; (iii) cross-hyperedge aggregation and global decision-making. We additionally discuss an optional extension for structural/weight adaptation, which is disabled in the main experiments.

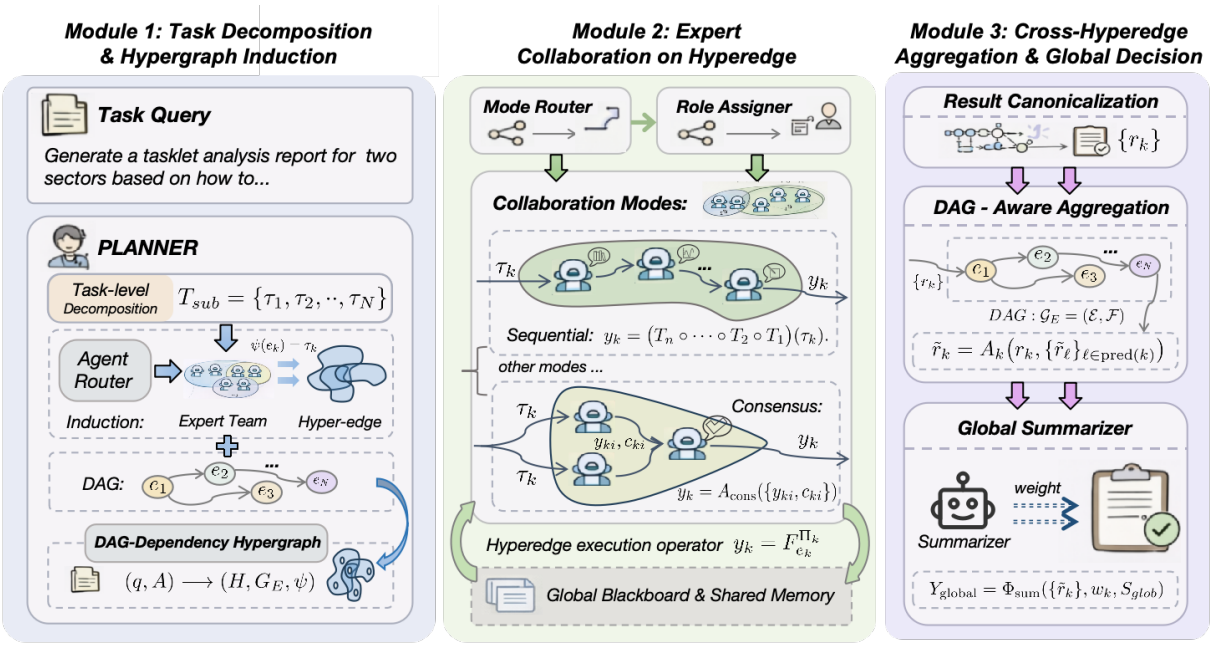

Figure 2: **HYPER-MACNET**, a hypergraph-structured multi-agent collaboration framework. Module 1 decomposes the query into interdependent subtasks and induces a DAG-dependency hypergraph. Module 2 binds each subtask to a hyperedge and executes it with mode routing and intra-unit role assignment. Module 3 canonicalizes and aggregates hyperedge results along the hyperedge-level dependency DAG to yield a traceable and globally consistent solution. Notation is simplified for illustration; see Sec. 4 for formal definitions.

4.1 Module 1: Task Decomposition and Hypergraph Induction

Given an input problem q (optionally with external knowledge $\mathcal{K}$), a planner agent decomposes q into a set of structured subtasks $\mathcal{T}_{\text{sub}} = \{\tau_k\}_{k=1}^{M}$. Each τ_k specifies (i) a concise subproblem description, (ii) required skills/tags for expert routing, and (iii) an expected output interface (e.g., code, structured arguments, statistics, or formatted text).

An expert router then selects a collaborating team $\mathcal{A}_k \subseteq \mathcal{A}$ for each τ_k. Using the node–agent alignment in Section 3.2, each team induces a hyperedge $e_k \subseteq \mathcal{V}$, yielding the hyperedge–subtask mapping $\psi(e_k) = \tau_k$. A positive hyperedge weight w_k is assigned to encode structural priors such as importance, uncertainty, or expected difficulty.

Finally, the planner constructs a DAG-Dependency Hypergraph by coupling the induced hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E}, w)$ with a directed acyclic dependency graph over hyperedges, $\mathcal{G}_E = (\mathcal{E}, \mathcal{F})$, where $(e_\ell, e_k) \in \mathcal{F}$ specifies that the subtask $\tau_\ell = \psi(e_\ell)$ must be completed before $\tau_k = \psi(e_k)$. We enforce $\mathcal{G}_E$ to be acyclic, so that it admits a topological order for schedule and aggregate subtasks in precedence order in Module 3.

4.2 Module 2: Expert Collaboration on Hyperedges

Given the task-driven hypergraph $\mathcal{H}$ and dependency graph $\mathcal{G}_E$, Module 2 specifies how agents collaborate within each hyperedge to solve its associated subtask.

For each hyperedge e_k with team $\mathcal{A}_k$ and subtask τ_k, we instantiate a collaboration protocol Π_k and execute a hyperedge-level operator:

$$y_k, \{s_i^+\}_{a_i \in \mathcal{A}_k} = \mathcal{F}_{e_k}^{\Pi_k}\Big(\{s_i\}_{a_i \in \mathcal{A}_k},\ \tau_k,\ S_{\text{glob}}\Big),$$

where s_i and s_i^+ denote the pre-/post-collaboration local states of agent a_i, S_{glob} is a shared global state, and y_k is a structured intermediate result.

Mode and role routing. Mode routing is formulated as discrete policy selection over a small, predefined set of collaboration modes $\mathcal{M}$ with distinct coordination semantics. We keep the router deterministic and parameter-free for reproducibility:

$$m_k = \pi_{\text{mode}}(\tau_k, e_k, w_k) \in \mathcal{M}.$$

Concretely, π_{mode} selects a mode based on lightweight subtask signals (e.g., whether constraints are tightly coupled or the subtask admits staged refinement): we route to **Consensus** (or

Leader/Decider) when τ_k requires joint reconciliation of coupled constraints, and to **Sequential** when τ_k admits staged refinement; **Brainstorming** is used when broad candidate exploration is beneficial. Roles are assigned by $\pi^{(k)}_{\text{role}} : \mathcal{A}_k \to \mathcal{R}$, and the protocol Π_k is determined by $(m_k, \pi^{(k)}_{\text{role}})$. We use the following canonical modes:

- **Consensus.** Agents produce candidate drafts and exchange critiques focusing on constraint satisfaction and internal consistency. An integrator merges the candidates:

$$y_k = \mathcal{A}_{\text{cons}}\Big(\{y^{(0)}_{k,i}\}_{a_i \in \mathcal{A}_k},\ \{c_{k,i}\}_{a_i \in \mathcal{A}_k}\Big).$$

