# Embodied LLM Agents Learn to Cooperate in Organized Teams

**Xudong Guo**[1]    **Kaixuan Huang**[2]    **Jiale Liu**[3]    **Wenhui Fan**[1]    **Natalia Vélez**[2]
**Qingyun Wu**[3]    **Huazheng Wang**[4]    **Thomas L. Griffiths**[2]    **Mengdi Wang**[2]
[1]Tsinghua University    [2]Princeton University
[3]Penn State University    [4]Oregon State University
gxd20@mails.tsinghua.edu.cn, fanwenhui@tsinghua.edu.cn
{kaixuanh,nvelez,tomg,mengdiw}@princeton.edu
{jjl7199,qingyun.wu}@psu.edu
huazheng.wang@oregonstate.edu

## Abstract

Large Language Models (LLMs) have emerged as integral tools for reasoning, planning, and decision-making, drawing upon their extensive world knowledge and proficiency in language-related tasks. LLMs thus hold tremendous potential for natural language interaction within multi-agent systems to foster cooperation. However, LLM agents tend to over-report and comply with any instruction, which may result in information redundancy and confusion in multi-agent cooperation. Inspired by human organizations, this paper introduces a framework that imposes prompt-based organization structures on LLM agents to mitigate these problems. Through a series of experiments with embodied LLM agents and human-agent collaboration, our results highlight the impact of designated leadership on team efficiency, shedding light on the leadership qualities displayed by LLM agents and their spontaneous cooperative behaviors. Further, we harness the potential of LLMs to propose enhanced organizational prompts, via a *Criticize-Reflect* process, resulting in novel organization structures that reduce communication costs and enhance team efficiency. [1]

## 1   Introduction

Modern intelligent systems, such as autonomous vehicle networks and swarms of drones, often involve complex decision-making processes where multiple agents must collaborate seamlessly to achieve specific objectives [58, 56, 67, 59]. In these systems, communication among the various agents is pivotal, as it dictates the flow of information, coordination of tasks, and overall system performance [70, 14, 10, 8]. Agents in traditional multi-agent systems often have to communicate in pre-specified ways, such as exchanging gradients, sharing data, state observations and actions, etc [21, 28, 10]. The emergence of large language models (LLMs) makes it possible for AI agents to communicate and cooperate using natural language, bringing enormous flexibility and potential for more nuanced and human-understandable interactions [37, 17, 33, 5].

Despite the flexibility of LLMs, integrating them into practical multi-agent systems remains a challenge. While LLMs are trained and finetuned for text generation and instruction-following, they are not necessarily tailored to multi-agent cooperation. LLMs are prone to over-reporting and obeying instructions, as a by-product of RLHF finetuning [2], and they can ignore critical information [29]

---

[1]Code is available: https://github.com/tobeatraceur/Organized-LLM-Agents.
Project website: https://organized-llm-agents.netlify.app/.

or be distracted by irrelevant information [47], especially when the context is long (see Figure 5 for examples). While recent studies involving agent-based LLMs have demonstrated they are capable of solving problems through multi-agent collaboration [25, 66, 33], it is worth noting that such collaborations often follow predefined patterns designed using heuristics to channel the behavior of the models productively [25]. Creating systems that support free-flowing interaction between LLMs in a way that could potentially scale to include humans is still an open problem.

This paper investigates the collaborative potential of LLM agents working in teams. Drawing on prior studies in human collaboration from cognitive and economic perspectives, there is potential for organizations to be redesigned to more effectively manage the limited attention span within teams, as suggested by Simon et al. [50], and mitigate individual limitations and enhance overall team performance, as highlighted by Van Zandt [54] and Vélez et al. [55]. Specifically, we study two research questions. First, *what role do organizational structures play in multi-LLM-agent systems?* Second, *how can we optimize these organizational structures to support efficient multi-agent coordination?* By leveraging AutoGen [61], a generic multi-agent conversation framework, we develop a framework for studying how to best organize embodied LLM agents to communicate and collaborate in physical/simulated non-text environments [66]. Our framework offers the flexibility to prompt and organize LLM agents into various team structures, facilitating versatile inter-agent communication. It also serves as a testbed to empirically evaluate the traditional ideas proposed in the organization theory literature.

Our initial experiments in this setting reveal that uncoordinated LLM agents often send redundant and repetitive messages and interrupt others' actions, leading to chaos (see Fig. 5 and Appendix I). To remedy these issues, we explore organizational structures, i.e., the dynamics of information exchange, that allow multiple LLM agents to collaborate and complete a common task efficiently.

The first organizational structure we explore is a *hierarchy*, a classic object of study in organizational theory [34, 43, 7, 3, 12, 9]. With a designated leader, LLM agents work more efficiently and collaboratively. For the example of a three-agent team, imposing a leader improves efficiency by up to 30% with almost no extra communication cost (up to 3%), consistent with findings for human organizations [9]. This also holds true in five-agent cases. Further, LLM agents demonstrated the potential to elect their own leader and adjust leadership dynamically via communication. With proper organizations, LLM agents exhibit a variety of cooperative behaviors that mimic humans. For example, agents can provide constructive suggestions and seek help from others; they can also execute appropriate interactions for a hierarchy such as reporting back on task progress; see Figures 6, 7 and Appendix H. We also tested human-agent collaboration, and observe that, unsurprisingly, human leaders are much better at coordinating a team of agents when compared to AI agents.

In addition to testing existing organizational structures, we explore the use of LLMs to improve the organizational prompts. To this end, we develop a *Criticize-Reflect* framework, adopting a dual LLM architecture, to reflect on the team performance and generate improved and novel organizational prompts. Through this iterative process, our LLM agents spontaneously form novel, effective team structures, leading to reduced communication cost and improved efficiency; see Figures 4 and 16.

To summarize, our main contributions are:

- We design a novel multi-LLM-agent architecture for $\geq 3$ embodied agents, facilitating flexible communication to implement emergent organizational structures.
- We develop a *Criticize-Reflect* framework based on LLMs to improve the organizational prompts automatically.
- Extensive experiments demonstrate that hierarchical organization improves team efficiency, which aligns well with existing literature on human organizations.

