

Exploring Advanced LLM Multi-Agent Systems Based on Blackboard Architecture

Bochen Han¹, Songmao Zhang¹,

¹State Key Laboratory of Mathematical Sciences
Academy of Mathematics and Systems Science, CAS
Beijing 100190, China

Correspondence: hanbochen@amss.ac.cn

Abstract

In this paper, we propose to incorporate the blackboard architecture into LLM multi-agent systems (MASs) so that (1) agents with various roles can share all the information and others' messages during the whole problem-solving process, (2) agents that will take actions are selected based on the current content of the blackboard, and (3) the selection and execution round is repeated until a consensus is reached on the blackboard. We develop the first implementation of this proposal and conduct experiments on commonsense knowledge, reasoning and mathematical datasets. The results show that our system can be competitive with the SOTA static and dynamic MASs by achieving the best average performance, and at the same time manage to spend less tokens. Our proposal has the potential to enable complex and dynamic problem-solving where well-defined structures or workflows are unavailable¹.

1 Introduction

With the prevalence of large language models (LLMs), LLM-based agentic systems have attracted much attention and produced achievements in various downstream tasks including planning and reasoning (Putta et al., 2024; Hao et al., 2023; Sun et al., 2024), code generation (Tang et al., 2024; Yang et al., 2024), and many more. From the powerful capabilities possessed by single agent systems, LLM-based multi-agent systems (MASs) emerged, aiming to utilize collective intelligence to further enhance problem-solving performances (Chan et al.; Li et al., 2023a,b; Yin et al., 2023).

Most LLM-based MASs proposed so far utilize fixed architectures with pre-defined agent roles and collaboration mechanisms, which often requires manual construction and thus lack generality. Some recent studies develop dynamic MASs (Hu et al., 2025; Zhang et al., 2025d; Shang et al.,

2025; Liu et al., 2024a; Zhang et al., 2025a), also called autonomous MASs, which configure structures and communication strategies based on tasks and environment feedbacks. Such MASs are modularized and the optimized MAS configurations are searched in specified spaces. Compared with fixed MASs, they often have an additional, time-consuming training step and the simplified search spaces cannot cover all kinds of collaboration architectures. These two-step approaches essentially use fixed collaboration mechanisms in problem solving obtained from the supervised training based on a small number of samples in the first step.

As far back as in the 80's of the last century, the blackboard architecture was proposed (Nii, 1986; Hayes-Roth, 1985) as a decentralized problem-solving approach imitating that a group of human experts work together around a shared blackboard and each contributes her/his own solutions to the blackboard until a collective decision is reached. We intend to incorporate the blackboard architecture into the LLM-based MAS and propose the blackboard-based LLM multi-agent system (bMAS) in this paper. We design a general bMAS to have three core components: (1) a control unit, (2) a blackboard, and (3) a group of LLM-based agents with different roles. For the given problem, the control unit selects the agents to participate in the current round of problem-solving according to the current messages on the blackboard. Then, each selected agent takes the contents of the whole blackboard as a prompt to its LLM and writes back to the blackboard the output produced by its LLM. Now the next round of problem-solving starts, and the process will not stop until a final solution is decided on the blackboard.

Based on the general framework of bMAS, we present and implement a concrete bMAS in this paper and evaluate its performance on mathematical and reasoning tasks. We call it LbMAS as the first blackboard-based LLM multi-agent system.

¹Code will be released soon [here](#)

In LbMAS, the agents are designated as query-related experts and constant agents including planner, decider, critic, cleaner and conflict-resolver, and their LLMs are chosen randomly from a set of LLMs when the problem-solving process begins. The blackboard is divided into two parts, a public space and private spaces, where the former is for all agents to use and the latter solely for those involved in a debate or verification of private matter. In the blackboard system, agents communicate solely through the blackboard that contains all historical contexts. The blackboard actually serves as a public storage space to replace the memory modules typically possessed by LLM agents. The control unit of LbMAS also depends on an LLM agent to make selections for the given problem. The evaluation on various benchmarks show that LbMAS, while token-economical, obtains competitive performance when compared with both fixed and dynamic SOTA MASs.

Our contribution in this paper is two-fold. Firstly, we propose to introduce the blackboard architecture into LLM-based MAS and design bMAS as a general framework where various LLM agents communicate through a blackboard and contribute to the blackboard in an autonomous way. Secondly, we develop an implementation of bMAS called LbMAS, which, differently from other autonomous MASs, can adjust collaboration mechanisms in a timely manner according to contents of the blackboard. LbMAS has the potential to enable complex and dynamic problem-solving where well-defined structures or workflows are unavailable, just as the blackboard architecture was initially proposed for thirty years ago. And with the power of LLMs, the ancient AI architecture can revive and in turn facilitate the development of agentic systems.

2 Related Work

Multi-Agent System. MAS is a computerized system composed of multiple interacting intelligent agents (Tran et al., 2025). In this paper, we only discuss the LLM-based multi-agent systems, whose key components are introduced as follows.

- Agents: a group of LLM agents with roles, capabilities, and actions. LLM-empowered capabilities like learning, planning, reasoning and decision making yield intelligence to the agents and overall system. Numerous agentic systems emerged for solving real-life tasks such as code generation (Tang et al., 2024; Yang et al., 2024), medical di-

agnosis (Tu et al., 2025; Schmidgall et al., 2025b; McDuff et al., 2025), and research assistant (Baek et al., 2025; Schmidgall et al., 2025a).

- Environment: the external world where agents are situated and can sense and act upon. Environments can be simulated or physical spaces such as factories, roads, power grids, etc.
- Collaboration mechanism: the pattern of collaboration and communication among agents. Generally there are two types of collaboration mechanism: cooperation (Islam et al., 2024; Li et al., 2023a; Shinn et al., 2023; He et al., 2023; Das et al., 2023) and competition (He et al., 2023; Chen et al., 2024a). Cooperation in an LLM-based MAS occurs when agents align their individual objectives with the same goal (Tran et al., 2025), whereas competition occurs when there are conflicting objectives or scenarios with limited resources (Tran et al., 2025). Debate is a specific type of competition, used when agents engage in argumentative interactions, presenting and defending their own viewpoints or solutions, and critiquing those of others, for the purpose of reaching a consensus or a more refined solution (Yin et al., 2023; Liang et al., 2024; Chan et al.).