- **Sequential.** Agents are ordered into a local pipeline, and each stage applies a transformation to the current artifact. The final output is the composed transformation:

$$y_k = \big(\mathcal{T}_r \circ \cdots \circ \mathcal{T}_2 \circ \mathcal{T}_1\big)(\tau_k).$$

- **Brainstorming.** A proposer subset generates diverse candidates in parallel, and integrators distill them into a constraint-aligned solution:

$$y_k = \mathcal{A}_{\text{distill}}\Big(\{y^{(0)}_{k,i}\}_{i \in \mathcal{P}},\ \tau_k\Big), \quad \mathcal{P} \subseteq \mathcal{A}_k.$$

- **Leader/Decider.** Team members contribute evidence or partial results, while a designated decider resolves conflicts and commits to a single output:

$$y_k = \mathcal{D}\Big(\{y^{(0)}_{k,i}\}_{a_i \in \mathcal{A}_k},\ a^{(k)}_{\text{dec}},\ \tau_k\Big).$$

- **Supervisor/Critic.** Supervisor agents audit intermediate artifacts against task-specific criteria and trigger revisions when violations are detected:

$$y_k = \mathcal{A}_{\text{audit}}\big(y_{k,\text{draft}},\ \text{Rules}(\tau_k)\big).$$

Module 2 outputs the set of hyperedge-level intermediate results $\mathcal{Y} = \{y_k\}_{k=1}^{M}$, together with updated local/global states.

4.3 Module 3: Cross-Hyperedge Aggregation and Global Decision

Module 3 integrates hyperedge-level results into a final solution under the dependency constraints encoded by $\mathcal{G}_E$. The key requirement is dependency-consistent aggregation: each subtask aggregates information from its predecessors.

Canonical result packaging. Each hyperedge output y_k is normalized into a canonical record (a structured text summary with fixed fields, e.g., claims, satisfied/violated constraints, evidence, open issues):

$$r_k = \Phi_{\text{pack}}(y_k, \tau_k, S_{\text{glob}}). \tag{1}$$

Predecessor-consistent aggregation on the DAG. For hyperedge e_k, define its predecessor set

$$\text{pred}(k) = \{\ell \mid (e_\ell, e_k) \in \mathcal{F}\}. \tag{2}$$

We compute dependency-consistent records $\tilde{r}_k$ in a topological order of $\mathcal{G}_E$:

$$\tilde{r}_k = \mathcal{A}_k\Big(r_k,\ \{\tilde{r}_\ell \mid \ell \in \text{pred}(k)\},\ w_k\Big). \tag{3}$$

This recursion aligns information flow with the precedence constraints in $\mathcal{G}_E$, preventing future-information leakage.

Global decision. A global summarizer produces the final answer by explicitly conditioning on the DAG structure, hyperedge weights, and global state:

$$y_{\text{global}} = \Phi_{\text{sum}}\Big(\{\tilde{r}_k\},\ \mathcal{G}_E,\ \{w_k\},\ S_{\text{glob}}\Big). \tag{4}$$

5 Experiments

5.1 Tasks and Datasets

We evaluate HYPER-MACNET on both standard reasoning/code benchmarks. Specifically, we report results on MMLU_hard (a curated subset of more quantitative/formal subjects from MMLU (Hendrycks et al., 2020)), subject list and full model performance on MMLU are shown in Appendix C), GSM8K (Cobbe et al., 2021), and HumanEval (Chen, 2021).

To evaluate long-horizon coordination under coupled constraints, we introduce **CLM** (**C**omplex **L**ong-horizon **M**ulti-task), a dataset of 50 multi-step, cross-domain tasks spanning six MacroSystems. Each task specifies explicit goals and typically includes 3–4 hard, interacting constraints (e.g., physical/temporal, safety/robustness, economic/resource, and ethics/compliance), requiring a structured multi-stage solution or a full reasoning trace; dataset statistics are provided in Appendix E.

CLM is intentionally constructed to isolate irreducible higher-order coupling—the target regime of our method. The claim is not universal dominance, but that when feasibility hinges on joint

Table 1: Comparison of existing single-agent and multi-agent reasoning frameworks and our HYPER-MACNET on standard benchmarks. Columns **Mul.** indicates whether a method uses multiple agents; **Ada.** indicates whether the interaction structure is adaptively organized rather than fixed (✓ = yes, ✗ = no, △ = partial).

Method	Mul.	Ada.	MMLU_hard	GSM8K	HumanEval	Avg.
Single-Agent Methods						
Vanilla	✗	✗	41.65	68.16	43.29	51.03
CoT	✗	✗	52.22	74.26	60.98	62.68
AutoGPT	✗	△	45.68	67.47	48.09	53.75
Static Multi-Agent Topologies (MacNet variants)						
Graph–Chain	✓	✗	44.06	66.26	32.72	47.68
Graph–Star	✓	✗	44.27	67.90	55.49	55.89
Graph–Tree	✓	✗	43.89	69.00	48.78	53.89
Graph–Mesh	✓	✗	43.89	67.40	51.22	54.17
Graph–Layer	✓	✗	43.69	68.08	49.39	53.72
Graph–Random	✓	✗	44.19	67.40	52.44	54.68
Adaptive Multi-Agent Frameworks						
MetaGPT	✓	△	46.54	67.68	64.41	59.28
GPTSwarm	✓	△	47.30	78.43	49.69	58.47
AgentVerse	✓	△	50.06	76.33	**72.56**	66.32
HYPER-MACNET (Ours)	✓	✓	**60.09**	**79.63**	71.34	**70.35**

constraint alignment, pairwise or unstructured coordination becomes unreliable; CLM operationalizes this setting in a controlled and reproducible way, complementing standard benchmarks.

5.2 Baselines and Implementation Details

Backbone and fairness. Unless otherwise stated, all methods use the same LLM backbone (`gpt-3.5-turbo`), identical decoding settings (temperature, top-p), and the same total generation budget upper bound per instance: L=4096 tokens for standard benchmarks and L=16384 tokens for CLM. We disable external tools and retrieval for all methods. For standard benchmarks, we use N=4 agents, and for CLM, we use N=5 to better match the higher task complexity. All multi-agent methods adopt the same N. We use a unified instruction template and output format across methods to reduce formatting/verbosity biases; baseline prompts and framework configurations are provided in Appendix E.

Baselines. We compare against (i) a single-agent chain-of-thought (Wei et al., 2022) baseline, (ii) AutoGPT-style (Richards, 2023) self-looping single-agent planning, (iii) generic multi-agent frameworks (MetaGPT (Hong et al., 2023), GPTSwarm (Zhuge et al., 2024), AgentVerse (Chen et al., 2024)), and (iv) graph-based multi-agent coordination (MacNet (Qian et al., 2024b)) instantiated with a family of fixed communication topologies (chain, star, tree, mesh, layer, random). We use Graph–Random as the main MacNet baseline since it provides the best quality–cost compromise among fixed graphs. For each baseline, we only tune a small, shared set of interface-level knobs (role prompt wording, termination rules) for specific tasks under identical trial budgets, without modifying any baseline algorithm.