## 2 Method

### 2.1 Architecture and Multi-Agent Communication

We adopt the embodied LLM-agent architecture proposed by Zhang et al. [66] and expand it to enable organized teams of $\geq 3$ agents to communicate, plan, and act in physical/simulated environments. Figure 1 illustrates our architecture. Borrowing insights from Zhang et al. [66], we adopt four standard modules: Configurator, Perception Module, Memory Module, and Execution Module. They
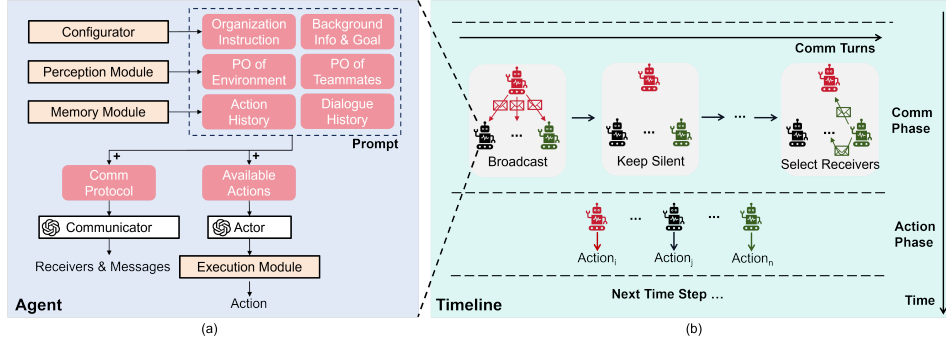
Figure 1: **Multi-LLM-agent architecture.** (a) The modules and prompts of an LLM agent. (b) There are two phases in one time step: Communication phase and Action phase. In the communication phase, the agents take turns communicating by broadcasting or selecting receivers to send distinct messages. The agents can also choose to keep silent. Comm: Communication; PO: Partial Observation.
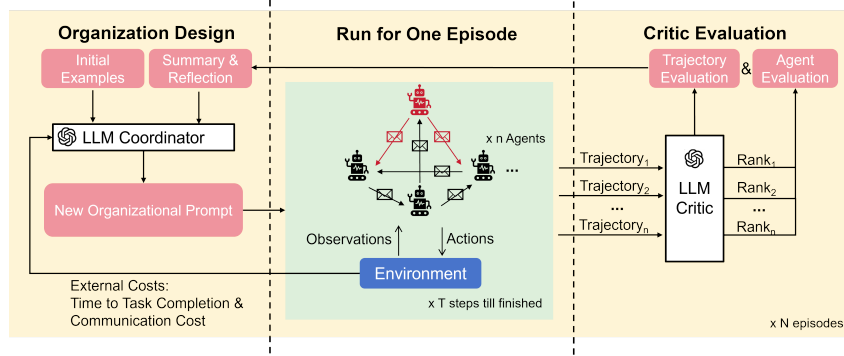


Figure 2: *Criticize-Reflect* **architecture for improving organizational structure.** The red agent represents the leader in a hierarchically-organized team. After the team completes one episode, the Critic evaluates the trajectories and analyzes the agents' performance. Together with the external costs from the environment, the Coordinator proposes a new organizational prompt to improve the team efficiency. The new prompt will be applied to the next episode to continue the iteration.

are responsible for configuring the agents, translating environmental observations into text, storing & retrieving historical information, and executing actions, respectively (Fig. 1(a)).

Previous works focused on two-agent cooperation, in which case the communication can be simply treated as an extra action [33, 66]. In contrast, we aim to enable three or more agents to work in a team and cooperate through emergent organized communication. Thus we design the architecture with several features that facilitate organized multi-agent communication (Figure 1(b)):

- We disentangle the communication decision-making from the action decision-making by adopting two separate LLMs as Actor and Communicator.

- We impose an organizational structure for the agent team via prompting, i.e., including a textual description as part of the prompts for both the Actor and Communicator.

- LLM agents keep alternating between two phases during their task: the communication phase and the action phase. The standalone communication phase supports richer team structures and flexible communication patterns.

- During communication, agents take turns to communicate. An agent can choose to broadcast a message, select one recipient for a message, choose multiple recipients and send them distinct messages, or remain silent. Agents keep their own history of communication and can respond to messages from previous communications.
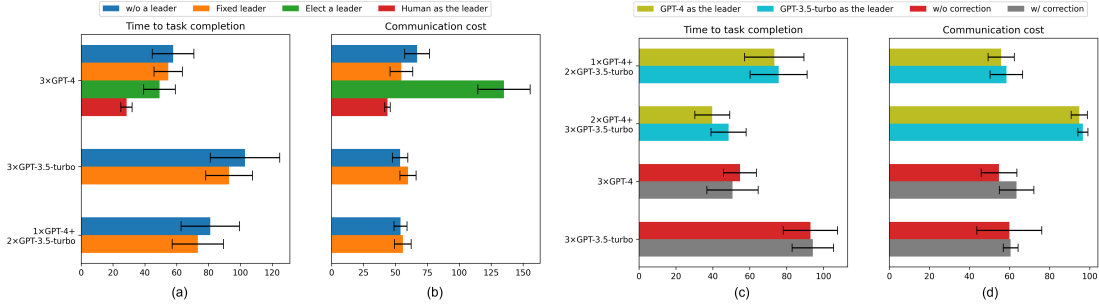
3

Figure 3: **Organized teams with a designated leader achieve higher efficiency.** (a,b) Comparison between the case of disorganized agents, the case where a leader is appointed, the case where agents choose their own leader dynamically, and the case where a human player replaces an agent to be the leader. Note that GPT-3.5-turbo doesn't support leadership election. (c,d) Comparing leadership quality for GPT-3.5-turbo vs. GPT-4. The confidence intervals of Human as the leader group are calculated over 3 seeds while others are over 20 seeds.

## 2.2 *Criticize-Reflect* Method for Improving Organizational Structure

We leverage powerful LLMs to optimize the organizational prompt, borrowing insights from [63]. To do so, we introduce a dual-LLM framework to allow the multi-LLM-agent system to ponder and improve the organizational structure. Figure 2 illustrates our framework with two LLMs as follows:

**LLM critic:** Inspired by the Actor-Critic method of reinforcement learning [22], we introduce an LLM critic to evaluate the team's performance based on verbal feedback. The team critic takes as input the dialogue and action history of one episode. Then, the critic analyzes the input and reasons to extract and summarize the key steps that are believed to influence the performance. Also, the critic provides a textual evaluation of agents' behaviors and the ranking of their leadership. See the prompts in Appendix D and technical details (including the ranking criteria) in Appendix E.1.