Collaboration mechanism can also be classified from other perspectives (Tran et al., 2025). From the structure, we can have centralized (Jiang et al., 2023; Qiao et al., 2024), decentralized (Yin et al., 2023; Zhang et al., 2023; Chen et al., 2024b; Zhang et al., 2024a), and hierarchical collaboration mechanism (Chan et al.; Li et al., 2023a; Du et al., 2023). According to protocols on when agents respond and how to interact with each other, we can divide the collaboration mechanism into rule-based (Zhang et al., 2024a; Chen et al., 2025), role-based (Chen et al., 2024b; Hong et al., 2024; Talebirad and Nadiri, 2023), and model-based (Li et al., 2023b; Xu et al., 2024).

The above-mentioned works use predefined collaboration mechanisms that cannot be dynamically adjusted. Recent studies focus on automatic optimization of multi-agent systems where the elements optimized are different. DsPy(Khattab et al., 2024) conducts prompt optimization of agents; DyLAN(Liu et al., 2024b), GPTSwarm(Zhuge et al., 2024) and AgentPrune(Zhang et al., 2025b) focus on orchestrating interactions among agents. Specifically, DyLAN dynamically activates the composition of agents and GPTSwarm optimizes the connections between agentic nodes using a policy gradient algorithm. State-of-the-art automatic meth-

ods attempt to do more comprehensive optimizations. ADAS(Hu et al., 2025) and AFlow(Zhang et al., 2025d) achieve multi-agent workflow automation; MaAS(Zhang et al., 2025a) searches for distribution of architectures rather than a single final structure. These search-based methods need a supervised training step with samples in advance and are thus not flexible enough.

Blackboard System. The Blackboard Architecture (BA) was introduced by Hayes-Roth in 1985 (Hayes-Roth, 1985) as a complicated task-solving strategy allowing different knowledge sources to communicate by means of a common information field. It offers a compelling approach to address the control problem in expert systems (Muhs Mukhtar et al., 2025).

The BA itself relies on three core components: knowledge sources, blackboard, and control unit. Knowledge sources are independent modules that contribute specific expertise to the problem. They do not need to know about the existence of others, but they have to understand the state of the problem-solving process and the representation of relevant information on the blackboard (Corkill, 1991). Knowledge sources can be represented with different types of knowledge, including rule-based systems, case-based systems, neural networks, fuzzy logic systems, genetic algorithms, legacy software systems and others (Rudenko and Borisov, 2008). Blackboard is used as a global database for sharing different information, such as input data, partial solutions, alternatives, and final solutions. Knowledge sources produce changes to the blackboard that incrementally lead to a solution, or a set of acceptable solutions, to the problem. The interaction among knowledge sources occurs only through changes on the blackboard (Nii, 1986). The control unit makes runtime decisions about which knowledge sources to execute next for optimal solution of the problem. (Corkill, 1991).

Blackboard systems have been designed specifically to deal with complex, ill-structured problem domains and to allow exploratory programming of knowledge-based systems by integrating heterogeneous knowledge sources (Cavazza et al., 2001). This is exactly the reason we propose to introduce BA into the current LLM multi-agent systems. Note that the combination of agents with BA has been studied in (Ettabaa et al., 2006; Szymanski et al., 2018; Dong et al., 2005), whereas the

agents then differ fundamentally from the agents nowadays empowered by LLMs.

3 Method

The framework of bMAS is both illustrated in Figure 1 and presented in Algorithm 1. The blackboard is a multifunctional space including public and private parts, where the former stores dialogues and knowledge that all agents can see, and the latter allows particular agents to debate or self-reflect. The agent group contains agents with various functions, such as planning, reasoning, criticizing, tool use, etc. The control unit takes query and the current content of blackboard as input, and selects suitable agents to act in the next round of problem-solving. Such a timely, iterated process enables a dynamic adaptation of collaboration mechanism among agents.

Further, we implement LbMAS to include an agent generation module in addition to the three general components of bMAS. The overall workflow of LbMAS is also illustrated in Figure 1. For a given query q , query-related agents are generated, and then the blackboard cycle starts, where the control unit iteratively selects agents from agent group to contribute to the blackboard. The cycle continues until the stopping condition is satisfied. Finally, the solution is given based on the messages on the blackboard. These steps are detailed as follows.

3.1 Generating agents

When dealing with diverse queries from different domains, it is advisable to generate query-related expert identities and instruct LLMs to respond as the generated experts (Long et al., 2024; Xu et al., 2025). Following this, we conduct the agent generation step in LbMAS. When generating an LLM agent there are two main factors that can be customized: the prompt and base LLM. We only consider role prompts and an agent generating agent (AG) is executed to randomly generate n expert prompts from domains related to query q based on the expert generation instruction I .

$$\{(E_1, D_1), \dots, (E_n, D_n)\} = AG(q, I) \quad (1)$$

Each expert prompt is a tuple (E_i, D_i) where E_i is the expert's identity and D_i a short description of its expertise, by which expert agents are then generated. For the base model, the study (Zhang et al., 2025c) shows that MASs using different LLMs rather than a single one have better performance

Algorithm 1: Algorithm of bMAS

Input: query q , agent group G , control unit $ConU$, maximum blackboard cycle round K , Blackboard contents B , solution extraction module $SolE$

Output: *Solution*

```
1  $t \leftarrow 1$ ; /* Current round */
2 while  $t \leq K$  do
3    $\{A_1, \dots, A_j\} \leftarrow ConU(q, B, G)$ ;
    /* the control unit selects agents from  $G$  */
4   for  $i \leftarrow 1$  to  $j$  do
5      $m_i \leftarrow Exec(A_i, B)$ ; /* execute agent to generate message */
6      $B \leftarrow B \cup m_i$ ; /* update blackboard contents */
7   end
8    $t \leftarrow t + 1$ 
9 end
10  $Solution \leftarrow SolE(B, G)$ ; /* get solution from the blackboard */
```

since model diversity can effectively compensate for individual model limitations while amplifying their strengths. We thus create a set of LLMs, and every time a new agent is created, it will randomly choose an LLM from the set as its base model.

3.2 The blackboard cycle

There are three modules involved in the blackboard cycle and we present them subsequently.