5.3 Evaluation Metrics and Judging Protocol

Objective benchmarks. For MMLU and MMLU_hard, we report accuracy. For GSM8K, we report answer accuracy under the official evaluation script. For HumanEval, we follow the official evaluation protocol and report pass@1.

CLM: LLM-as-a-judge. CLM outputs are open-ended and multi-criteria; we thus adopt an LLM-as-a-judge protocol. For each instance, we collect each method's final solution under the same output template, which reduces stylistic and verbosity-driven advantages. For fairness, we anonymize method identities and randomly permute candidates before judging. We use four strong judges (GPT-5.1-think, Gemini-3-pro, Grok-4-reason, DeepSeek-V3.2) under a consistent rubric (cost-effectiveness, feasibility, completeness, final impact, scalability, and alignment) to score and rank all candidates. We report the mean rank aggregated over tasks and judges, with standard de-

Table 2: Ranking of solutions from different models under AI evaluation on the CLM dataset. Each value denotes the mean ± standard deviation of the ranking score assigned by LLM judges. The same evaluation protocol is used for all compared models.

Method	GPT-5.1-think	Deepseek-V3.2	Gemini-3-pro	Grok-4-reason	Avg.
Vanilla	5.08 ± 1.68	3.72 ± 1.62	4.36 ± 1.35	4.28 ± 1.51	4.36 ± 1.54
CoT	5.24 ± 1.56	6.40 ± 0.91	6.24 ± 1.27	5.80 ± 1.25	5.92 ± 1.25
MacNet	4.60 ± 1.85	5.08 ± 1.47	5.20 ± 1.73	4.24 ± 1.98	4.78 ± 1.76
MetaGPT	4.72 ± 1.79	4.36 ± 0.99	3.08 ± 1.29	3.76 ± 1.69	3.98 ± 1.44
GPTSwarm	3.08 ± 1.19	3.20 ± 1.50	4.00 ± 1.38	3.40 ± 1.70	3.42 ± 1.44
AgentVerse	4.08 ± 1.71	3.96 ± 1.86	3.68 ± 1.86	4.76 ± 2.10	4.12 ± 1.88
HYPER-MACNET (Ours)	**1.24 ± 0.52**	**1.20 ± 0.65**	**1.44 ± 1.00**	**1.64 ± 0.99**	**1.38 ± 0.79**

Table 3: Performance evaluation across six dimensions on the CLM dataset. Each entry denotes mean of the normalized score assigned by LLM judges. For each dimension, scores are normalized with respect to the Vanilla method, so that Vanilla has value 1.0 and other methods are measured relative to it.

Dimension	CoT	AgentVerse	MacNet	MetaGPT	GPTSwarm	HYPER-MACNET
Cost-effectiveness	-7.42%	0.26%	-5.07%	0.68%	5.96%	**14.26%**
Feasibility	-7.35%	-1.95%	-4.09%	-0.21%	5.23%	**11.86%**
Completeness	-13.67%	1.96%	-3.00%	7.62%	7.30%	**20.82%**
Final impact	-10.29%	1.26%	-4.13%	0.24%	7.79%	**19.66%**
Scalability	-10.65%	5.53%	-4.06%	8.75%	6.56%	**18.01%**
Alignment with task	-0.57%	-5.75%	-2.25%	0.20%	7.63%	**10.36%**

viation computed across tasks and repeated evaluations. To assess reliability, we repeat judging $K = 10$ times; ranking stability is high (Kendall's $W \in [0.84, 0.87]$ across judges) and score consistency is strong (ICC$(3, 10)$ typically > 0.94 across dimensions). Full prompts, rubrics, aggregation, and reliability analyses are provided in Appendix D.

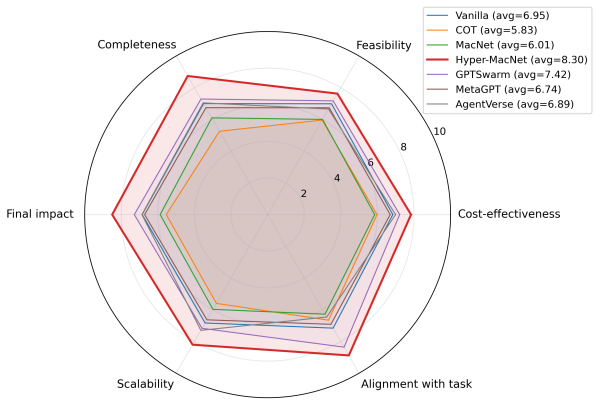

Figure 3: Radar plot of six-dimensional CLM scores, averaged over tasks and LLM judges. HYPER-MACNET achieves consistently highest scores.

5.4 Performance on Complex Tasks (CLM)

CLM tasks feature irreducible higher-order coupling: key decisions require jointly aligning heterogeneous information and interdependent constraints within a shared collaboration unit, which is hard to decompose into independent pairwise exchanges and recover by simple aggregation. Under the LLM-as-a-judge protocol, HYPER-MACNET achieves the best overall performance, with an average rank of 1.38 ± 0.79 across tasks and judges, and it ranks first under all four judges (Table 2). Strong baselines—generic MAS frameworks (MetaGPT, GPTSwarm, AgentVerse), and graph-based MACNET (best random fixed-topology variant)—form a middle group, while the single-agent CoT baseline performs worst.

Judges consistently prefer HYPER-MACNET because its solutions exhibit clearer multi-stage structure, stronger constraint coverage, and more coherent global plans. In contrast, baselines often spend substantial budget on loosely relevant branches, yielding lengthy yet partially redundant deliberation and less stable end-to-end quality. Even the strongest fixed-graph variant (MACNET) remains systematically below HYPER-MACNET, especially on instances requiring explicit multi-stage planning and cross-domain constraint coordination. We also observe that pairwise-collaboration baselines more frequently produce locally consistent partial plans that violate global constraints after merging, consistent with the proposed divide-and-

Table 4: Ablation study of different components in HYPER-MACNET. We evaluate the contribution of each key component across four benchmarks. The CLM metrics are the average values in Table 3 ΔAvg is the average difference relative to the full HYPER-MACNET.

Methods	CLM	MMLU_hard	GSM8K	HumanEval	Avg.	ΔAvg.
Hyper-MACNET (Full)	**15.82**	**60.09**	**79.63**	**71.34**	**56.72**	–
w/o Task DAG (NODAG)	12.26	55.24	77.26	67.07	52.04	-4.68
w/o Hyper-edges (NOHYPEREDGE)	13.30	52.36	75.81	63.41	51.45	-5.28
w/o Mode Switching (SINGLEMODE)	11.68	56.20	77.71	68.90	53.62	-3.10

reconcile failure mode. These results suggest that weakly structured pairwise interactions are insufficient, and that task-DAG-driven hypergraph organization with explicit collaboration modes is crucial for robust multi-agent reasoning on complex tasks.