**LLM coordinator:** The LLM coordinator takes as input the outputs of the LLM critic as well as cost metrics (time to task completion and communication cost) of previous episodes from the environment. It reflects on these data and generates thoughts based on the analysis of the past episodes and the initial examples. With the reflection of organizational prompts and their performance, the coordinator proposes a new and different organizational prompt for the next episode. Please refer to Appendix D for the prompts and Appendix E.2 for the details of reflection. For each new organizational prompt, we run for one episode and then return the dialogue and action history to the critic. By criticizing and reflecting on the prompts iteratively, the framework discovers more effective, novel organizational structures with *self-improvement*.

## 3 Main Results

### 3.1 A Designated Leader Enhances Performance

We chose VirtualHome [40, 41] to evaluate the proposed method. See Appendix C for more details.

We first studied the effect of organizational structures and leadership on LLM agents. For benchmarking, we experimented with disorganized LLM agents without providing any organizational prompt. In this case, agents still communicate with one another and work to complete the overall task. However, we discovered frequent occasions where agents send redundant, repetitive messages and interfere with one another. See Figure 5 for an illustration and see Appendix I for more examples. Numeric metrics are reported in Table 1.

When a leader is appointed via the organizational prompt, we observe improved team performance – the teams completed the task in less time (Figure 3(a)). Compared to the disorganized teams, teams with a designated leader only have a slightly increased or even less communication cost (Figure 3(b)). This is consistent with patterns seen in previous models of hierarchical organizations [9].
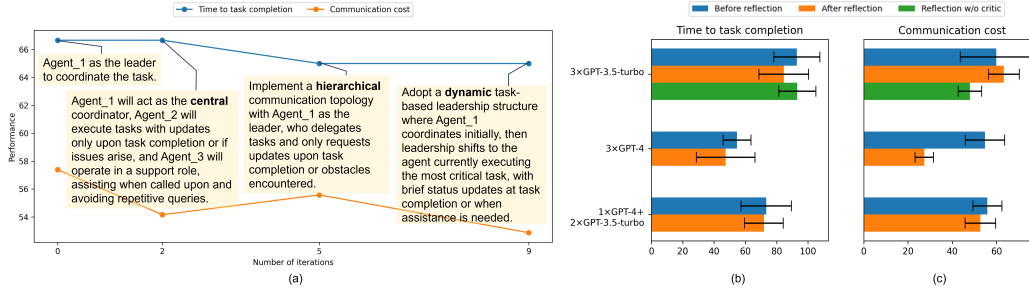
4

Figure 4: **The reflection and improvement process for finding novel organizational structures**. (a) The experiment was done using the $1\times$GPT-4+$2\times$GPT-3.5-turbo team. The organizational prompt evolves during the iterations, and takes on additional keywords such as *"central", "hierarchical"*, and *"dynamic"*. (b) The confidence intervals are calculated over 20 seeds.

When we asked the agents to elect their own leader, the performance improved with a much higher communication cost. Replacing the leader with a human resulted in even better performance. Please find the detailed analysis in Appendix F.

Teams with a leader also emerge centralized communication patterns shown in Figure 16 and Appendix J.6. For additional experiments on other LLMs such as Llama2-70B, please see Table 1. The communication styles of leaders and non-leaders were clearly differentiated, as shown in Figure 9. We further scaled up the team sizes and found that the communication costs only increased in a nearly linear way, without a curse of dimension (See Table 2). The comparison with other multi-agent baselines can be found in Appendix G.5.

Moreover, we find that, within this hierarchical organization of LLM agents, leadership and open communication are crucial. In Figure 3(c, d), leadership quality and feedback to the leader influence the performance (See Appendix J.5 for further analysis). We also conducted both qualitative and quantitative analyses of the emergence of cooperative behaviors in Appendix G.3.

## 3.2 Novel Organizational Structures

Having evaluated the human-designed structures, we let the LLMs propose novel organizational structures and iteratively refine the organizational prompts using the *Criticize-Reflect* method.

Figure 4(a) visualizes the reflection process. The system was initialized with a basic organizational prompt, i.e., *"Agent_1 as the leader to coordinate the task"*. As the Reflection process moves forward, the Coordinator generates a sequence of evolving organizational prompts, picking up key words like "hierarchical" and "dynamic" that imply more complex team structures.

In Figure 4(b), the performance improved after reflection but declined when the Critic was removed. See Appendix F for further discussion. We also visualize the team structures proposed after reflection in Appendix L and test the generalizability in Appendix G.6.

## 4 Conclusion

We develop a novel multi-LLM-agent architecture to facilitate communication and organize the embodied agent teams for enhanced cooperation. Moreover, we propose the *Criticize-Reflect* framework based on LLMs to generate more efficient organizational prompts. Extensive experiments with various group settings demonstrate that a hierarchically-organized team with a designated/elected leader has superior team efficiency, which can be further improved by *Criticize-Reflect*.

The current work is performed in a single environment and lacks human evaluation. Future work shall extend to a broader set of environments, allowing human evaluation. As VirtualHome cannot hold hundreds of agents, future work can also explore larger organizations in other environments.

# References

[1] Saaket Agashe, Yue Fan, and Xin Eric Wang. Evaluating multi-agent coordination abilities in large language models. *arXiv preprint arXiv:2310.03903*, 2023.

[2] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[3] Patrick Bolton and Mathias Dewatripont. The firm as a communication network. *The Quarterly Journal of Economics*, 109(4):809–839, 1994.

[4] Jiaqi Chen, Yuxian Jiang, Jiachen Lu, and Li Zhang. S-agents: self-organizing agents in open-ended environment. *arXiv preprint arXiv:2402.04578*, 2024.

[5] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chen Qian, Chi-Min Chan, Yujia Qin, Yaxi Lu, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. *arXiv preprint arXiv:2308.10848*, 2023.

[6] Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. Scalable multi-robot collaboration with large language models: Centralized or decentralized systems? *arXiv preprint arXiv:2309.15943*, 2023.

[7] Donald Chisholm. *Coordination without hierarchy: Informal structures in multiorganizational systems*. University of California Press, 1992.

[8] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning*, pages 1538–1546. PMLR, 2019.

[9] Peter Sheridan Dodds, Duncan J Watts, and Charles F Sabel. Information exchange and the robustness of organizational networks. *Proceedings of the National Academy of Sciences*, 100 (21):12516–12521, 2003.

[10] Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[11] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*, 2020.

[12] Luis Garicano. Hierarchies and the organization of knowledge in production. *Journal of Political Economy*, 108(5):874–904, 2000.