Agent group. The agent group in LbMAS contains a set of predefined agents in addition to query-specified agents. In the original blackboard systems messages are generated and transferred following rules and logical inferences, which in LbMAS can be realized by the predefined LLM agents that are decider, planner, critic, conflict-resolver, and cleaner. The decider agent assesses whether messages on the blackboard are enough to yield the final solution; and if so, it will give the solution and the blackboard cycle stops. The planner agent makes plans to solve the query, and particularly would decompose the query into smaller tasks when it is complex. The critic agent points out errors in messages on the blackboard, which may come from hallucinations of LLMs, and forces relevant agents to rethink their output. The conflict-resolver agent detects contradictions among messages on the blackboard and then name the agents

that have conflicting messages. These called agents would discuss about the conflicts, which can be conducted in the private space of the blackboard. After discussion, each involved agent generates a new message and writes it in the public blackboard space. The cleaner agent is typically designed to ensure the quality of messages on the blackboard. It detects useless or redundant messages and removes them. The cleaning is necessary as effective management of content of the blackboard can facilitate meaningful communication among agents and at the same time reduce token consumption.

Blackboard. The blackboard public space serves as a shared memory in which each LLM agent can read and write, enabling seamless communication and collaboration. Agents can access and utilize messages stored in the public space, allowing them to incrementally build upon each other's results. Unlike MetaGPT (Hong et al., 2024) that uses a shared memory pool as an assistant for knowledge storage, in LbMAS agents communicate solely through the blackboard without any direct contact; in other words the blackboard is responsible for all agent communication and agents decide on their own what to write on the blackboard.

With the availability of blackboard, we remove the memory module commonly existing in LLM-based agents. The main function of the memory module is to ensure that the agent remembers the context and makes consistent output. In LbMAS agents' messages are all stored on the blackboard and the memory modules become unnecessary. This way agents can ensure the consistency of each message while the overall length of prompts for all agents can be reduced, enabling a MAS to have more discussions under the token constraint.

Control unit. The control unit $ConU$ manages the flow of information and the execution of tasks among agents, pursuing that right actions are taken according to current messages on the blackboard. Again we use an LLM to fulfill these functions. When the cycle starts, the control unit iteratively selects agents based on query q , current messages B on the blackboard and agent abilities D_1, \dots, D_n .

$$\{E_{i_1}, E_{i_2}, \dots, E_{i_j}\} = ConU(q, B, \{D_1, \dots, D_n\}) \quad (2)$$

When the maximum iteration number is reached or the decider agent is selected and gives the solution, the cycle comes to an end. Normally there are two ways for a MAS to yield the final answer,

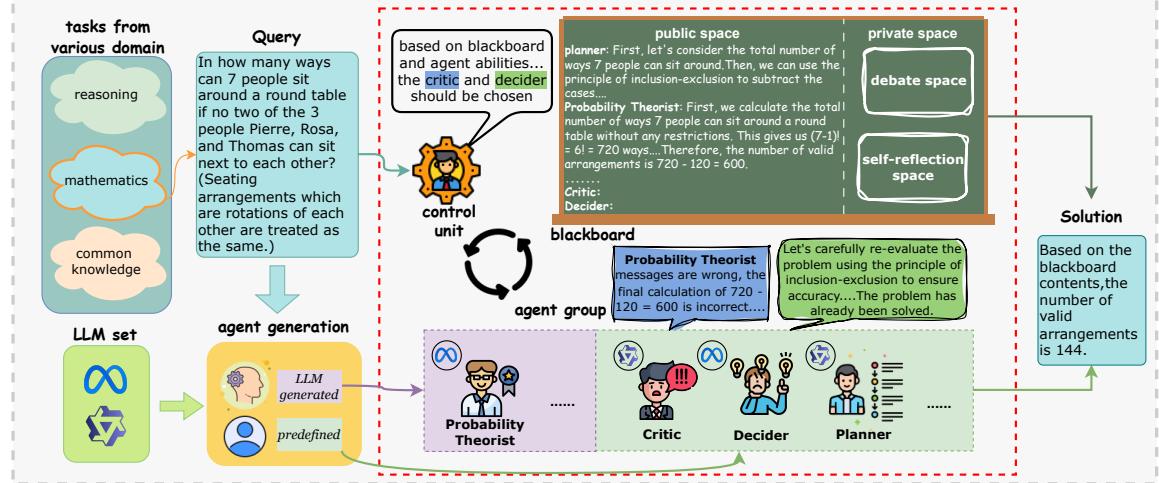


Figure 1: The general framework of bMAS is shown in the red dashed box. The illustration of LbMAS is in the green dashed box

depending on a specific agent to make decisions, or conducting a majority vote in which all relevant agents can participate. We have implemented both: the decider will give the answer when possible, and when the cycle stops all agents are asked to present an answer based on current content of the blackboard. The cumulative similarity is calculated for each answer relative to the others, denoted as $V(a_i) = \sum_{j=1, j \neq i}^N \text{sim}(a_i, a_j)$ where N is the number of answers obtained. The one with the highest cumulative similarity is then chosen as the final answer, i.e., $\text{argmax}_{a_i} V(a_i)$.

4 Experiments

4.1 Implementation Setup

Tasks and benchmarks. We evaluate LbMAS on six benchmarks: the knowledge dataset MMLU (Hendrycks et al., 2021a), reasoning datasets ARC-Challenge (Clark et al., 2018), GPQA-Diamond (Rein et al., 2024) and BBH-dateunderstand (Suzgun et al., 2023), and mathematical datasets MATH (Hendrycks et al., 2021b) and GSM8K (Cobbe et al., 2021). For MMLU, we randomly select 20 questions from each category and obtain a total of 1140 questions. For MATH, we use a subset of 500 questions randomly selected by OpenAI².

Baselines. We compare our method with Chain-of-Thought (CoT) (Wei et al., 2022), Major-Vote (junyou li et al., 2024), static multi-agent systems including Chateval (Chan et al.), Exchange-of-

Thought (EoT) (Yin et al., 2023), and MultiPersona (MP) (Liang et al., 2024), and autonomous multi-agent systems including GPTSwarm (Zhuge et al., 2024), AFlow (Zhang et al., 2025d), and MaAS (Zhang et al., 2025a).

Implementation details. Our main experimental results were obtained using two open-source LLMs with varying sizes and capacities: Llama-3.1-70b-Instruct (Dubey et al., 2024) and Qwen-2.5-72b-Instruct (Qwen et al., 2025). To eliminate the impact of model differences, when creating a new agent we randomly select one of the two LLMs as its base model. For the sake of fairness, all MASs in comparison have adopted this LLM selection way. All models are accessed via APIs with the temperature set to 0.7. We set the maximum rounds of the blackboard cycle to 4.

