DOI: 10.2478/ijasitels-2025-0015

Emergent Strategies in Multi-Agent Systems

Vlad OLEKSIK[1], Cantemir MIHU[1], Antoniu PITIC[1]
[1]Computer Science and Electrical and Electronics Engineering Department, Faculty of Engineering, "Lucian Blaga" University of Sibiu, Romania
{vladandrei.oleksik, cantemir.mihu, antoniu.pitic} @ulbsibiu.ro

Abstract

This article aims to present an overview of the emerging protocols and strategies used for achieving scalability of performance- and efficiency-centric multi-agent systems as this field is rapidly developing and focus is shifting towards architectures employing multiple, tool-enabled models that need to both split tasks, working concurrently, and follow step-based problem-solving strategies, while maintaining effective communication throughout the system.

Keywords: Multi-agent system, memory, agentic workflow, evaluation, communication, dispatching.

1. Introduction

A significant part of today's development in terms of machine learning applications comes from the recent waves of breakthroughs in the field of large language model (LLM)-based systems. The emergence of LLMs has started with the perfection of the transformer with the attention mechanism [1], a component that represents the functional unit of a system that enables 'talk' between different features of data streams situated at a significant distance away.

However, as information recollection and filtering became a problem in the context of large time costs for cold-start finetuning, the idea of delegating long-term memory to a different service resulted in the introduction of the concept of retrieval-augmented generation, employing embedding models to point the 'attention' of the model towards a specific subset of given data and introducing this data straight into the context window of the model to obtain much improved results. Despite this practice being rapidly adoptable, multiple different paradigms for data storage exist today, from centralized vector stores to distributed constructs.

Moreover, the finite number of rounds of transformations, as well as some intrinsic properties of language involved in some problem-solving steps led to a problem as most of the computation essential to solving a problem would be focused into just several predictions [2], which led to the problem of task planning, with different 'reasoning' schemes being introduced to encourage more generation in order to enable the model to use as many steps as necessary to reach the desired output.

In this context, the lingering limitations of single models marked, in part, by the scarcity of new and high-quality training data have opened the door to the use of agentic paradigms, in which a model is given access to tools for information processing and retrieval and imposed planning paradigms to solve a problem. The emergence of multi-agent paradigms, characterized as a multi-level, multi-cluster network of agents exchanging and processing data, has thus furthered the horizons of high-performance, as well as high-efficiency systems for both reasoning and simpler processing tasks.

To date, there is little standardization in the protocols used in task delegation, evaluation, planning and memory management, [3] while specific use-cases are benefiting from particular architectural traits and more paradigms are still being introduced. Thus, the aim of this paper is to provide a representative overview of the different approaches and paradigms present in modern LLM systems and their applications, as well as a comparison between the performance of different approaches for benchmarks representing various high-applicability tasks.

2. Theoretical concepts

This section outlines the main sources of variation in multi-agent system design, both from a technical perspective and in terms of system integration.

As Yan et. al. presented in their survey [4] of such systems, with a highlight, among others, on communication, specific approaches emerge as solutions for the previously described problems. Yan et. al. challenges the previous classification of design "patterns" into *agent-specific* and *architectural* strategies, highlighting the need for a paradigm to address the highly varying structural and functional traits of architectures.

As described in this paper, along with studies such as the one published by Han et. al. [3], some of the most relevant aspects to consider in a multi-agent LLM system architecture and implementation are task delegation, communication, planning and tooling, and memory management. Each of these will be briefly discussed in the following subsections.

2.1. Task delegation

With the introduction of multiple LLMs in an agentic workflow, one of the first issues that arises is the topology to be used for creating and handling these agents. Some of the approaches represent a variation on the hierarchical model, where each agent can generally spawn and manage other agents, tasked with solving a subproblem of the current goal, while others resemble a horizontal, clustered structure, where models associate based on certain criteria to solve problems and shift these themes over time.

2.1.1. Hierarchical structures

This first approach resumes to the protocol of an agent being able to add several others at each point, up to a certain depth, each one being tasked with a subdivision of the task of the parent agent.

Thus, higher-level agents will perform management and supervision tasks, while lower-level ones will be relied upon for detailed, stepwise problem-solving actions [4]. As a characteristic of this strategy, when working with specific tasks requiring various, well-defined subtasks and/or roles, multiple different agent types can be instantiated to fit the problem with high flexibility and very little friction (Fig. 1).