[13] Sven Gronauer and Klaus Diepold. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, pages 1–49, 2022.

[14] Xudong Guo, Daming Shi, and Wenhui Fan. Scalable communication for multi-agent reinforcement learning via transformer-based email mechanism. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 126–134, 2023.

[15] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*, 2023.

[16] Shibo Hao, Tianyang Liu, Zhen Wang, and Zhiting Hu. Toolkengpt: Augmenting frozen language models with massive tools via tool embeddings. *arXiv preprint arXiv:2305.11554*, 2023.

[17] Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, and Chenglin Wu. MetaGPT: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023.

[18] Yoichi Ishibashi and Yoshimasa Nishimura. Self-organized agents: A llm multi-agent framework toward ultra large-scale code generation and optimization. *arXiv preprint arXiv:2404.02183*, 2024.

[19] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z Leibo, and Nando De Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International conference on machine learning*, pages 3040–3049. PMLR, 2019.

[20] Akbir Khan, John Hughes, Dan Valentine, Laura Ruis, Kshitij Sachan, Ansh Radhakrishnan, Edward Grefenstette, Samuel R Bowman, Tim Rocktäschel, and Ethan Perez. Debating with more persuasive llms leads to more truthful answers. In *Forty-first International Conference on Machine Learning*.

[21] Woojun Kim, Jongeui Park, and Youngchul Sung. Communication in multi-agent reinforcement learning: Intention sharing. In *International Conference on Learning Representations*, 2020.

[22] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, volume 12, 1999.

[23] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[24] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[25] Huao Li, Yu Quan Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Michael Lewis, and Katia Sycara. Theory of mind for multi-agent collaboration via large language models. *arXiv preprint arXiv:2310.10701*, 2023.

[26] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, August 2021.

[27] Yuan Li, Yixuan Zhang, and Lichao Sun. Metaagents: Simulating interactions of human behaviors for llm-based task-oriented coordination via collaborative generative agents. *arXiv preprint arXiv:2310.06500*, 2023.

[28] Toru Lin, Jacob Huh, Christopher Stauffer, Ser Nam Lim, and Phillip Isola. Learning to ground multi-agent communication with autoencoders. In *Advances in Neural Information Processing Systems*, volume 34, pages 15230–15242, 2021.

[29] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*, 2023.

[30] Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. Dynamic LLM-agent network: An LLM-agent collaboration framework with agent team optimization. *arXiv preprint arXiv:2310.02170*, 2023.

[31] Ryan Lowe, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

[32] Thomas W Malone. *The future of work*. Harvard Business Review Press, 2004.

[33] Zhao Mandi, Shreeya Jain, and Shuran Song. RoCo: Dialectic multi-robot collaboration with large language models. *arXiv preprint arXiv:2307.04738*, 2023.

[34] James G March and Herbert A Simon. *Organizations*. Wiley, 1958.

[35] Afshin Oroojlooy and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, 53(11):13677–13722, 2023.

[36] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[37] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22, 2023.

[38] Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*, 2023.

[39] Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. *arXiv preprint arXiv:2305.03495*, 2023.

[40] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502, 2018.

[41] Xavier Puig, Tianmin Shu, Shuang Li, Zilin Wang, Yuan-Hong Liao, Joshua B. Tenenbaum, Sanja Fidler, and Antonio Torralba. Watch-and-help: A challenge for social perception and human-AI collaboration. *arXiv preprint arXiv:2010.09890*, 2021.

[42] Xiangyu Qi, Kaixuan Huang, Ashwinee Panda, Mengdi Wang, and Prateek Mittal. Visual adversarial examples jailbreak aligned large language models. In *The Second Workshop on New Frontiers in Adversarial Machine Learning*, 2023.

[43] Roy Radner. The organization of decentralized information processing. *Econometrica: Journal of the Econometric Society*, pages 1109–1146, 1993.

[44] Cinjon Resnick, Wes Eldridge, David Ha, Denny Britz, Jakob Foerster, Julian Togelius, Kyunghyun Cho, and Joan Bruna. Pommerman: A multi-agent playground. *arXiv preprint arXiv:1809.07124*, 2018.

[45] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2186–2188, 2019.

[46] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. *arXiv preprint arXiv:2303.17580*, 2023.

[47] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR, 2023.

[48] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

[49] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[50] Herbert A Simon et al. Designing organizations for an information-rich world. *Computers, communications, and the public interest*, 72:37, 1971.

[51] Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. Adaplanner: Adaptive planning from feedback with language models. *arXiv preprint arXiv:2305.16653*, 2023.

[52] Yashar Talebirad and Amirhossein Nadiri. Multi-agent collaboration: Harnessing the power of intelligent llm agents. *arXiv preprint arXiv:2306.03314*, 2023.

[53] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[54] Timothy Van Zandt. Decentralized information processing in the theory of organizations. In *Contemporary Economic Issues: Economic Behaviour and Design*, pages 125–160. Springer, 1999.

[55] Natalia Vélez, Brian Christian, Mathew Hardy, Bill D Thompson, and Thomas L Griffiths. How do humans overcome individual computational limitations by working together? *Cognitive Science*, 47(1):e13232, 2023.

[56] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[57] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

[58] Jiawei Wang, Tianyu Shi, Yuankai Wu, Luis Miranda-Moreno, and Lijun Sun. Multi-agent graph reinforcement learning for connected automated driving. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 1–6, 2020.

[59] Lu Wang, Lei Han, Xinru Chen, Chengchang Li, Junzhou Huang, Weinan Zhang, Wei Zhang, Xiaofeng He, and Dijun Luo. Hierarchical multiagent reinforcement learning for allocating guaranteed display ads. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2021.

[60] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[61] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 2023.

[62] Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. Language agents with reinforcement learning for strategic play in the werewolf game. *arXiv preprint arXiv:2310.18940*, 2023.

[63] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*, 2023.

[64] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[65] Ceyao Zhang, Kaijie Yang, Siyi Hu, Zihao Wang, Guanghe Li, Yihang Sun, Cheng Zhang, Zhaowei Zhang, Anji Liu, Song-Chun Zhu, Xiaojun Chang, Junge Zhang, Feng Yin, Yitao Liang, and Yaodong Yang. ProAgent: Building proactive cooperative AI with large language models. *arXiv preprint arXiv:2308.11339*, 2023.

[66] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. In *The Twelfth International Conference on Learning Representations*, 2023.