5.5 Performance on Standard Benchmarks

Table 1 presents a comprehensive comparison on three standard benchmarks. HYPER-MACNET consistently outperforms all baselines, achieving an average score of 70.35 across the three tasks, which constitutes a substantial improvement +4.08 over the strongest baseline AgentVerse (66.32) and representative multi-agent systems such as GPTSwarm and MetaGPT. Concretely, HYPER-MACNET attains 60.09 accuracy on MMLU_hard, 79.63 on GSM8K, and 71.34 on HumanEval. The results highlight the advantage of organizing collaboration through task-aware hypergraphs rather than fixed or pairwise-only communication graphs: HYPER-MACNET more effectively aggregates and coordinates information within collaboration units than static topologies or pipeline-style frameworks. Furthermore, as indicated by the **Mul.** and **Ada.** columns, HYPER-MACNET is among the few methods that simultaneously support genuine multi-agent operation and fully adaptive interaction structures, and this combination translates into the largest performance gains on benchmarks that demand intensive coordination and structured reasoning. Although these benchmarks do not explicitly require long-horizon coordination, task decomposition still induces implicit structure (e.g., reasoning steps or subgoals), allowing hyperedge-level coordination to reduce redundancy and inconsistency even in short-horizon setting.

5.6 Ablation Studies

Table 4 summarizes an ablation study of HYPER-MACNET across four benchmarks. The full model performs best overall (Avg. $= 56.72$), and removing any component consistently degrades performance, indicating that task-structure induction, higher-order collaboration, and mode-aware coordination are complementary.

Removing the explicit task DAG and topological scheduling decreases Avg. by 4.68 and drops CLM from 15.82 to 12.26, suggesting weaker handling of long-horizon dependencies and cross-step consistency. Replacing hyper-edges with standard pairwise interactions yields the largest degradation (Avg. -5.28), consistent with losing irreducible higher-order coupling: constraints that require joint alignment across multiple agents are more likely to fragment and conflict during aggregation. Disabling mode switching also hurts performance (Avg. -3.10), with a notable CLM drop ($15.82 \rightarrow 11.68$), reflecting the need to alternate collaboration patterns across stages (decomposition, exploration, consolidation) under a fixed budget.

Overall, the gains stem from jointly modeling task dependencies (DAG), preserving higher-order coupling (hyperedges), and adapting collaboration dynamics via mode routing.

6 Conclusion

In this work, we tackle complex cross-domain reasoning for large language models by introducing HYPER-MACNET, a multi-agent framework that organizes collaboration through task-aware hypergraphs. Instead of relying on a single fixed communication topology, HYPER-MACNET decomposes each problem into a task DAG and uses hyperedges to group role-specialized agents around shared subtasks, enabling coherent, structured planning via mode-aware collaboration. Experiments on a cross-domain complex-task and standard benchmarks show that HYPER-MACNET consistently outperforms strong single-agent and multi-agent baselines, highlighting task-structured hypergraph organization as an effective paradigm for scalable multi-agent LLM systems.

Limitations

No external tool interfaces. To isolate the contribution of structured collaboration and keep evaluation controlled and fair, we run all methods without external tools (search/retrieval, code execution, solvers, or databases). Future work will study composition with tool use by enabling plug-in tool calling within each hyperedge (e.g., retrieval for fact checking, scripts/solvers for constraint validation) and writing tool outputs back to the shared blackboard, alongside executable constraint checkers to improve verifiability and reproducibility.

Text-only setting (no multimodal inputs). Our study is limited to text-only inputs and does not incorporate visual, auditory, or other multimodal signals. For tasks where constraints, evidence, or feasibility depend on perception (e.g., diagrams, tables, logs, or sensor data), performance remains bounded by pure language reasoning. A natural extension is to encode multimodal evidence as structured constraints or verifiable artifacts on the blackboard and integrate them into hyperedge-level coordination, enabling agents to jointly reason over language, perception, and tool-derived signals.

References

David Abella, Piero Birello, Leonardo Di Gaetano, Sara Ghivarello, Narayan G Sabhahit, Christel Sirocchi, and Juan Fernández-Gracia. 2023. Unraveling higher-order dynamics in collaboration networks. *arXiv preprint arXiv:2306.17521*.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Federico Battiston, Enrico Amico, Alain Barrat, Ginestra Bianconi, Guilherme Ferraz de Arruda, Benedetta Franceschiello, Iacopo Iacopini, Sonia Kéfi, Vito Latora, Yamir Moreno, and 1 others. 2021. The physics of higher-order interactions in complex systems. *Nature physics*, 17(10):1093–1098.

Federico Battiston, Giulia Cencetti, Iacopo Iacopini, Vito Latora, Maxime Lucas, Alice Patania, Jean-Gabriel Young, and Giovanni Petri. 2020. Networks beyond pairwise interactions: Structure and dynamics. *Physics reports*, 874:1–92.

Austin R Benson, David F Gleich, and Jure Leskovec. 2016. Higher-order organization of complex networks. *Science*, 353(6295):163–166.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and 1 others. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17682–17690.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, and 1 others. 2025. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657*.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. 2023. Chateval: Towards better llm-based evaluators through multi-agent debate. *arXiv preprint arXiv:2308.07201*.

Mark Chen. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, and 1 others. 2024. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *ICLR*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3558–3565.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, and 1 others. 2023. Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008.

Thierry Njougouo, Timoteo Carletti, and Elio Tuci. 2025. Collective decision-making with higher-order interactions on d-uniform hypergraphs. *arXiv preprint arXiv:2511.13452*.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, and 1 others. 2024a. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186.

Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, and 1 others. 2024b. Scaling large language model-based multi-agent collaboration. *arXiv preprint arXiv:2406.07155*.

Toran Bruce Richards. 2023. Autogpt. in https://github.com/significant-gravitas/autogpt, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. 2023. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arXiv:2305.17493*.

Yu Tian, Xingliang Huang, Ruigang Niu, Hongfeng Yu, Peijin Wang, and Xian Sun. 2022. Hypertron: Explicit social-temporal hypergraph framework for multi-agent forecasting. In *IJCAI*, pages 1356–1362.

Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. 2023. On the planning abilities of large language models-a critical investigation. *Advances in Neural Information Processing Systems*, 36:75993–76005.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Anita Williams Woolley, Christopher F Chabris, Alex Pentland, Nada Hashmi, and Thomas W Malone. 2010. Evidence for a collective intelligence factor in the performance of human groups. *science*, 330(6004):686–688.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, and 1 others. 2024. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. 2019. Hypergcn: A new method for training graph convolutional networks on hypergraphs. *Advances in neural information processing systems*, 32.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models, 2023. *URL https://arxiv. org/abs/2305.10601*, 3:1.