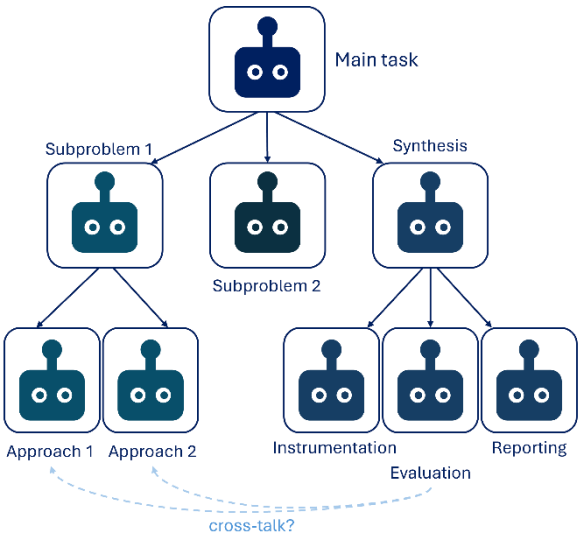

Fig. 1. The hierarchical (compartmentalized) model for multi-agent systems.

This is used in many versatile frameworks, however, the main limitation of this scheme is the lack of flexible and scalable opportunities for communication among agents, which affects the unitary structure of results through compartmentalization.

The Mixture-of-Experts topology [5] is an example of such a hierarchy, encountered with a model designated to decide what models (called “experts”) to dispatch for each input.

2.1.2. Homogeneous and clustered structures

As an alternative to using the hierarchy as a state and structure handling tool, many less hierarchical task delegation topologies exist, the main common element of which is the existence of a horizontal component to communication – that is, the ability of an agent to refer and report to agents not directly overseeing its activity (Peer-to-Peer structures).

These approaches vary from a fully-horizontal (homogeneous) model, to more structured approaches, where, despite it being possible to communicate with any other model, subgroups (teams/clusters) can be distinguished among the agents for certain tasks (Fig. 2).

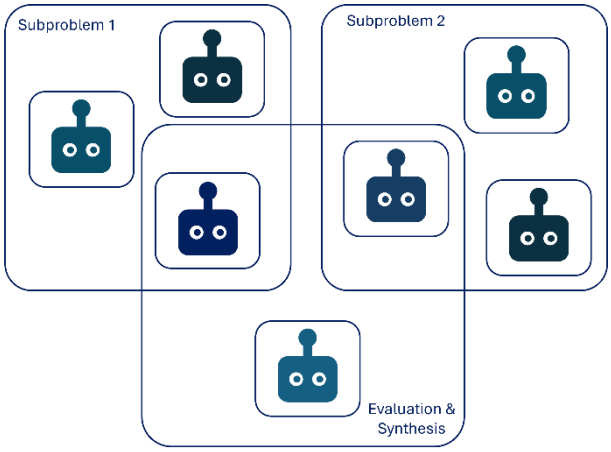

Fig. 2. Homogeneous models in multi-agent LLM systems

The downside of such systems is that it is often more difficult to exercise a given planning or structure for solving a given problem. Thus, elements of planning from the hierarchical model might be observed in some of the systems studied.

2.2. Memory management

Another mandatory feature of a performant multi-agent system is a form of information storage and retrieval mechanism. This needs to be implemented in order to enhance the relatively short working memory (context window) of a single agent, while being able to work in an environment of highly specific or quickly updating information. While most applications use a centralized data store – typically a vector store – some decentralized alternatives are better suited for scenarios involving rapid, localized updates that need to propagate only to a small subset of relevant agents.

2.2.1. Centralized stores

The idea of a vector database emerged in industry implementations for various similarity search algorithms, relying on previously well-known multi-dimensional data storage and retrieval structures [6]. In the context of multi-agent systems, it represents a database designed to retrieve data based on multi-dimensional vector key-value similarity, where all agents are free to read from and writes performed to this medium have global effect on the system (Fig. 3).

Integration of such a store into the agentic workflow is done as a tool/oracle that can be accessed (called) by an agent in a specific context. For the purpose of this study, a memory system is considered centralized not specifically from the point of view of the database implementation but based on the visibility of documents for specific agents.

Vector databases are a crucial component of most retrieval-augmented generation implementations and are, by far, the most widespread approach [7] for multi-agent systems, due to their high reliability and ease of integration into the vast majority of systems, regardless of their planning workflow and communication topology.