4.2 Performance Analysis

The main results of the experiments are shown in Table 1. One can see that LbMAS obtained the best performance on most test benchmarks, outperforming CoT methods by an average of 4.33% and static methods by 5.02%. The two bottom lines of Table 1 compare the performance of two decision-making ways LbMAS adopts. The decider and majority vote way performed similarly on all benchmarks, indicating that in most cases the decider and other agents reached a consensus after discussion. As a matter of fact, the proportions of consensus reached by all agents on the MMLU, GPQA and MATH datasets are 89.8%, 52.5% and 29.4%, respectively.

Note that in mathematical problems, the CoT

²<https://huggingface.co/datasets/HuggingFaceH4/MATH-500>

Table 1: Comparing LbMAS with single-agent and static multi-agent systems. The best result are highlighted in bold and the runners-up are underlined.

Method	MMLU	ARC-Challenge	GPQA-Diamond	BBH	MATH	GSM8K	Avg.
Vanilla (Llama)	78.16	90.87	36.86	64.00	26.00	34.57	54.74
Vanilla (Qwen)	76.40	<u>89.50</u>	36.36	74.40	40.20	42.83	<u>59.94</u>
CoT (Llama)	81.31	92.74	35.35	81.20	62.20	94.84	74.60
CoT (Qwen)	84.82	92.74	44.94	<u>87.20</u>	76.40	94.46	80.09
Major-Vote	81.49	91.97	41.41	73.20	36.00	39.80	60.64
MultiPersona	81.84	90.35	44.44	85.60	65.80	93.85	76.98
Exchange-of-Thought	81.05	91.12	37.87	80.80	60.40	83.32	72.42
Chateval	84.21	93.25	<u>51.26</u>	90.00	70.20	94.46	80.56
<i>LbMAS</i>	85.35	<u>93.43</u>	54.04	90.00	<u>72.8</u>	94.46	81.68
<i>LbMAS-Majorvote</i>	85.17	93.51	54.04	90.00	<u>72.60</u>	94.61	<u>81.65</u>

Table 2: Comparing systems with a single base LLM in mathematical benchmarks. The best results are highlighted in bold.

Method	MATH	GSM8K
Vanilla (Llama)	26.0	34.57
CoT (Llama)	62.2	94.84
<i>LbMAS (Llama)</i>	58.4	92.57
Vanilla (Qwen)	40.2	42.83
CoT (Qwen)	76.4	94.46
<i>LbMAS (Qwen)</i>	78.6	96.05

method using Qwen LLM achieved the best performance. To further analyze, we conducted a comparison using one base model for all agents as shown in Table 2. Our system solely using Qwen outperformed the CoT method on the MATH and GSM8K benchmarks, whereas LbMAS with Llama as the only base model performed worse than CoT. The results indicate that the Llama model may not be as good as Qwen at handling mathematical problems that usually require multi-step reasoning and accurate calculations, and such a performance gap among LLMs can affect the overall performance of MASs.

We also counted the number of queries answered correctly by a single method, as shown in Figure 2. Our method uniquely solved the highest number of problems overall in the MMLU, GPQA and MATH datasets, while other methods including Vanilla are all able to uniquely solve problems, showing the respective potential of each method.

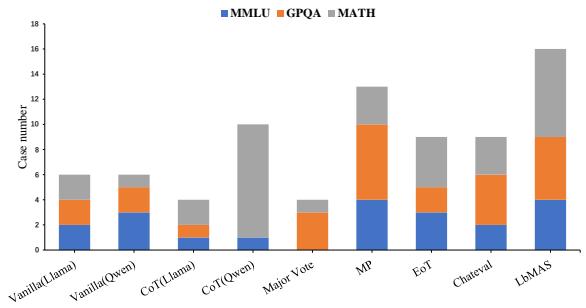


Figure 2: The number of queries answered correctly by a single method

4.3 Cost Analysis

We compared the token cost as well as the performance between LbMAS and the static and autonomous MAS baselines on the MATH benchmark, as shown in Table 3. One can see that LbMAS not only stood out in performance but also managed to cost the second lowest number of tokens. Note that the three autonomous MASs take a two-step strategy to search for an optimized workflow first before solving the problem, thus spend a lot more extra tokens. In addition, they require training data from related domains to perform searching and shall become useless when such data is unavailable. Our method on the other hand takes a dynamic adjusting strategy to select and execute agents on-the-fly according to the current blackboard messages in an iterative way until a consensus solution is obtained.

4.4 Case Study

We visualize the process of using LbMAS to solve four problems of different difficulty levels in Figure 3. When facing easy problems, LbMAS can

Table 3: Comparing the token cost and performance of bMAS, static MAS and autonomous MAS on the MATH dataset. The lowest total tokens and highest performances are highlighted in bold and the second ones underlined.

Method	Training		Inference		Overall			
	Prompt token	Completion token	Prompt token	Completion token	Total prompt	Total completion	Total token	Performance
EoT	-	-	5,475,154	2,853,618	5,475,154	2,853,618	8,328,772	60.40
MP	-	-	1,351,821	820,069	<u>1,351,821</u>	820,069	2,171,890	65.80
Chateval	-	-	3,338,508	2,113,312	<u>3,338,508</u>	2,113,312	5,451,820	70.20
GPTSwarm	445,674	1,961,010	559,728	2,483,013	1,005,402	4,444,023	5,449,425	67.25
AFlow	6,643,444	7,977,945	1,562,388	513,255	8,205,832	8,491,200	16,697,032	69.25
MaAS	4,150,172	2,700,925	3,773,460	2,388,101	7,923,632	5,089,026	13,012,658	<u>70.77</u>
<i>LbMAS</i>	-	-	3,352,594	1,013,015	3,672,222	<u>1,049,267</u>	<u>4,721,489</u>	72.60

stop after one round of execution, while for hard problems LbMAS would have agents engage in multiple rounds of discussion to obtain the final answer. This shows that our method can generate suitable multi-agent collaborations for diverse problems. In addition, agents can effectively play their roles, as shown in query a) where the cleaner detects and removes redundant messages and in query c) where the conflict resolver identifies agents with conflicting messages and asks them to have a discussion.

4.5 Framework Study

Round analysis. We analyze the impact of the maximum number of blackboard cycle rounds on the system performance, as illustrated in Table 4. For the three datasets MMLU, GPQA and MATH, the average rounds are 2.88, 3.05, and 3.29 respectively. This is why we set the maximum round number to 4. Additionally, we find that easy problems such as those in MMLU need lower numbers of rounds whereas hard problems such as GPQA and MATH require more rounds of execution.