Rui Ye, Shuo Tang, Rui Ge, Yaxin Du, Zhenfei Yin, Siheng Chen, and Jing Shao. 2025. Mas-gpt: Training llms to build llm-based multi-agent systems. *arXiv preprint arXiv:2503.03686*.

Heng Zhang, Yuling Shi, Xiaodong Gu, Zijian Zhang, Haochen You, Lubin Gan, Yilei Yuan, and Jin Huang. 2025. Hyperagent: Leveraging hypergraphs for topology optimization in multi-agent communication. *arXiv preprint arXiv:2510.10611*.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2006. Learning with hypergraphs: Clustering, classification, and embedding. *Advances in neural information processing systems*, 19.

Heng Zhou, Hejia Geng, Xiangyuan Xue, Li Kang, Yiran Qin, Zhiyong Wang, Zhenfei Yin, and Lei Bai. 2025. Reso: A reward-driven self-organizing llm-based multi-agent system for reasoning tasks. *arXiv preprint arXiv:2503.02390*.

Tianyu Zhu, Xinli Shi, Xiangping Xu, Jie Gui, and Jinde Cao. 2024. Hypercomm: Hypergraph-based communication in multi-agent reinforcement learning. *Neural Networks*, 178:106432.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. 2024. Gptswarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*.

A Contents

B Additional Spectral Hypergraph View and Structural Context

This appendix provides a formal treatment of the incidence structure and normalized operators for weighted hypergraphs. While the main text employs hypergraphs primarily as an executable organizational interface (defining team membership and dependency constraints), the spectral view presented here offers an analyzable structural prior. This formulation serves as an interpretability lens that quantifies the couplings induced by shared hyperedges and supports a lightweight, structure-weighted aggregation mechanism (Section B.6). Crucially, this spectral analysis is intended as an analytical tool rather than a learned component; it informs the system's coordination logic without introducing any trainable parameters, thereby maintaining the framework's parameter-free nature.

B.1 Incidence, Weights, and Degrees

Consider a weighted hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E}, w)$ with $|\mathcal{V}| = N$ and $|\mathcal{E}| = M$. The node–hyperedge incidence structure is encoded by a binary incidence matrix

$$\mathbf{H} \in \{0,1\}^{N\times M}, \quad \mathbf{H}_{ik} = \begin{cases} 1, & v_i \in e_k, \\ 0, & v_i \notin e_k. \end{cases}$$

Let $\mathbf{W} = \mathrm{diag}(w_1, \ldots, w_M)$ be the diagonal matrix of hyperedge weights with $w_k = w(e_k) > 0$.

We define the (weighted) node degree and the hyperedge size (degree) as

$$d(v_i) = \sum_{k=1}^{M} w_k \, \mathbf{H}_{ik}, \quad \delta(e_k) = \sum_{i=1}^{N} \mathbf{H}_{ik},$$

and the corresponding diagonal degree matrices

$$\mathbf{D}_v = \mathrm{diag}\big(d(v_1), \ldots, d(v_N)\big),$$
$$\mathbf{D}_e = \mathrm{diag}\big(\delta(e_1), \ldots, \delta(e_M)\big).$$

Intuitively, $\mathbf{W}$ emphasizes important hyperedges, $\mathbf{D}_e^{-1}$ normalizes for team size, and $\mathbf{D}_v^{-1/2}$ prevents high-participation nodes from dominating propagation.

B.2 Normalized Hypergraph Adjacency and Laplacian

Following standard constructions in spectral hypergraph theory, we define the normalized hypergraph adjacency operator

$$\mathbf{A}_{\mathcal{H}} = \mathbf{D}_v^{-\frac{1}{2}} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{D}_v^{-\frac{1}{2}} \in \mathbb{R}^{N\times N}, \quad (5)$$

and the corresponding normalized hypergraph Laplacian

$$\mathbf{L}_{\mathcal{H}} = \mathbf{I} - \mathbf{A}_{\mathcal{H}}. \quad (6)$$

The operator in Eq. (5) admits a natural two-stage interpretation: information propagates from nodes to incident hyperedges and back to other nodes within the same hyperedges, with contributions modulated by hyperedge weights and normalized by team sizes and node degrees.

B.3 Coupling Strength Induced by Shared Hyperedges

We interpret each entry of $\mathbf{A}_{\mathcal{H}}$ as a *structurally induced coupling strength*. Define

$$\alpha_{ij} = \big(\mathbf{A}_{\mathcal{H}}\big)_{ij}.$$

Expanding Eq. (5) yields the closed-form expression

$$\alpha_{ij} = \frac{1}{\sqrt{d(v_i)\, d(v_j)}} \sum_{k=1}^{M} \frac{w_k}{\delta(e_k)} \, \mathbf{H}_{ik} \mathbf{H}_{jk}. \quad (7)$$

Thus, α_{ij} increases when v_i and v_j co-occur in many hyperedges, especially high-weight ones, and decreases for large hyperedges (via $\delta(e_k)$) and high-participation nodes (via $d(v_i)$), which improves interpretability and prevents trivial amplification.

B.4 Structural Context Operator (Optional)

For analysis or optional structural bias, let $x_i^t \in \mathbb{R}^d$ denote a vector representation of the state of agent a_i (aligned with node v_i) at step t, e.g., an embedding of a *structured textual state record* (role description, memory summary, and intermediate artifacts). Stacking row-wise gives $\mathbf{X}^t \in \mathbb{R}^{N\times d}$. We define the structural context transformation

$$\widetilde{\mathbf{X}}^t = \mathbf{A}_{\mathcal{H}} \, \mathbf{X}^t, \quad \tilde{x}_i^t = \sum_{j=1}^{N} \alpha_{ij} \, x_j^t, \quad (8)$$

which aggregates neighbor states according to couplings in Eq. (7). This operator is not required by our main pipeline and introduces no trainable parameters; we use it mainly as an interpretable structural prior.

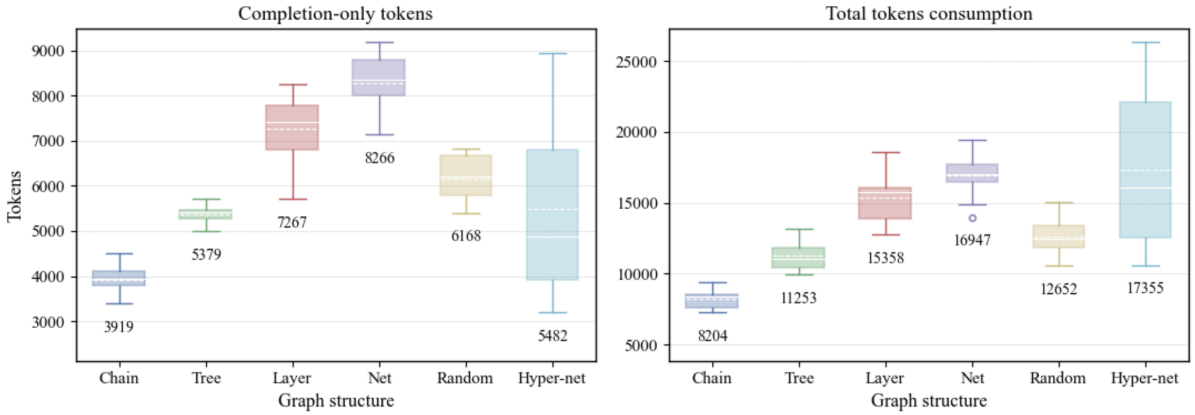

Figure 4: CLM token usage under different collaboration topologies; We vary only the interaction topology, with the left plot showing completion tokens and the right plot showing total tokens. Hyper-net corresponds to our task-aware HYPER-MACNET architecture.