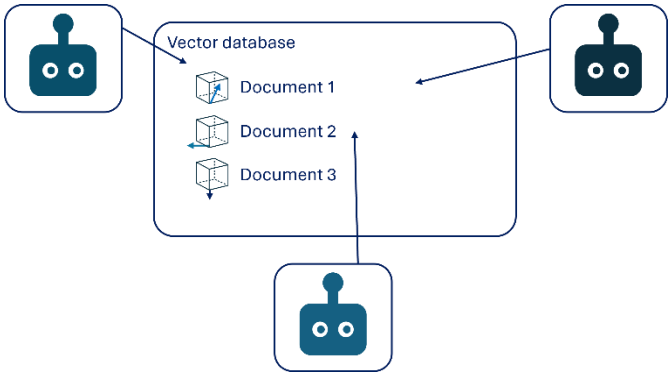

Fig. 3. Centralized memory approach

2.2.2. Decentralized topologies for memory management

Considering many of the multi-agent LLM systems have to handle growing complexity in order to produce quality results, overcoming limitations of particular agents, alternative memory management systems can not only help control the complexity when introducing many agents, but also limit the propagation of bad reasoning and low-quality documentation in tasks concerning only a subset of agents.

In this regard, a variety of approaches can be found across implementations in practice, with some topologies splitting the information stored among nodes/agent memories and submitting information to a propagation process for retrieval (Fig. 4).

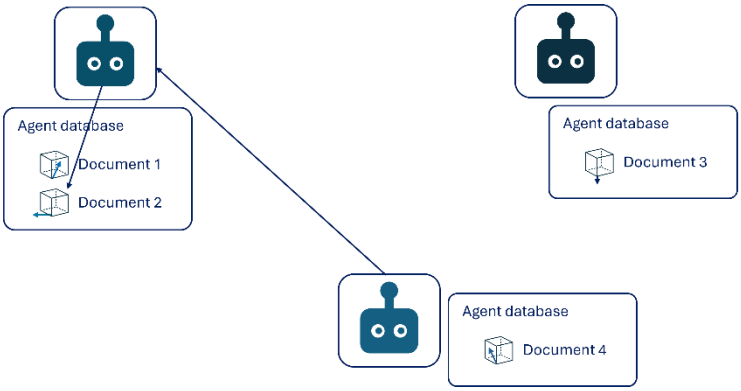

Fig. 4. The decentralized storage architecture

One emerging topology belonging to the class of decentralized memory management in multi-agent systems is an implementation relying on the concept of transactive memory. Stemming from a model in psychology proposed by Wegner [8], this structure translates in the current context to a specific workflow for updating the memory within the scope of each agent, as well as a mechanism for heuristically querying the system, relying on insight into which node is the most likely to hold particular pieces of information [9]. Still, while this aspect can enable scalability, few models, like the one proposed by Shang et. al. [10] have implemented this specific memory model to date.

2.3. Planning and tooling

Although not unique to multi-agent workflows, the concept of planning – that is, encouraging or enforcing a defined sequence of steps for each agent when generating a response – can enhance the output quality of individual models. Moreover, this approach can serve as a bridge from single-model systems to multi-agent settings by enforcing structured interaction through traditional interfaces and inter-agent communication alike.

A thorough overview of planning frameworks is presented by Aratchige and Ilmini [11] in a review encompassing an array of schemes for planning ranging from static (same flow of steps repeated for every stage of a problem) to dynamic (updating based on context, in designs such as AdaPlanner [12]) and from sequential (such as the ReAct framework [13], where steps are invariably executed one after another) to concurrent (such as the Tree of Thoughts framework [14], which introduces the possibility to handle multiple reasoning paths).

2.4. Communication and evaluation

While the problem of choosing a communication topology is partly influenced by and treated in the choice of task delegation framework, there are aspects specific to structuring of information passed among agents that significantly impact the functionality of a multi-agent system. As the complexity of the system increases with that of the tasks and the overlap between multiple agents begins to decrease in size, the methodology used to perform evaluation and to achieve consensus becomes of growing importance.

Among the frameworks already implemented, the treatment of the evaluation question is split between it being performed by the supervisor agent, shall one exist, and a more horizontal evaluation, performed by a subset of the other agents, according to a specific topology. It is of note that, while similar to the hierarchy problem described in 2.1., the approaches used for task delegation and evaluation are not necessarily correlated in the studied implementations.