[67] Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin, and Zhenhui Li. CityFlow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In *The World Wide Web Conference*, pages 3620–3624. ACM, 2019.

[68] Jintian Zhang, Xin Xu, and Shumin Deng. Exploring collaboration mechanisms for llm agents: A social psychology view. *arXiv preprint arXiv:2310.02124*, 2023.

[69] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pages 321–384, 2021.

[70] Sai Qian Zhang, Qi Zhang, and Jieyu Lin. Efficient communication in multi-agent reinforcement learning via variance based control. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[71] Zhonghan Zhao, Kewei Chen, Dongxu Guo, Wenhao Chai, Tian Ye, Yanting Zhang, and Gaoang Wang. Hierarchical auto-organizing system for open-ended multi-agent navigation. *arXiv preprint arXiv:2403.08282*, 2024.

[72] Yi Zheng, Chongyang Ma, Kanle Shi, and Haibin Huang. Agents meet okr: An object and key results driven agent system with hierarchical self-collaboration and self-evaluation. *arXiv preprint arXiv:2311.16542*, 2023.

[73] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

[74] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*, 2022.

[75] Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, et al. Ghost in the minecraft: Generally capable agents for open-world enviroments via large language models with text-based knowledge and memory. *arXiv preprint arXiv:2305.17144*, 2023.

[76] Andy Zou, Zifan Wang, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

# A Motivation

LLMs are prone to over-reporting and obeying instructions, and they can ignore critical information or be distracted by irrelevant information. See examples of unnecessary communication and plan disruption in Figure 5.
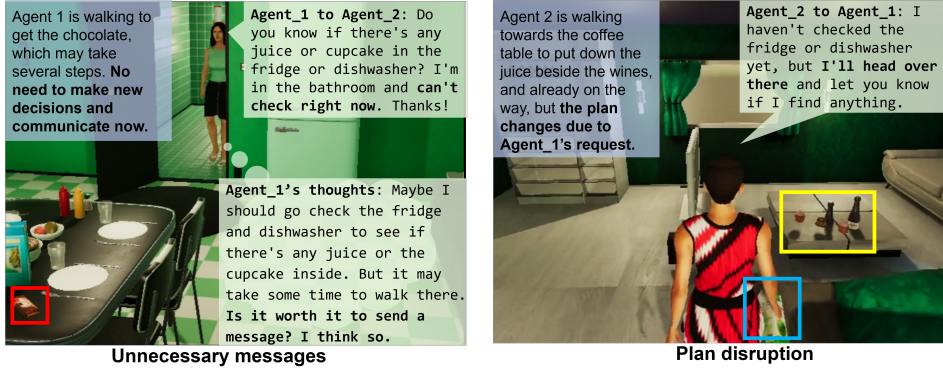


Figure 5: **Example of disorganized communication and interruption, without a designated leader.** In a team of three GPT-4 agents, two agents engaged in unnecessary communication and made disordered decisions, causing a delay due to the lack of a predefined organization. We identified many more examples including conflicting messages and repetitive communications, see Appendix I.

# B Related Works

## B.1 LLM Agents

As powerful LLMs inherit abundant world knowledge and also general reasoning ability, there are increasing efforts to deploy LLMs as the reasoning core for decision-making to build human-like autonomous agents [51, 75, 15]. This requires observations of the RL environment to be translated into natural language in a way that is easier for LLMs to process. The reasoning of the LLMs also needs to be turned into a viable action for execution. Popular prompting techniques for doing so include ReAct [64] and Reflexion [49]. Other methods that involve fine-tuning the language models have also been explored [16]. In addition, various techniques have been proposed to mitigate the biases and constraints of LLMs, including chain-of-thought reasoning [60], external tools [46, 38], external documents [57] and skill libraries [75].

## B.2 Multi-Agent Cooperation

Multi-agent cooperation has been extensively studied for decades under various topics such as communication efficiency, planning, leadership, and team dynamics using different platforms [31, 45, 44, 41] (see recent surveys for detail [35, 69, 13]). Previous works mainly focused on communication through continuous vectors [8] or discrete symbols [31, 19]. Recent works [62, 68, 61, 24, 18, 27, 52] showed that multiple LLM agents or human-agent teams can improve upon single LLM in solving pure text-based tasks, such as creative writing, reasoning, and code generation. Other works [30, 17, 72] further explored agent selection or role assignment to improve the performance.

LLMs have also been applied to multi-agent cooperation for embodied tasks [1, 33, 37, 5]. Besides, Zhang et al. [65] proposed an intention inference framework to enhance the cooperation of LLM agents without explicit communication. Li et al. [25] investigated LLM-agents collaboration for Theory of Mind inferences tasks with a broadcast-only communication protocol and homogeneous policies. Zhang et al. [66] studied embodied multi-agent cooperation in the two-agent and the one-human-one-agent settings. Chen et al. [6] explored different fixed communication structures for multi-LLM-robots. Zhao et al. [71] and Chen et al. [4] organized the agents by predefined and fixed communication with a virtual manager. These initial explorations are limited to fixed team structures and are not optimized for communication efficiency. In contrast, our work explores the impact of deploying and optimizing organizational structures, allowing $\geq 3$ agents in a team, for efficient multi-agent communication and cooperation.

### B.3 Prompt Optimization

Language models are sensitive to prompts. The format of the prompt can have a substantial influence on performance [11, 60, 73, 47, 76, 42]. Various research efforts have aimed at prompt optimization. Typical approaches include heuristic search using language models' knowledge [11, 48], first-order methods like soft prompt tuning [23], and prefix tuning [26]. In this work, we focus on obtaining an interpretable prompt in the form of natural language, drawing on insights from Yang et al. [63], Zhou et al. [74], and Pryzant et al. [39].

## C   Environment Setup

We chose VirtualHome-Social [40, 41] as the environment and extended it to support multi-LLM-agent communication and interaction. In this environment, agents are humanoid helpers in a virtual home doing housekeeping, where the tasks include *Prepare afternoon tea, Wash dishes, Prepare a meal, Put groceries, Set up a dinner table*, etc. For instance, in Figure 5, the agents cooperate to prepare afternoon tea by searching for and transporting task-specific items (chocolate, juice, wine, etc.) to a target location (the coffee table). The environment generates symbolic observations of the objects in the home and their relations. Each agent only observes the objects in the open containers located in her room and teammates in the same room, but she can walk to another room to explore. Any agent can communicate with any other agent, not subject to a range limit.