B.5 Irreducibility Beyond Pairwise Graph Expansions

A common reduction from hypergraphs to ordinary graphs is a pairwise (clique) expansion, e.g.,

$$\mathbf{A}_{ij}^{\text{clique}} = \sum_{k=1}^{M} \mathbb{I}\big[\mathbf{H}_{ik}\mathbf{H}_{jk} = 1\big],$$

which replaces each hyperedge by pairwise links. However, such expansions do not faithfully capture (i) team-size normalization (the role of $\delta(e_k)$), (ii) hyperedge importance weighting (the role of w_k), and (iii) the execution semantics of a *joint* collaboration unit (a hyperedge team) as opposed to a collection of independent dyadic interactions. The normalized operator in Eq. (5) preserves these factors through $\mathbf{W}$, $\mathbf{D}_e^{-1}$, and $\mathbf{D}_v^{-1/2}$, providing a more principled characterization of higher-order couplings.

B.6 Structure-Weighted Aggregation Interface

To avoid disconnect between the spectral formalization and the system implementation, we describe a minimal interface that can be optionally plugged into cross-hyperedge/global aggregation. Let u_j denote an evidence vector (or scalar confidence) produced by node/agent j (e.g., an embedding of an intermediate artifact, a scored candidate, or a structured record). We define a structure-weighted aggregation for node i as

$$\mathrm{Agg}_i(\{u_j\}_{j=1}^{N}) = \sum_{j=1}^{N} \alpha_{ij}\, u_j, \qquad (9)$$

where α_{ij} is given by Eq. (7). This aggregation prioritizes evidence from structurally coupled agents and attenuates signals from weakly related ones, supporting globally coherent integration under higher-order collaboration structure. Importantly, Eq. (9) can be used as a lightweight bias term or an analysis tool and does not require any additional learning beyond the base LLM.

C Additional Experiments

C.1 Inference Cost under Different Collaboration Topologies

To analyze how collaboration topology affects the cost of multi-agent reasoning, we fix the backbone model, the number of agents (five), and all prompt templates, and vary only the interaction structure. On the CLM dataset, we instantiate MacNet with several representative static topologies (Chain, Tree, Layer, Net, Random) and compare them with our proposed HYPER-MACNET. and plot the token distributions in Figure 4.

Overall, the static topologies exhibit a clear trade-off between structural simplicity and communication cost: simpler structures such as Chain and Tree tend to require fewer tokens but offer limited capacity for rich coordination, whereas denser structures such as Layer and Net allow more information exchange but quickly amplify the number of messages and tokens, especially on harder tasks. Random lies in between and provides a reasonable but still task-sensitive quality–cost compromise, which is why we adopt it as the MacNet baseline in our main comparisons.

HYPER-MACNET differs from these static designs in that its "Hyper-net" structure is not a sin-

gle fixed graph, but a task-aware hypergraph that activates different hyper-edges and collaboration modes according to the task DAG and subtask dependencies. As a result, its token usage reflects an adaptive adjustment of structural complexity: for relatively straightforward tasks, the induced interaction pattern remains close to sparse topologies, while for constraint-heavy or cross-domain tasks, local regions of the hypergraph become denser where additional coordination is needed. Empirically, HYPER-MACNET operates in a similar or slightly higher cost regime than the stronger static topologies, but achieves significantly better judged quality on CLM and shows more stable token usage across tasks, indicating a better quality–per–token trade-off than hand-tuned fixed graphs.

C.2 MMLU and MMLU_hard

Subjects of MMLU_hard. The MMLU-Hard benchmark evaluates model performance on a subset of particularly challenging questions drawn from the original MMLU suite. The following subjects are included in our evaluation (total 1000 tasks):

- **abstract_algebra.** Covers groups, rings, fields, homomorphisms, and quotient structures. Questions emphasize formal definitions, theorem-level reasoning, and structural properties rather than numerical computation.
- **college_computer_science.** Includes undergraduate topics in algorithms and data structures, computational complexity, operating systems, computer architecture, and programming language principles, focusing on algorithmic reasoning and foundational system-level understanding.
- **college_mathematics.** Encompasses standard undergraduate mathematics (e.g., calculus, linear algebra, probability theory, differential equations). Questions typically require multi-step derivations and careful conceptual reasoning.
- **college_physics.** Covers introductory university physics such as classical mechanics, electromagnetism, thermodynamics, and basic modern physics, assessing both qualitative physical intuition and quantitative analysis.
- **high_school_mathematics.** Draws from secondary-level algebra, geometry, functions, sequences, and analytic geometry. In the hard subset, problems often combine multiple concepts and require non-trivial logical inference.
- **high_school_statistics.** Focuses on basic probability and statistics, including distributions, expectation/variance, and elementary hypothesis testing, emphasizing statistical reasoning and interpretation.
- **machine_learning.** Includes conceptual questions on supervised/unsupervised learning, loss functions, optimization, and generalization, emphasizing theoretical understanding and method comparison rather than implementation details.

Results of MMLU. On the MMLU benchmark, Hyper-MACNET achieves the highest overall accuracy at 72.28%, outperforming all baseline methods by a clear margin (Table 5). Compared with the vanilla single-agent baseline (66.36%), Hyper-MACNET yields a gain of nearly 6 percentage points, indicating the effectiveness of structured multi-agent coordination beyond simple prompting. While chain-of-thought prompting (CoT), AutoGPT, and other multi-agent frameworks such as GPTSwarm and Agentverse provide moderate improvements over the vanilla baseline, their performance remains clustered around 68–69%, suggesting limited benefits from unstructured or fixed interaction patterns. In contrast, Hyper-MACNET consistently surpasses these approaches, highlighting the advantage of explicitly modeling task structure and higher-order collaboration when addressing heterogeneous, multi-domain questions in MMLU.

D LLM-as-a-Judge Protocol

Many tasks evaluated in this work are open-ended and multi-criteria, and thus do not admit a single verifiable ground-truth solution. In such settings, candidate solutions may differ in structure and trade-offs (e.g., feasibility vs. completeness), making standard automatic metrics insufficient. To enable scalable and reproducible evaluation, we adopt an LLM-as-a-Judge protocol to score and rank solution-level outputs.