Ablation study. We performed an ablation study on the control unit to test the autonomy of LLM agents in LbMAS. The agents are instructed to determine on their own whether to respond to the current content on the blackboard without the control unit. The results in Table 5 show that the control unit can significantly reduce the token cost of LbMAS with a slight turbulence in performance. Competitive results indicate that LLM agents have a certain degree of autonomy and that the control unit can select appropriate agents in each round. Further, the bottom line in Table 5 shows that the performance declines on three datasets if the cleaner agent is asked to mark the redundant messages rather than directly remove them, indicating

Table 4: Comparing the average execution rounds of the blackboard cycle and performances (in parentheses) under different maximum round settings. The best performances are highlighted in bold.

Max Round	MMLU	GPQA	MATH
2	2.00 (84.28)	2.00 (52.02)	2.00 (72.80)
4	2.88 (85.17)	3.29 (54.04)	3.04 (72.60)
6	3.06 (84.21)	3.44 (53.03)	3.34 (71.60)
8	3.15 (84.21)	3.46 (51.01)	3.35 (74.80)

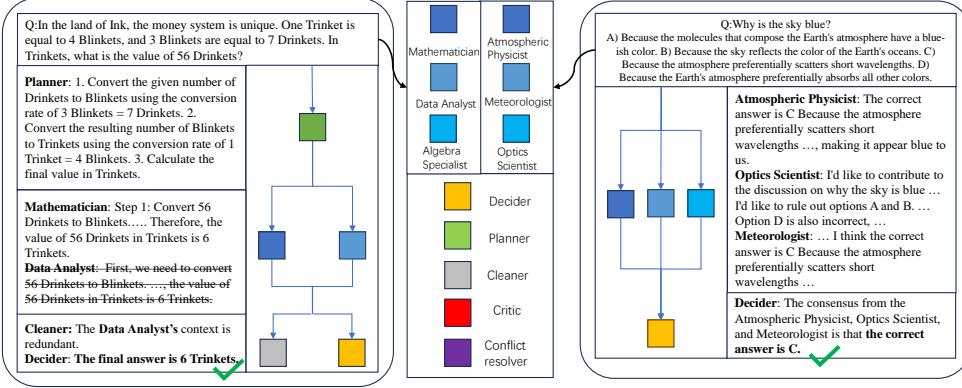
Table 5: Ablation study of LbMAS. The best performances are highlighted in bold.

Method	MMLU	GPQA	MATH
LbMAS(full)	performance	85.17	54.04
	token cost	5,074,931	2,984,204
w/o control unit	performance	85.52	53.53
	token cost	18,829,644	13,325,826
w/o message removal	performance	84.38	53.53
			71.00

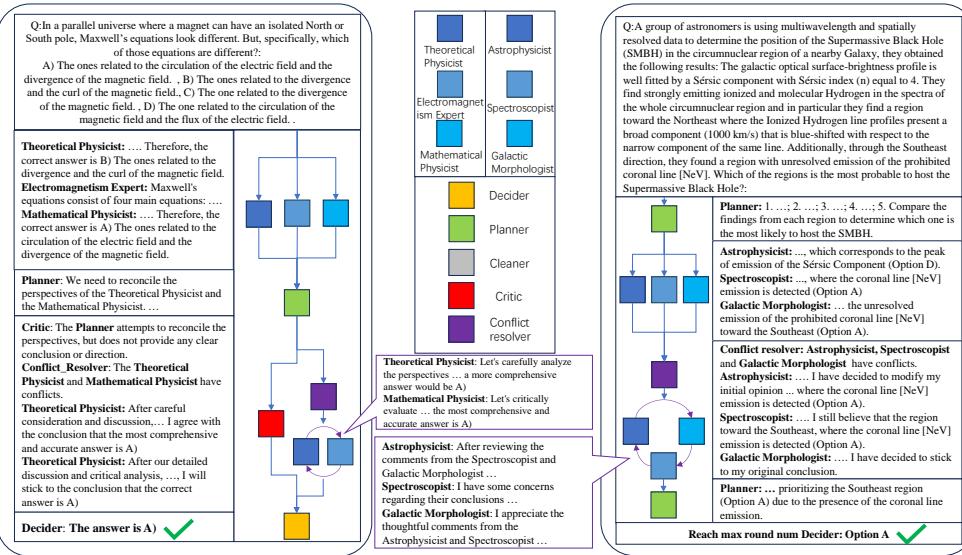
the necessity of managing the content of blackboard.

5 Discussion

Advancement of the blackboard-based LLM multi-agent framework. We summary the advancement of LbMAS as follows. Firstly, our framework is capable of dynamically selecting agents, making it a highly generalized framework that can solve multi-domain problems without the need for additional prompt modifications or collaboration mechanism optimization. Secondly, our framework enables a high degree of autonomy in both agent generation and interaction, and agents with various roles can complete their tasks properly. Thirdly, our framework has high scalability as there are no constraints about the agent group, making it easy to add new agents and functions whenever



Easy cases a) on the left and b) on the right



Hard cases c) on the left and d) on the right

Figure 3: Case study of LbMAS.

they are needed. Meanwhile, as a medium for agent interaction, the blackboard can meet various kinds of interaction needs including public vs. private space, flat vs. hierarchical structure, and so on.

Why does the blackboard-based LLM multi-agent system work? Compared with the SOTA static and autonomous MAS methods, we analyze that the performance improvement and cost saving of LbMAS can come from two aspects: (1) a shared memory pool by the blackboard, which enables more comprehensive information exchange among agents, and (2) dynamically selecting agents from the agent group based on the current blackboard, which can ensure that the selected agents are suitable for handling existing messages and thus avoid unnecessary agent executions. Particularly, unlike recent autonomous MASs that search for optimal workflows first before running them to solve

problems, our system adapts the agent selection and execution on-the-fly alongside the changing on the blackboard. Although in the end the whole execution order may not be optimal, we do not need a supervised training step that requires extra data and token cost.

Conclusions. In this paper, we propose the blackboard-based LLM multi-agent framework, which integrates flexibility and autonomy of the traditional blackboard architecture into nowaday MASs, aiming to enable dynamic and efficient problem-solving when workflows are unknown. We develop the first implementation and conduct experiments on six knowledge, reasoning and mathematical datasets. The results have demonstrated the cost-effectiveness of the system and we believe that our proposal shall contribute to the development of fully automated and self-organizing MASs.