In multi-agent systems – given their diverse and often complex topologies – it's important to distinguish between two notions: the local processes of voting or evaluation used to assess the quality and relevance of an individual agent's output, and the global consensus that must be maintained across the network to produce a coherent, unified result. In order to achieve the latter, techniques can be employed relying on expander graphs [15] to study and scalably enforce such convergence in the generation process.

Another issue regarding the communication in multi-agent LLM systems is that of the scope: while most implementations employ cooperative agents, there are notable examples of adversarial systems, such as the one proposed by Zhang and Eger [16]. The latter are often engaged in architectures where output generated by competitor

models is judged by a third agent, emulating feedback extrapolated from select human input.

A factor not extensively studied is that of the format of communication among agents. Many paradigms are in place for multi-agent systems, with Model Context Protocol (MCP) being the most used one in practice in the context of a single agent interacting with its available functional tools, while Agent-to-Agent (A2A) is among the most prominent protocols for handling inter-agent communication. While most frameworks employ plain-text communication or text-based tags for issuing signals or events, little is studied about the passing of embeddings from agent to agent as "paralinguistic" information. As a multi-agent system scales upwards, such information can serve as a mechanism to facilitate consensus by regulating the flow and scope of inter-agent communication.

3. Representative Multi-Agent Architectures

Recent advances in multi-agent LLM systems have led to a growing interest in frameworks to guide and integrate multiple LLM instances' cooperation or competition for solving complex tasks [3]. These systems seek to exploit specific emergent behaviors of distributed reasoning, division of labor, and interactive problem solving. In this context, recent research has focused mainly on defining scalable architectures, novel, efficient communication protocols, and structured memory mechanisms to improve the performance and interpretability of multi-agent LLM systems.

A detailed examination of the scalability problem forms the foundation of the MegaAgent architecture, proposed by Wang et al. [17]. This framework is designed to achieve relatively high efficiency on tasks of arbitrary complexity through a hierarchical team-based model. In MegaAgent, GPT-4o-based agents operate within a unified shared memory and employ a divide-and-conquer approach to task delegation. An important design choice is the prescriptive communication strategy: interactions occur exclusively along the vertical hierarchy or within localized groups. While this design enhances control and scalability, it can reduce the semantic cohesion of the group's collective reasoning. Despite these trade-offs, MegaAgent achieves competitive results with up to approximately 600 active agents, demonstrating one of the most scalable implementations reported to date.

A similar direction is explored in AgentVerse, introduced by Chen et al. [18], which represents a significant and methodically constructed contribution to the field. The authors distinguish between vertical and horizontal organizational structures, showing that a vertically layered arrangement benefits analytical tasks such as mathematical reasoning and software development, while a Peer-to-Peer structure performs better for decision-making and consultation scenarios. AgentVerse also contributes dedicated benchmarks for evaluating multi-agent collaboration patterns, establishing a valuable foundation for comparative research in this rapidly evolving domain.

In contrast to Q. Wang's approach, TalkHier, proposed by Zh. Wang and Moriyama [19], prescribes a rigorously defined hierarchical task organization coupled with a structured communication topology. The system integrates hierarchical evaluation mechanisms to assess intermediate outputs at multiple levels of the hierarchy. Also implemented using GPT-4o as the underlying model, TalkHier is claimed to outperform the selected baselines, highlighting the effectiveness of combining structured dialogue with systematic oversight. This reinforces the notion that well-balanced topologies and supervisory hierarchies can enhance the quality and coherence of distributed reasoning processes.

Based on current research, multi-agent systems are increasingly characterized through a diverse set of attributes drawn from multiple domains, including distributed computing, complex adaptive systems, organizational theory, and collective intelligence, traits largely dependent on the type of structure it uses. Table 1 captures the impact system topologies have on such systems' ability to coordinate, scale, adapt, and maintain robustness. Different system topologies — such as centralized, decentralized, or peer-to-peer structures — exhibit these qualities to differing degrees.