D.1 Candidate Collection and Judging Models

For each task instance, we collect the final outputs produced by all methods. When a task specifies explicit requirements, we additionally collect the

	Vanilla	CoT	AutoGPT	MacNet-best	GPTSwarm	Agentverse	**Hyper-MACNET**
Accuracy(%)	66.36	69.00	67.58	68.77	69.11	68.54	**72.28**

Table 5: Overall accuracy on MMLU.

corresponding structured plans and necessary reasoning artifacts, so that judges can verify requirement coverage and constraint satisfaction.

We use a panel of high-performance LLMs as judges: `GPT-5.1-think`, `Gemini-3-pro`, `Grok-4-reason`, and `DeepSeek-V3.2`. Each judge independently scores and ranks all candidates for each task.

D.2 Anonymization and Randomization

To reduce identity- and presentation-induced bias, we (i) anonymize all candidate solutions by removing any source-identifying information, and (ii) randomly shuffle the order of candidates for each task instance prior to evaluation. All judging models are prompted with the same unified evaluation prompt to ensure consistent criteria.

D.3 Scoring Dimensions and Rubric Prompt

Each candidate is evaluated along six dimensions on a 10-point scale (rounded to two decimals; baseline score 6.00). We use the following anchor rubric for calibration. Scores may take intermediate values (e.g., 7.50) when a solution falls between anchors.

(1) Cost-effectiveness.

- **3-point anchor:** Disproportionately high input cost (compute, interaction, or implementation) for marginal gains; clear resource waste or "overkill".
- **6-point anchor:** Costs are broadly proportional to achieved quality; comparable to common practice; neither notably efficient nor wasteful.
- **9-point anchor:** Achieves strong results with minimal additional cost; high leverage (large quality gain per unit cost); avoids unnecessary overhead.

(2) Feasibility.

- **3-point anchor:** Relies on unavailable capabilities, unrealistic assumptions, or extremely complex execution steps; high risk of failure in practice.
- **6-point anchor:** Logically sound and implementable, but requires non-trivial expertise, engineering effort, or a standard development/execution cycle.
- **9-point anchor:** Simple, actionable, and near plug-and-play; clear steps with low barrier to execution; minimal dependency on special resources.

(3) Completeness.

- **3-point anchor:** Missing essential steps; unclear or absent input/output specification; ignores boundary conditions and potential failure modes.
- **6-point anchor:** Covers the main workflow and solves the nominal case, but has gaps for edge cases, exceptions, or secondary constraints.
- **9-point anchor:** End-to-end and logically tight; clearly specifies inputs/outputs; proactively addresses exceptions, edge cases, and includes fallback or contingency plans.

(4) Final impact.

- **3-point anchor:** Fails to solve the task or introduces new bugs/safety issues; produces negative or misleading outcomes.
- **6-point anchor:** Resolves the explicit problem but results are mediocre or rough; "works" without notable quality, efficiency, or clarity.
- **9-point anchor:** High-quality outcome that exceeds baseline expectations; efficient, clean, and well-structured; demonstrates strong professional judgment and execution.

(5) Scalability / extensibility.

- **3-point anchor:** Highly coupled and hard-coded; brittle; one-off solution that is difficult to reuse or adapt; small changes require large rewrites.

- **6-point anchor:** Moderately modular; exposes basic parameters/interfaces; can accommodate a reasonable range of requirement changes.

- **9-point anchor:** Highly decoupled and modular; designed for reuse and iteration; easy to migrate, extend, and evolve into a general component.

(6) Alignment with task requirements.

- **3-point anchor:** Severely violates explicit constraints, uses prohibited tools/methods, or is substantially off-topic.

- **6-point anchor:** Satisfies the core requirements but has minor deviations in secondary constraints (e.g., format, length, or non-critical instructions).

- **9-point anchor:** Strictly follows all explicit and implicit instructions; matches constraints with near machine-level precision.

Self-consistency constraint. We apply a judging consistency rule: when a solution is rated as low feasibility, its final impact should typically not exceed the high-point anchor, to avoid rewarding impractical but superficially impressive proposals.

D.4 Stability and Consistency Analysis

To assess the reliability of LLM-as-a-Judge evaluation, we analyze **score consistency** and **ranking stability** under repeated judging. Concretely, we evaluate **7 methods** and repeat the judging **10 times** (by re-randomizing candidate order and re-invoking the judges under the same unified prompt), yielding $K = 10$ repeated outcomes per method.

Score consistency (ICC). Score-level consistency measures whether the *absolute* scores assigned to the same method remain stable across repeated judging. Let x_{ij} denote the scalar score of method $i \in \{1, \dots, N\}$ in repeat $j \in \{1, \dots, K\}$, where $N = 7$ and $K = 10$. Define means

$$\bar{x}_{i\cdot} = \frac{1}{K}\sum_{j=1}^{K} x_{ij}, \quad \bar{x}_{\cdot j} = \frac{1}{N}\sum_{i=1}^{N} x_{ij},$$

$$\bar{x}_{\cdot\cdot} = \frac{1}{NK}\sum_{i=1}^{N}\sum_{j=1}^{K} x_{ij}.$$

We compute the two-way ANOVA mean squares

$$MS_R = \frac{K\sum_{i=1}^{N}(\bar{x}_{i\cdot} - \bar{x}_{\cdot\cdot})^2}{N-1},$$

$$MS_E = \frac{\sum_{i=1}^{N}\sum_{j=1}^{K}(x_{ij} - \bar{x}_{i\cdot} - \bar{x}_{\cdot j} + \bar{x}_{\cdot\cdot})^2}{(N-1)(K-1)}.$$

We report the intraclass correlation coefficient (ICC) for single-measure agreement:

$$\mathrm{ICC} = \frac{MS_R - MS_E}{MS_R + (K-1)MS_E},$$

where a higher ICC indicates more repeatable (less noisy) scores across the 10 repeated judging runs.

Ranking stability (Kendall's W). Ranking stability measures whether the *relative ordering* of methods is consistent across repeated judging. Let r_{ij} be the rank of method $i \in \{1, \dots, M\}$ in repeat $j \in \{1, \dots, K\}$, with $M = 7$ and $K = 10$ (smaller rank is better). Define rank sums

$$R_i = \sum_{j=1}^{K} r_{ij}, \quad \bar{R} = \frac{1}{M}\sum_{i=1}^{M} R_i,$$

and dispersion

$$S = \sum_{i=1}^{M}(R_i - \bar{R})^2.$$

We compute Kendall's coefficient of concordance

$$W = \frac{12S}{K^2(M^3 - M)} \in [0, 1],$$

where larger W indicates stronger agreement among the 10 repeated rankings (i.e., higher ranking stability).