Limitations

Our work is the first attempt to combine blackboard architecture with LLM multi-agent systems, and there are limitations in the implementation. Firstly, there are relatively fewer predefined types of agents in the agent group. We implemented agents with roles, and commonly used functions such as code writing and tool usage are not yet considered. In addition, experiment benchmarks are limited, and conducting experiments on a wider range of datasets can explore the potential of the framework, which will be one of our future works. Secondly, the agent generation module is relatively simple without considering enough factors. Results on the mathematical datasets show the necessity of selecting suitable LLMs, and works like MASRouter (Yue et al., 2025) that can optimize agent prompts and base LLMs in the agent generation process should be applied to our system. Thirdly, exploration of the blackboard is not adequate. Comparative experiments should be conducted to explore the differences between shared memory pool and agent memory module, as memory of LLM agent is the key component to support agent-environment interactions (Zhang et al., 2024b).

References

- Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2025. ResearchAgent: Iterative research idea generation over scientific literature with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6709–6738, Albuquerque, New Mexico. Association for Computational Linguistics.
- Marc Cavazza, Steven J Mead, Alexander I Strachan, and Alex Whittaker. 2001. A blackboard system for interpreting agent messages. In *Proceedings GameOn 2000: International Conference on Intelligent Games & Simulation*.
- Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. Chateval: Towards better llm-based evaluators through multi-agent debate, 2023. URL <https://arxiv.org/abs/2308.07201>, 3.
- Huaben Chen, Wenkang Ji, Lufeng Xu, and Shiyu Zhao. 2025. Multi-Agent Consensus Seeking via Large Language Models. *arXiv preprint*. ArXiv:2310.20151 [cs].
- Junzhe Chen, Xuming Hu, Shuodi Liu, Shiyu Huang, Wei-Wei Tu, Zhaofeng He, and Lijie Wen. 2024a. LLMArena: Assessing Capabilities of Large Language Models in Dynamic Multi-Agent Environments. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13055–13077, Bangkok, Thailand. Association for Computational Linguistics.
- Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2024b. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.
- Daniel D Corkill. 1991. Blackboard systems.
- Ayushman Das, Shu-Ching Chen, Mei-Ling Shyu, and Saad Sadiq. 2023. Enabling synergistic knowledge sharing and reasoning in large language models with collaborative multi-agents. In *2023 IEEE 9th International Conference on Collaboration and Internet Computing (CIC)*, pages 92–98.
- J. Dong, S. Chen, and J.-J. Jeng. 2005. Event-based blackboard architecture for multi-agent systems. In *International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II*, volume 2, pages 379–384 Vol. 2.
- Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multi-agent debate. In *Forty-first International Conference on Machine Learning*.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.
- K.S. Ettabaa, I.R. Farah, B. Solaiman, and M.B. Ahmed. 2006. Distributed blackboard architecture for multispectral image interpretation based on multi-agent system. In *2006 2nd International Conference on Information & Communication Technologies*, volume 2, pages 3070–3075.

- Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.
- Barbara Hayes-Roth. 1985. A blackboard architecture for control. *Artificial Intelligence*, 26(3):251–321.
- Zhitao He, Pengfei Cao, Yubo Chen, Kang Liu, Ruopeng Li, Mengshu Sun, and Jun Zhao. 2023. LEGO: A multi-agent collaborative framework with role-playing and iterative feedback for causality explanation generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9142–9163, Singapore. Association for Computational Linguistics.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.
- Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.
- Shengran Hu, Cong Lu, and Jeff Clune. 2025. Automated design of agentic systems. In *The Thirteenth International Conference on Learning Representations*.
- Md. Ashraful Islam, Mohammed Eunus Ali, and Md Rizwan Parvez. 2024. MapCoder: Multi-agent code generation for competitive problem solving. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4912–4944, Bangkok, Thailand. Association for Computational Linguistics.
- Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.
- junyou li, Qin Zhang, Yangbin Yu, QIANG FU, and De-heng Ye. 2024. More agents is all you need. *Transactions on Machine Learning Research*.
- Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan A, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. In *The Twelfth International Conference on Learning Representations*.
- Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbulin, and Bernard Ghanem. 2023a. Camel: Communicative agents for "mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008.
- Huao Li, Yu Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Charles Lewis, and Katia Sycara. 2023b. Theory of mind for multi-agent collaboration via large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 180–192, Singapore. Association for Computational Linguistics.
- Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2024. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17889–17904, Miami, Florida, USA. Association for Computational Linguistics.
- Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2024a. A dynamic llm-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*.
- Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2024b. A dynamic LLM-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*.
- Do Xuan Long, Duong Ngoc Yen, Anh Tuan Luu, Kenji Kawaguchi, Min-Yen Kan, and Nancy F. Chen. 2024. Multi-expert prompting improves reliability, safety and usefulness of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20370–20401, Miami, Florida, USA. Association for Computational Linguistics.
- Daniel McDuff, Mike Schaeckermann, Tao Tu, Anil Palepu, Amy Wang, Jake Garrison, Karan Singhal, Yash Sharma, Shekoofeh Azizi, Kavita Kulkarni, and 1 others. 2025. Towards accurate differential diagnosis with large language models. *Nature*, pages 1–7.
- Hana Munira Muhd Mukhtar, Husna Sarirah Husin, Suriana Ismail, Azizah Rahmat, and Roslan Ismail. 2025. Advancing Blackboard Framework: Responsive Trigger Mechanisms and Dynamic Feasibility Conditions for Optimal Knowledge Source Selection. In *2025 19th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, pages 1–5.