Table 1. Qualitative assessment of traits for the main system topologies

Attribute / Quality	Centralized	Hierarchical	Peer-to-Peer
Scalability	Medium	Good	Good
Coordination Overhead	Low	Medium	High
Fault Tolerance	Poor	Medium	Good
Communication Efficiency	Excellent	Good	Medium
Decision Latency	Low	Medium	High
Control Transparency	Excellent	Good	Poor
Specialization Support	Medium	Excellent	Good
Adaptability / Flexibility	Poor	Good	Excellent
Conflict Resolution	Excellent (via hub)	Good (via manager)	Poor
Robustness to Noise / Error	Poor	Medium	Good
Implementation Complexity	Low	Medium	High
Learning Convergence	Good	Good	Variable
Explainability / Auditability	Excellent	Good	Poor
Emergent Behavior Potential	Poor	Medium	Excellent

Regarding the use of decentralized memory in multi-agent architectures, Shang et al. [10] present a detailed implementation that integrates a shared, well-defined transactive memory topology governed by moderation-based communication. The system, as proposed, employs Gemini models of varying sizes and includes a thorough evaluation across multiple benchmarks. However, this framework lacks a structured evaluation component and does not support dynamic restructuring of the agent network during execution. Moreover, the authors do not employ heterogeneous agent scales (i.e., models of different sizes within a single system) nor maintain interpretability logs that trace the system's internal reasoning steps. These omissions point toward several promising directions for future research, including adaptive team composition, explainability mechanisms, and dynamic reconfiguration of agent hierarchies.

From a communication standpoint, among the models studied and presented, the overwhelming majority use cooperation as the main tactic for generation, as opposed to competitiveness-based frameworks. One common factor of these approaches is represented by the low computational overhead needed to implement such a framework, over the design challenges of rigorous formats needed to achieve quality consensus with a limited number of agents (among which the most commonly used formats are inspired by voting or debating) or the scalability issues of evolutionary systems that need to employ a multitude of agents and a referee system, such as the model proposed by [20].

A developing line of inquiry involves interpreting multi-agent LLM systems from the point of view of criticality theory**.** In the context of multi-agent LLM architectures, this concept has been used to characterize the phase transitions between under-coordinated (fragmented) and over-constrained (rigidly hierarchical) interaction regimes. Hierarchical and modular systems such as MegaAgent and TalkHier tend to operate below the critical point, favouring stability and control but limiting emergent creativity. Conversely, horizontally-structured or loosely-moderated environments, as explored in AgentVerse, move closer to the critical regime, where agent interactions can generate novel behaviours at the cost of reduced predictability.

Overall, current multi-agent LLM research, while proving rapid progress, creates significant open areas of interest. While hierarchical and modular structures—such as those implemented in MegaAgent, AgentVerse, and TalkHier—enable controlled scaling and improved efficiency, they also risk constraining creativity and reducing emergent coordination. Meanwhile, systems employing transactive or shared memory, such as the work by Shang et al. [10], show promise for enhancing collective recall and continuity, yet still face limitations in adaptability and transparency.

4. Synthesis and conclusions

The present paper has presented an overview of the most relevant framework architectures for multi-agent LLM systems, classifying them from various points of view, such as architecture, tooling, and communication.

While many frameworks exhibit features not fully inscribed in the proposed classification, a discussion section has aimed to address the most significant talking points of their architectures.

The study identified the most widely used models to be hierarchical ones, featuring centralized memory and sequential planning for reasons of flexibility. Nevertheless, limitations of these models open the possibility for use-case-specific architectures. This paper has also identified as possible future study areas the use of transactive memory systems, of diversifying the communication formats, as well as of graph-based evaluation schemes for highly scalable, decentralized frameworks.

Future work is also expected to focus on interpretable coordination strategies, dynamic scalability, and the development of new metrics to support design principles for next-generation multi-agent LLM ecosystems.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, *Attention is all you need,* Advances in Neural Information Processing Systems, **30**, Curran Associates, Inc., 2017.

[2] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, D. Zhou, *Chain-of-thought prompting elicits reasoning in large language models*, Proceedings of the 36th International Conference on Neural Information Processing Systems, pp. 24824–24837, Red Hook, New York, 2022. https://dl.acm.org/doi/10.5555/3600270.3602070

[3] S. Han, Q. Zhang, Y. Yao, W. Jin, Z. Xu, C. He. *LLM Multi-Agent Systems: Challenges and Open Problems*, preprint, arXiv:2402.03578, 2024. https://doi.org/10.48550/arXiv.2402.03578

[4] B. Yan, X. Zhang, L. Zhang, L. Zhang, Z. Zhou, D. Miao, C. Li, *Beyond Self-Talk: A Communication-Centric Survey of LLM-Based Multi-Agent Systems*, preprint, arXiv:2502.14321, 2025. https://doi.org/10.48550/arXiv.2502.14321