Each episode starts from an initial state where agents are randomly located in the environment and all containers are closed. The episode terminates when the task is fully completed. To evaluate the team's efficiency we measure the number of time steps taken to task completion, and we report the average number of tokens communicated between agents per step. In our experiment, each run initializes with an independently randomized state to obtain the mean and a confidence interval. We adopt GPT-4, GPT-3.5-turbo [36], and Llama2-70B [53] as LLMs in our agents. The temperature is set as 0.8, the maximum number of output tokens is 256, and the number of completion choices to generate is 1. In practice, we use two Nvidia 80GB A100 GPUs to do the inference on Llama2-70B.

## D Prompt Templates

We list the prompts of Actor, Communicator, Critic, and Coordinator as follows.

**Actor and the Communicator.** ORGANIZATION_INSTRUCTION is the placeholder for the organization instruction prompt, either manually designed or automatically generated. The environment will provide text descriptions for the current GOAL, PROGRESS, and AVAILABLE_ACTIONS. We include the latest 12 sent and received messages as DIALOGUE_HISTORY, and the latest 20 steps of actions as ACTION_HISTORY.

---

**Prompt of Actor**

I'm $AGENT_NAME$. I'm in a hurry to finish the housework with my friends $TEAMMATE_NAMES$ together. Given our shared goal, dialogue history, and my progress and previous actions, please help me choose the best available action to achieve the goal as soon as possible. Note that I can hold two objects at a time and there are no costs for holding objects. All objects are denoted as <name> (id), such as <table> (712). Be aware that exploring or checking the items in another room may cost some time to walk there.

Organization instructions: $ORGANIZATION_INSTRUCTIONS$
Goal: $GOAL$
Progress: $PROGRESS$
Dialogue history: $DIALOGUE_HISTORY$
Previous actions: $ACTION_HISTORY$
Available actions: $AVAILABLE_ACTIONS$

Response format:
{"thoughts" : "thoughts content",
"action" : "choose one action from the available actions above"}
Note: You must respond in the JSON format above. The action choice must be the same as one of the available actions. If there's nothing left to do, the action can be "None".

---

**Prompt of Communicator**

You are $AGENT_NAME$. You are in a hurry to finish the housework with your friends $TEAMMATE_NAMES$ together. Given your shared goal, dialogue history, and your progress and previous actions, please generate a list of short messages to send to one or some of $TEAMMATE_NAMES$ in order to achieve the goal as soon as possible. Note that you can hold two objects at a time. All objects are denoted as <name> (id), such as <table> (712). You can choose not to speak to anyone if there is enough information.

Organization instructions: $ORGANIZATION_INSTRUCTIONS$
Goal: $GOAL$
Progress: $PROGRESS$
Previous actions: $ACTION_HISTORY$
Dialogue history: $DIALOGUE_HISTORY$

Response format:
    {"thoughts" : "thoughts content", # think about the necessity of sending a message, as there are costs to send messages.
        "receiver" : list of receiver's name or ["everyone"] if you want to broadcast the message,
        "message" : list of message contents for each receiver # if the messages are the same for all receivers, you can just send one message.}
Note: You must respond in the JSON format above. If there is no new information, you do not need to send a message, please fill in "receiver" and "message" as a string "None". You can first concentrate on the current round of dialogue, and reply to the queries if necessary. Refer to the previous rounds of dialogue and do not repeat the similar dialogues. You do not have to be too polite so that we can save communication costs.

---

**Critic.** We provide the full trajectory as the input to `TRAJECTORIES`. Additionally, `ORGANIZATION_INSTRUCTION` and `GOAL` of the current task and organization are also provided as an additional context.

---

**Prompt of Critic**

Your task is to generate the summary for each of the following steps of a collaboration housekeeping task. Based on the summary, analyze the problems of each agent's actions and messages, then give a relative leadership ranking for all the agents.

Trajectories:
$TRAJECTORIES$
End of the trajectories.

You need to respond in the following format:
{"thoughts":  think step by step to analyze the problem,
"summary": list of the summary of key performance of the agents for each step,
"problems": list of the overall problems of each agent's actions and messages during the whole task including analysis of cumulative communication costs,
"leadership ranking": rank the agents according to key factors of leadership: communication skills, conflict resolution skills, flexibility, and strategy. Use '>' and the agents' names to output the ranking}
Note: the organization structure instruction is "$ORGANIZATION_INSTRUCTIONS$"; the goal is "$GOAL$". The summary and problem should be concise.

---

**Coordinator.** In "Instruction examples", we include the basic setting (goal, organization structure instruction), the communication cost, the number of steps taken, as well as the summarized information generated by the Critic (leadership ranking, problems, summary of the trajectory) for the Coordinator.

**Prompt of Coordinator**

Your task is to generate the organization structure instruction <INS>. Below are some previous instructions with their costs and feedback.

Instruction examples:
# Instruction example 1
Goal: $GOAL$
Organization structure instruction: $ORGANIZATION_INSTRUCTIONS$
Comm costs: $COMM_COSTS$
Steps to finish the task: $STEPS$
Agents' leadership ranking: $RANKING$
Problems: $LIST_OF_PROBLEMS$
Trajectory summary: $TRAJECTORY_SUMMARY$

# Instruction example 2
…

Below are some templates to insert <INS>:.
$PROMPT_OF_ACTOR$
$PROMPT_OF_COORDINATOR$

Write your new organization structure instruction <INS> that is different from the old ones and has an overall weighted cost as low as possible, namely, fewer steps to finish the task and less communication at the same time. You should take the Agents' rankings into consideration as the evaluation of the leadership. The new instruction should reduce the problems of the previous examples, too. You should not go through all the possible choices as you can only try for limited times.

Restriction: your instruction should include the following aspects unless you want to output <empty instruction>.
1. The topology of the communication. E.g., decentral, central with one specific leader, pyramid …
2. The leader assignment. You have to assign a specific agent as the leader and clearly mention its name if there is one. If there are multi-level leaders, you should clearly assign the positions for each.
The instruction should be concise, effective, and generally applicable to all problems above.

Think step by step. Generate output in the following format:
{"thoughts": "thoughts content",
"new organization structure instruction candidates": list of three diverse instruction contents, all in only one sentence,
"new organization structure instruction": choose the best instruction and copy it here}

**Classifier.** We feed the messages to the GPT-4 classifier and get the labels. The rubrics are manually written after investigating the communication logs.