- H Penny Nii. 1986. *Blackboard systems*. Knowledge Systems Laboratory, Depts. of Medical and Computer Science
- Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*.
- Shuofei Qiao, Ningyu Zhang, Runnan Fang, Yujie Luo, Wangchunshu Zhou, Yuchen Jiang, Chengfei Lv, and Huajun Chen. 2024. *AutoAct: Automatic agent learning from scratch for QA via self-planning*. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3003–3021, Bangkok, Thailand. Association for Computational Linguistics.
- Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. *Qwen2.5 technical report*. Preprint, arXiv:2412.15115.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. *GPQA: A graduate-level google-proof q&a benchmark*. In *First Conference on Language Modeling*.
- Darya Rudenko and Arkady Borisov. 2008. Blackboard architecture for product life cycle stage definition. In *Proceedings of the 14th International Conference on Soft Computing (Mendel 2008)*, pages 252–257.
- Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Zicheng Liu, and Emad Barsoum. 2025a. *Agent laboratory: Using llm agents as research assistants*. Preprint, arXiv:2501.04227.
- Samuel Schmidgall, Rojin Ziae, Carl William Harris, Ji Woong Kim, Eduardo Pontes Reis, Jeffrey K Jopling, and Michael Moor. 2025b. *Agentclinic: a multimodal agent benchmark to evaluate AI in simulated clinical environments*.
- Yu Shang, Yu Li, Keyu Zhao, Likai Ma, Jiahe Liu, Fengli Xu, and Yong Li. 2025. *Agentsquare: Automatic LLM agent search in modular design space*. In *The Thirteenth International Conference on Learning Representations*.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. 2023. *Reflexion: language agents with verbal reinforcement learning*. In *Thirty-seventh Conference on Neural Information Processing Systems*.
- Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024. *Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph*. In *The Twelfth International Conference on Learning Representations*.
- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. *Challenging BIG-bench tasks and whether chain-of-thought can solve them*. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.
- Lukasz Szymanski, Bartłomiej Śnieżyński, and Bipin Indurkha. 2018. *Multi-agent blackboard architecture for supporting legal decision making*. *Computer Science*, 19:459.
- Yashar Talebirad and Amirhossein Nadiri. 2023. Multi-agent collaboration: Harnessing the power of intelligent llm agents. *arXiv preprint arXiv:2306.03314*.
- Xunzhu Tang, Kisub Kim, Yewei Song, Cedric Lothritz, Bei Li, Saad Ezzini, Haoye Tian, Jacques Klein, and Tegawendé F. Bissyandé. 2024. *CodeAgent: Autonomous communicative agents for code review*. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11279–11313, Miami, Florida, USA. Association for Computational Linguistics.
- Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O’Sullivan, and Hoang D Nguyen. 2025. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*.
- Tao Tu, Mike Schaeckermann, Anil Palepu, Khaled Saab, Jan Freyberg, Ryutaro Tanno, Amy Wang, Brenna Li, Mohamed Amin, Yong Cheng, and 1 others. 2025. Towards conversational diagnostic artificial intelligence. *Nature*, pages 1–9.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. *Chain of thought prompting elicits reasoning in large language models*. In *Advances in Neural Information Processing Systems*.
- Benfeng Xu, An Yang, Junyang Lin, Quan Wang, Chang Zhou, Yongdong Zhang, and Zhendong Mao. 2025. *Expertprompting: Instructing large language models to be distinguished experts*. Preprint, arXiv:2305.14688.
- Lin Xu, Zhiyuan Hu, Daquan Zhou, Hongyu Ren, Zhen Dong, Kurt Keutzer, See-Kiong Ng, and Jiashi Feng. 2024. *MAGIC: Investigation of large language model powered multi-agent in cognition, adaptability, rationality and collaboration*. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7315–7332, Miami, Florida, USA. Association for Computational Linguistics.
- John Yang, Carlos Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. *Swe-agent: Agent-computer interfaces*

- enable automated software engineering. *Advances in Neural Information Processing Systems*, 37:50528–50652.
- Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuanjing Huang, and Xipeng Qiu. 2023. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.
- Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyi Qi. 2025. *Masrouter: Learning to route llms for multi-agent systems*. Preprint, arXiv:2502.11133.
- Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. 2025a. Multi-agent architecture search via agentic supernet. *arXiv preprint arXiv:2502.04180*.
- Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. 2025b. Cut the crap: An economical communication pipeline for LLM-based multi-agent systems. In *The Thirteenth International Conference on Learning Representations*.
- Hangfan Zhang, Zhiyao Cui, Xinrun Wang, Qiaosheng Zhang, Zhen Wang, Dinghao Wu, and Shuyue Hu. 2025c. If Multi-Agent Debate is the Answer, What is the Question? *arXiv preprint*. ArXiv:2502.08788 [cs].
- Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. 2025d. AFlow: Automating agentic workflow generation. In *The Thirteenth International Conference on Learning Representations*.
- Jintian Zhang, Xin Xu, Ningyu Zhang, Ruibo Liu, Bryan Hooi, and Shumin Deng. 2024a. Exploring collaboration mechanisms for LLM agents: A social psychology view. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14544–14607, Bangkok, Thailand. Association for Computational Linguistics.
- Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. 2023. Cumulative reasoning with large language models. *arXiv preprint arXiv:2308.04371*.
- Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2024b. A survey on the memory mechanism of large language model based agents. Preprint, arXiv:2404.13501.
- Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbulin, and Jürgen Schmidhuber. 2024. GPTSwarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*.

A Experiment details

A.1 Datasets

In Table 6, we provide an introduction to the datasets used in our experiments, including problem type, answer format, and test samples. Note that in MATH dataset the answer format is varied, including number, word and expression. Accuracy is used as the evaluation metric in our experiments. We require the intelligent agent to output answers in "boxedanswer" format for the convenience of extracting answers. For datasets with multiple-choice answer format, accuracy is calculated by checking if the option extracted from the response matches the correct answer. For other format we perform a match with the ground-truth answer.

A.2 Baselines

In this section, we provide a description of the configurations for baseline MAS methods. For static MAS MultiPersona, Exchange-of-Thought, and Chateval, we utilize the implementation from(Zhang et al., 2025c). Three autonomous MAS methods are implemented in accordance with the original implementation(Zhang et al., 2025d; Zhuge et al., 2024; Zhang et al., 2025a).

A.3 Prompts

In this section, we present the prompts for each module. The Agent generation prompt is based on Multi-expert Prompting(Long et al., 2024). The system prompt is unique to every predefined agent and includes the query and the agent’s role. Generated experts use the roles generated by the agent generation module.

B Additional experiments

Table 7 shows the performance comparison of LbMAS based on a single LLM and two LLMs. Generally, LbMAS with two LLMs outperforms the average of a single LLM. In GPQA-Diamond and MMLU datasets LbMAS has the best performance. Although experimental results have proven the generalization LbMAS, we can see that in relatively easy dataset MMLU, the performance improvement brought about by the LbMAS is limited. It is advisable to test LbMAS on more types of tasks.

Agent Prompt

System: You are {role_name} cooperating with other agents to solve the problem. The problem is:{question}.

There is a blackboard that everyone of you can read or write messages.

Decider: If you think the messages on the blackboard enough to get the final answer then You should output the final answer with your answer in the form {{the final answer is boxed[answer]}}, at the end of your response. otherwise you need other agents provide more information then say {{\"continue, waiting for more information\"}} and wait other agent giving new factors. do not output other information.