[5] S. Mu, S. Lin, *A Comprehensive Survey of Mixture-of-Experts: Algorithms, Theory, and Applications*, preprint, arXiv:2503.07137, 2025. https://doi.org/10.48550/arXiv.2503.0713

[6] L. Ma, R. Zhang, Y. Han, S. Yu, Z. Wang, Z. Ning, J. Zhang, P. Xu, P. Li, W. Ju, C. Chen, D. Wang, K. Liu, P. Wang, P. Wang, Y. Fu, C. Liu, Y. Zhou, C.-T. Lu. *A Comprehensive Survey on Vector Database: Storage and Retrieval Technique, Challenge*, preprint, arXiv:2310.11703, 2021.

[7] J. Pan, J. Wang, G. Li, *Survey of Vector Database Management Systems*, VLDB J., **33**, pp. 1591-1615, 2023. https://doi.org/10.48550/arXiv.2310.14021

[8] D. M. Wegner, *Transactive memory: A contemporary analysis of the group mind*, Theories of group behavior, pp. 185–205, Springer-Verlag, New York, 1986.

[9] M. Catasta, A. Tonon, D. E. Difallah, G. Demartini, K. Aberer, P. Cudré-Mauroux, *TransactiveDB: Tapping into Collective Human Memories*, Proc. VLDB Endow., **7** (14), pp. 1977-1980, 2014. https://www.vldb.org/pvldb/vol7/p1977-catasta.pdf

[10] H. Y. Shang, X. Liu, Z. Liang, J. Zhang, H. Hu, S. Guo, *United Minds or Isolated Agents? Exploring Coordination of LLMs under Cognitive Load Theory*, preprint, arXiv:2506.06843v1, 2025. https://doi.org/10.48550/arXiv.2506.06843

[11] R.M. Aratchige and W. M. K. S. Ilmini, *LLMs Working in Harmony: A Survey on the Technological Aspects of Building Effective LLM-Based Multi Agent Systems*, preprint, arXiv:2504.01963, 2025. https://doi.org/10.48550/arXiv.2504.01963

[12] H. Sun, Y. Zhuang, L. Kong, B. Dai, C. Zhang, *AdaPlanner: Adaptive Planning from Feedback with Language Models*, preprint, arXiv:2305.16653, 2023. https://doi.org/10.48550/arXiv.2305.16653

[13] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, *ReAct: Synergizing Reasoning and Acting in Language Models*, preprint, arXiv:2210.03629, 2022. https://doi.org/10.48550/arXiv.2210.03629

[14] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, K. Narasimhan, *Tree of Thoughts: Deliberate Problem Solving with Large Language Models*, preprint, arXiv:2305.10601, 2023. https://doi.org/10.48550/arXiv.2305.10601

[15] L. Becchetti, A. Clementi, E. Natale, *Consensus Dynamics: An Overview*, ACM SIGACT News, **51** (1), pp.57, 2020. https://doi.org/10.1145/3388392.3388402

[16] R. Zhang, S. Eger, *Llm-based multi-agent poetry generation in non-cooperative environments*, preprint, arXiv:2409.03659, 2024. https://doi.org/10.48550/arXiv.2409.03659

[17] Q. Wang, T. Wang, Z. Tang, Q. Li, N. Chen, J. Liang, B. He, *MegaAgent: A Large-Scale Autonomous LLM-Based Multi-Agent System Without Predefined SOPs*, preprint, arXiv:2408.09955, 2024. https://doi.org/10.48550/arXiv.2408.09955

[18] W. Chen, Y. Su, J. Zuo, C. Yang, *AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors in Agents*, preprint, arXiv:2308.10848, 2023. https://doi.org/10.48550/arXiv.2308.10848

[19] Z. Wang, S. Moriyama, W.-Y. Wang, B. Gangopadhyay, S. Takamatsu, *Talk Structurally, Act Hierarchically: A Collaborative Framework for LLM Multi-Agent Systems*, preprint, arXiv:2502.11098v1, 2025. https://doi.org/10.48550/arXiv.2502.11098

[20] J. Cai, J. Li, M. Zhang, M. Li, C.-S. Wang, K. Tei, *Language Evolution for Evading Social Media Regulation via LLM-based Multi-agent Simulation*, IEEE Congress on Evolutionary Computation (CEC), 1–10, 2024..