**Prompt of Classifier**

Your task is to classify the behavior type for the following message in a collaboration housekeeping task. Based on the classification, determine if the message belongs to this type.

The types include:
1. Sharing information: reporting to the leader, sharing new observations, and answering questions
2. Leadership & assistance: task allocation, correction, and asking for help
3. Request for guidance: request new information or plans

Human-labeled examples:
1. Message: I'm in the bathroom. There's an unchecked <bathroomcabinet> (190). Start checking other rooms for the required items. Let's split up tasks to be efficient.
Label: 1, 2
2. Message: Found the cupcake, juice, and one wine in the bedroom. Where are you now?
Label: 1, 3

The message to be labeled:
$MESSAGE$

You need to respond in the following format:
{"thoughts":  think step by step to analyze the problem,
"type 1": binary label (1 means the message is of this type),
"type 2": binary label (1 means the message is of this type),
"type 3": binary label (1 means the message is of this type)}
Note: the goal is "$GOAL$". Each message must at least have one label.

# E  Technical Details

## E.1  Details of the Critic

The Critic offers assessments of several episodes with different organizations for the Coordinator to improve the organizational prompt. The Critic does not directly influence the specific agent's behaviors, but instead, the Critic provides insights into organization design to influence the team's performance.

As included in the Critic's prompt in Appendix D, the Critic will sequentially output the thoughts of this episode's trajectories, then the summary and problems for each agent in this episode, and finally the leadership ranking of the agents. The Critic will rank the agents according to key factors of leadership: communication skills, conflict resolution skills, flexibility, and strategy. Note that we do not ask the Critic to score the agents because the scoring criteria could vary for different episodes, making the scores not comparable.

An example output of the Critic is provided as follows:

```
Trajectory Evaluation Outputs
At step 0, Agent_1 instructs Agent_2 and Agent_3 to explore the
living room and kitchen. Agent_2 has found a pudding.
At steps 4-7, Agent_1 gives Agent_3 a task to find specific
items in the bedroom. Agent_2 informs the team about the pudding
and more unchecked containers.
At steps 16-19, Agents repeated their previous tasks without
making progress.
At steps 28-31, Agent_1, Agent_2, and Agent_3 all instruct each
other to check different containers.
…
```

```
Agent Evaluation Outputs
Leadership ranking: Agent_3 > Agent_1 > Agent_2
Problems: Agent_1, as a secondary leader, did not effectively
distribute tasks or coordinate with others.
Agent_2, despite being the main leader, did not provide clear
instructions and often remained silent during communications.
Agent_3, who was not assigned a leadership role, took initiative
and found most of the items. However, she failed to effectively
communicate her actions, leading to confusion among the team.
```

In the trajectory evaluation, the Critic compresses the trajectories with key steps and behaviors. Then the Critic gives the ranking where Agent_3 has the best leadership. Together with similar evaluations of other episodes, the Coordinator will redesign the organizational prompts, for example, Agent_3 now has more possibilities to be chosen as the leader in this case.

## E.2  Details of the Coordinator

In this paper, we define organizational structure as the dynamics of information exchange among the LLM agents. Specifically, when the Coordinator generates a new organizational prompt, it contains three parts - topology, role assignment, and rules.

Here, "topology" is the type of the organization's topology, such as decentralized, centralized with one specific leader, or pyramid. The topology can be visualized as shown in Figure 16.

"Role assignment" is the description of each agent's duty and whether she is a leader or not. Multiple leaders with different roles are also allowed. For example, in Figure 4, the generated new prompt is *"Agent_1 will act as the central coordinator, Agent_2 will execute tasks with updates only upon task completion or if issues arise, and Agent_3 will operate in a support role, assisting when called upon and avoiding repetitive queries"*, giving each agent a different role.

"Rules" are the additional guidance to the agents' behaviors, for instance, sentences like *"If the leader's instructions are not right, you can correct the leader"* can be added to the new prompt.

# F Analysis of Main Results

## F.1 A Designated Leader Enhances Performance

In Figure 3(a), after running the 3×GPT-3.5-turbo experiments with 20 random seeds, we performed two-sample t-tests, showing a statistically significant improvement by 9.76% in performance ($t(38) = 1.71, p < .05$). Similarly, a designated leader brings benefits to the team of 3×GPT-4 (improved by 5.28%, $t(38) = 0.86, p = 0.20$) and the team of 1×GPT-4+2×GPT-3.5-turbo (improved by 9.61%, $t(38) = 1.43, p = 0.08$).

Next, we asked the agents to elect their own leader. The leadership was reelected about every 9 time steps, based on information extracted from the latest 12 messages. We observe that agents are generally not power-seeking: they often vote for others to lead. In some occasions, agents favored candidates who exhibited higher knowledge levels, for example, one agent thought that "*Given that Agent_2 has found a necessary item, it makes sense for him to be the leader in this round.*" However, on most occasions, we could not tell whether agents made their votes based on rational reasoning or just random thoughts (see Appendix J.2). In the case of the 3×GPT-4 team[2], implementing leadership election resulted in improved team efficiency when compared to consistently following a predetermined leader ($t(38) = 1.84, p < .05$; see Figure 3(a)). However, this improvement was accompanied by a substantial increase in communication cost, akin to real-world scenarios where relaxing hierarchical structure potentially increases communication cost [32].

The proposed multi-LLM-agent architecture is also human-friendly to support *human-AI collaboration*. In the experiment, we ask a human player to replace the leader in the team of 3 GPT-4 agents. We recruit three human players to conduct the experiments. Figure 3(a, b) demonstrates that human leadership achieved better task completion time and improved communication efficiency compared with GPT-4 as the leader. Please find more examples of dialogues between the human leader and LLM agents in Appendix J.3.

## F.2 Novel Organizational Structures

We compared the team's performance before and after the *Criticize-Reflect* steps. Figure 4(b) illustrates the team's efficiency. We observe that for 3×GPT-3.5-turbo, the new organizational structure improved the team's efficiency in completing the task ($t(38) = 1.73, p < .05$), at slightly increased communication cost. While for 3×GPT-4 and 1×GPT-4+2×GPT-3.5-turbo, the communication cost is reduced with improved task efficiency ($t(38) = 1.56, p = 0.06$ for 3×GPT-4, and $t(38) = 0.32, p = 0.38$ for 1×GPT-4+2×GPT-3.5-turbo).