Planner: Generate plans to solve the original problem based on blackboard contents. Strictly follow the json format as follows: {{\"[problem]\":string //describe the problem, \"[planning]\":string //was the solving plan of the problem}}, If there already have plan or problem is simple enough to solve then say \\{{\"there is no need to decompose tasks, waiting for more information\"}}\\|. Do not solve the task.

Cleaner: If you think there are messages on the blackboard useless or redundant, you should output useless messages and your explanation. your output should follow the json format follow the form: {{\"clean list\":[{{\"useless message\":string //write useless message exactly, \"explanation\":string //was your explanation why the message is useless or redundant}}]}}. If you think there are no useless messages then you should write {{\"no useless messages, waiting for more information\"}} and wait for other agents to provide more information.

Critic: If you think the messages on the blackboard are wrong or misleading, your output should Strictly follow the json format as follows: {{\"critic list\":[{{\"wrong message\":string //write whose message and which message is wrong, \"explanation\":string //was your explanation why the message is wrong}}]}}. Otherwise you think there are no wrong messages then you should write {{\"no problem, waiting for more information\"}} and wait for other agents to provide more information.

Conflict_Resolver: If you think other agents' messages on the blackboard have conflicts, you should output all conflict agents and their messages. Output strictly follow the json format as follows: \n{{\"conflict list\":[{{\"agent\":string //was the name of conflict agent, \"message\":string //was the conflict message of agent on the blackboard}}]}}\n. Otherwise you think there are no conflicts then you should write {{\"no conflicts, waiting for more information\"}}.Do not output other information.

Generated Expert: You are an excellent {role} described as {roles_description}. Based on your expert knowledge and contents currently on the blackboard, solve the problem, output your ideas and information you want to write on the blackboard. It's not necessary to fully agree with viewpoint on the blackboard. Your output should strictly follow the json form:\n \\{{\"output\":\"\"}}.

Figure 4: Agent prompt

Agent Generation Prompt

You are provided a question. Give me a list of 1 to 3 expert roles that most helpful in solving question. Question: {question}. Only give me the answer as a dictionary of roles in the Python programming format with a short description for each role. Strictly follow the answer format below: \nAnswer: {{\"[role name 1]\": \"[description 1]\", \"[role name 2]\": \"[description 2]\", \"[role name 3]\": \"[description 3]\"}}

Figure 5: Agent-generation prompt

Control Unit Prompt

Your task is to schedule other agents to cooperate and solve the given problem. The agent names and descriptions are listed below:\n{role_list}. The given problem is:{question}. Agents are sharing information on the blackboard. Based on the contents existed on the blackboard, you need to choose suitable agents from agent list to write on the blackboard. Remember Output the agent names in the json form: \"{{\"chosen agents\":[list of agent name]}}}

Figure 6: Control unit prompt

Table 6: statistics of datasets utilized in our experiment.

DATASET	TYPE	ANSWER FORMAT	TEST
MMLU	General Knowledge	Multi-choice	1140
ARC-Challenge	Scientific Reasoning	Multi-choice	1172
GPQA-Diamond	Scientific Reasoning	Multi-choice	198
Date-Understanding	Symbolic Reasoning	Multi-choice	250
MATH	Mathematical Reasoning	Uncertain	500
GSM8K	Mathematical Reasoning	Number	1319

Table 7: Comparing systems with a single base LLM.

Method	MMLU	ARC-Challenge	GPQA-Diamond	BBH
CoT (Llama)	81.31	92.74	35.35	81.20
CoT (Qwen)	84.82	92.74	44.94	87.20
LbMAS (Llama)	81.22	91.72	50.50	90.00
LbMAS (Qwen)	84.12	94.11	47.47	87.60
- average	82.67	92.91	48.98	88.80
LbMAS	85.35	93.43	54.04	90.00

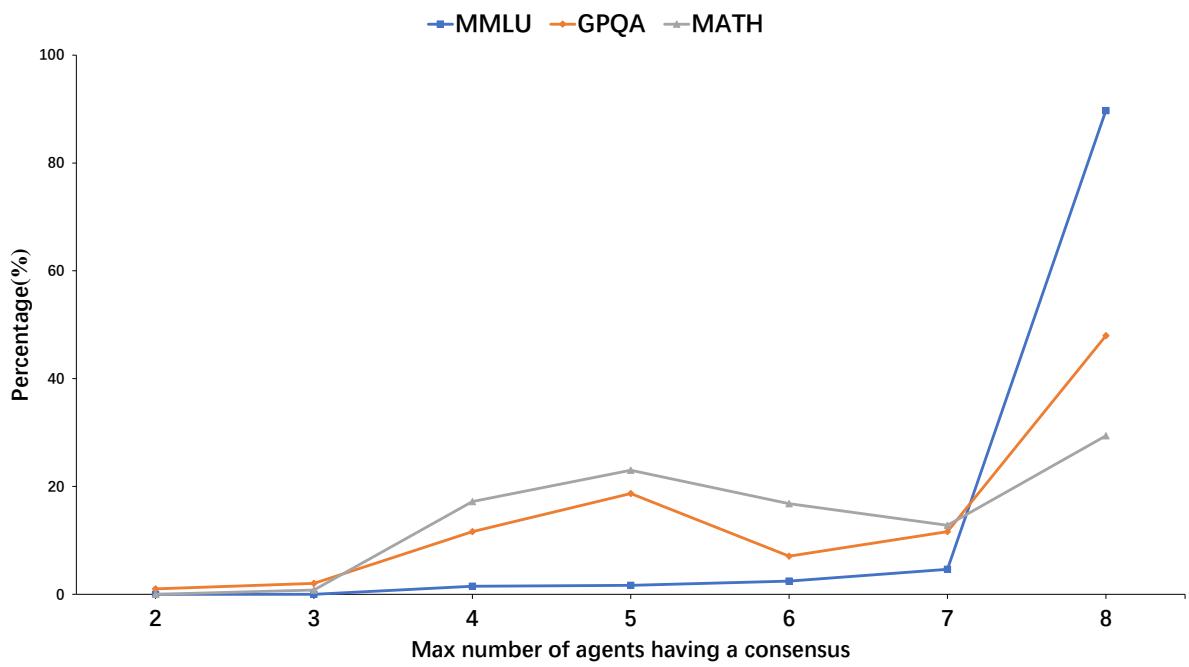


Figure 7: Max number of agents having a consensus in three datasets.