From Table 6, across all four judging LLMs, score-level reliability is consistently high: the **ICC(3,10)** values exceed **0.94** for most dimensions, indicating that repeated judging produces highly reproducible scores. The strongest consistency is observed for Completeness, Final impact, and Scalability (typically ≥ 0.97). Ranking-level stability is also strong, with Kendall's W in $[0.8407, 0.8736]$ across judges, implying that the relative ordering of the 7 methods remains largely consistent over the 10 repeated evaluations. Overall, these results support that our LLM-as-a-Judge protocol is stable in both absolute scoring and comparative ranking.

Table 6: Judge consistency and ranking stability in LLM-as-a-Judge evaluation.

Judging LLM	**Grok-4-reason**	**Deepseek-V3.2**	**Gemini-3-pro**	**GPT-5.1-think**
Score consistency ICC (3,10)				
Cost-effectiveness	0.9777	0.9798	0.9643	0.9494
Feasibility	0.9424	0.8720	0.9389	0.9588
Completeness	0.9811	0.9807	0.9729	0.9785
Final impact	0.9687	0.9856	0.9809	0.9744
Scalability	0.9807	0.9799	0.9761	0.9678
Alignment with task	0.8805	0.7882	0.9365	0.9386
Ranking stability (Kendall's W)				
Rank	0.8736	0.8407	0.8407	0.8593

E Description and statistical results of the CLM dataset

To evaluate long-horizon planning under complex, coupled constraints, we construct **CLM** (*Complex Long-horizon Multi-task Benchmark*), a curated benchmark of **50** high-difficulty task scenarios. CLM is designed around intrinsic interdisciplinarity: each task reflects a real-world techno-social engineering problem whose feasible solution must jointly integrate *physical*, *informational*, and *socio-economic* factors, rather than being solvable by single-domain knowledge alone.

CLM tasks are constructed following three principles:

(1) each task involves multiple interdependent constraints that cannot be verified independently;

(2) successful solutions require coordination across at least three roles or subtasks;

(3) no single agent can trivially enumerate a complete solution in one step.

E.1 Taxonomy and Macro-level Distribution

CLM adopts a two-level taxonomy for systematic organization. At the top level, **MacroSystem** defines six pillars of modern techno-social systems; at the second level, **EcoSystem** refines each MacroSystem into application subdomains. This taxonomy is defined by the *primary application context* of each task and does not imply domain isolation—cross-domain coupling is a central feature throughout the dataset. The six MacroSystems and their counts are: *Energy & Enviro* (12/50, 24%), *Computing & Network* (10/50, 20%), *Smart Cities* (7/50, 14%), *Life Sciences* (7/50, 14%), *Social & Digital Finance* (7/50, 14%), and *Aerospace & Physics* (7/50, 14%).

Figure 5: Distribution of CLM Dataset

E.2 Within-task Cross-domain Coupling and Constraints

A defining characteristic of CLM is within-task cross-disciplinary coupling: most tasks require simultaneously invoking two or more distinct knowledge bases, where feasibility emerges from logical entanglement rather than simple knowledge aggregation. Representative coupling patterns include:

(i) physics–perception–chemistry coupling (e.g., mapping nonlinear chemical mixtures to psychophysical perception thresholds, or closing real-time loops between thermodynamic constraints and learned inversion in industrial processes);

(ii) law–logic–code coupling (e.g., translating international commercial law into deontic-logic code with formal verification);

(iii) biology–cryptography–finance coupling (e.g., genomic pricing that jointly requires actuarial risk modeling, bioinformatic mutation analysis, and provable privacy/security constraints).

These patterns intentionally stress cross-domain

coordination over isolated domain expertise.

Each CLM task embeds multi-dimensional coupled constraints, with **3–4 hard requirements per task on average**. We group constraints into four major categories:

(i) physical & temporal constraints (strict latency, dynamics, or physics limits),

(ii) safety & robustness constraints (stability under failures, attacks, or uncertainty),

(iii) economic & resource-efficiency constraints (cost, energy, or resource budgets),

(iv) ethical & compliance constraints (fairness, privacy, and legal requirements).

These constraints are intentionally coupled: satisfying only a subset typically yields solutions that are incomplete or logically infeasible, thereby requiring globally consistent, long-horizon reasoning.

E.3 CLM Problem Examples and Output Prompt

CLM Problem examples and output prompts are shown in Figure 6 and Figure 7.

CLM Task Input Example: Capital Allocation under Coupled Constraints

Background.
Zhilian Cloud Services Co., Ltd. is a mid-sized enterprise SaaS provider with foreign investment background. The company focuses on enterprise-level cloud services and operates in a highly competitive cloud infrastructure market. Its core offering is basic cloud storage services, while it is actively expanding into emerging markets and AI-integrated product lines. Facing intense competition from major cloud incumbents, the company urgently seeks differentiated growth strategies and brand positioning. The company plans to allocate funds from an overseas financing round of RMB 20 million. Among these, RMB 12 million is designated for rapid validation of three business lines to explore growth boundaries. The founder explicitly requires that at least RMB 3 million must be reserved for the core business to ensure service-level agreement (SLA) compliance and operational security for existing customers.

Business Lines.

- **A. Core Business: Enterprise Cloud Storage Subscription (Low Risk)**
  Provides standardized cloud storage and backup services to small and medium-sized enterprises on an annual subscription basis. Customer churn rate is below 5%, and this line generates stable cash flow and serves as the primary profit source.
- **B. Market Expansion: Latin American SME Penetration and High-end Industry Solutions (Medium Risk)**
  Investment focuses on building local compliance teams, participating in industry summits, and delivering customized solutions. Target verticals include manufacturing and retail sectors.
- **C. R&D / Breakthrough Project: Generative AI Customer Service Engine (High Risk, High Return)**
  Aims to develop a multilingual generative AI customer service model for productization and large-enterprise procurement. Funding is used for data acquisition, model iteration, and global patent deployment.

Task.
Given the above context, determine how the company should allocate capital across the three business lines to address current challenges while enabling sustainable future growth. The solution must respect the minimum funding constraint for the core business and balance risk, return, and strategic optionality.

Figure 6: CLM Task Input Example

System Output Prompt (Assistant)

As a solution expert, you must:

1. Provide a complete, structured solution in Markdown.
2. Your output must include the following sections:
   - **Problem Analysis**: your core understanding of the user's needs
   - **Solution**: detailed, actionable step-by-step instructions
   - **Key Points**: critical considerations and core logic
   - **Expected Results**: a clear description of the deliverables
3. If code examples are needed, use a standard Markdown code block in the following format:

````
```python
# code comments
implementation
```
````

4. Do not output any explanatory preface or any concluding remarks.
5. Do not add any extra characters (e.g., # or /) before Markdown markers.
6. The content must be directly usable; avoid placeholders and ellipses.
7. Use clear, concise professional language.
8. Output the complete Markdown solution only, with no additional text.

Figure 7: System Output Prompt