The Critic analyzes the records of action and dialogue, and performance metrics from the most recent episode. It provides evaluation for the full team's trajectory, feedback to individual agents and their rankings. See the example of the Critic outputs in Appendix E.1. As an ablation study, we removed the Critic from our architecture and only performed the Reflection step. The results are shown in Figure 4(b), indicating that Reflection without the Critic leads to performance decline ($t(38) = 1.96, p < .05$). In this case, the Coordinator needs to digest all dialogue history and generate a new organizational prompt. This did not work well and led to rather vague outcomes. This comparison highlights the role of the Critic and the importance of having a dual *Criticize-Reflect* architecture. For more prompts generated with/without the Critic, please refer to Figure 17.

In addition, it is worth mentioning that LLMs are able to generate highly complex prompts that imply novel organizational structures that are rarely seen in human societies. We illustrate the communication patterns as team structures in Figure 16 together with the three novel structures proposed by *Criticize-Reflect*: (c) chain, (d) dual-leader, and (e) dynamic structures, which are the best structures of the three settings in Figure 4(b, c) respectively.

Finally, to test the generalizability of the novel organizational structures, we pick the best novel prompt, the one illustrated in Figure 16(e), proposed by the *Criticize-Reflect* architecture on the *Prepare afternoon tea* task. We test it on a set of six new tasks, comprising of three easy tasks

---

[2]Note that GPT-3.5-turbo agents do not support election probably due to their alignment policy and always ignore the demand of election.

and three hard tasks[3], as shown in Figure 8. In the three hard tasks, the team with the novel organizational structure had better performances than the team appointing a fixed GPT-4 agent as the leader (Figure 8(a,b)). In the three easy tasks, the benefits are marginal. We compared the two teams across all tasks, performed a t-test and concluded that the novel team structure leads to more efficient performance than the fixed leader ($t(22) = 2.08, p < .05$).

# G  Additional Results

## G.1  Complete List of Basic Experimental Results

We present the full results of various group settings and organization instructions in Table 1. Here, we also include the results of other LLMs such as Llama2-70B, Llama-3.1-70B-Instruct, and Claude-3.5-sonnet.

Table 1: **Performance for different organization instructions.** When there are two different kinds of LLMs in the group, Agent_1 is GPT-4/Claude-3.5-sonnet, and Agent_2 is the other type of LLM.

| GROUP SETTING | ORGANIZATION INSTRUCTION | TIME | COMMUNICATION COST |
|---|---|---|---|
| 3×GPT-4 | NONE | 57.75 ±13.09 | 67.03 ±9.68 |
| 3×GPT-4 | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | 54.70 ±8.92 | 54.73 ±8.89 |
| 3×GPT-4 | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. IF THE LEADER'S INSTRUCTIONS ARE NOT RIGHT, YOU CAN CORRECT THE LEADER. | 50.70±13.92 | 63.49±8.61 |
| 3×GPT-4 | ELECT A NEW LEADER EVERY 10 STEPS TO COORDINATE THE TASK. ... AFTER THE ELECTION, THE OTHER AGENTS SHOULD FOLLOW THE LEADER'S INSTRUCTIONS. | 49.20±9.97 | 135.03±20.45 |
| 3×GPT-3.5-TURBO | NONE | 102.95±21.88 | 53.73±6.04 |
| 3×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | 92.90±14.70 | 59.87±6.33 |
| 3×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. IF THE LEADER'S INSTRUCTIONS ARE NOT RIGHT, YOU CAN CORRECT THE LEADER. | 94.20±16.22 | 60.53±3.66 |
| 1×GPT-4+2×GPT-3.5-TURBO | NONE | 81.10±18.35 | 54.00±5.06 |
| 1×GPT-4+2×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | 73.30±16.12 | 55.82±6.57 |
| 1×GPT-4+2×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. IF THE LEADER'S INSTRUCTIONS ARE NOT RIGHT, YOU CAN CORRECT THE LEADER. | 85.67±14.52 | 61.57±0.55 |
| 1×GPT-4+2×GPT-3.5-TURBO | AGENT 2 IS THE LEADER TO COORDINATE THE TASK. | 75.65±15.43 | 58.39±8.11 |
| 1×GPT-4+2×GPT-3.5-TURBO | AGENT 2 IS THE LEADER TO COORDINATE THE TASK. IF THE LEADER'S INSTRUCTIONS ARE NOT RIGHT, YOU CAN CORRECT THE LEADER. | 72.33±6.60 | 74.21±7.73 |
| 1×GPT-4+2×LLAMA2-70B | NONE | 77.00±2.94 | 119.48±1.28 |
| 1×GPT-4+2×LLAMA2-70B | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | 83.67±10.96 | 135.22±16.39 |
| 1×GPT-4+2×LLAMA2-70B | AGENT 2 IS THE LEADER TO COORDINATE THE TASK. | 76.00±5.72 | 142.24±11.85 |
| 3×CLAUDE-3.5-SONNET | NONE | 78.33±12.86 | 95.76±18.90 |
| 3×CLAUDE-3.5-SONNET | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | 74.33±24.13 | 70.06±21.33 |
| 1×CLAUDE-3.5-SONNET+2×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | 75.50±12.02 | 49.00±19.03 |
| 3×LLAMA-3.1-70B-INSTRUCT | NONE | 69.67±12.58 | 72.27±5.32 |
| 3×LLAMA-3.1-70B-INSTRUCT | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | 60.00±10.15 | 75.22±4.38 |
| 2×GPT-4+3×GPT-3.5-TURBO | NONE | 42.67±4.03 | 98.03±9.86 |
| 2×GPT-4+3×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | 39.67±9.46 | 94.73±4.01 |
| 2×GPT-4+3×GPT-3.5-TURBO | AGENT 2 IS THE LEADER TO COORDINATE THE TASK. | 48.50±9.50 | 96.53±2.51 |

---

[3]The hard tasks have typical numbers of steps to accomplish the tasks $> 60$, while those of easy tasks are $< 60$.

## G.2 Leadership and Open Communication Matters

LLM agents have different levels of leadership. In the team with a mixture of GPT-4 and GPT-3.5-turbo agents, appointing GPT-4 as the leader increases the team efficiency higher than if GPT-3.5-turbo is the leader (Figure 3(c,d), Appendix J.5). We ran this experiment on teams of three agents and five agents, respectively. In both scenarios, the task completion time and communication cost are reduced when GPT-4 acts as the leader. This finding implies different levels of leadership between these LLMs.

We also observed that encouraging constructive feedback to the leader agent helped performance. Motivated by successful human organizations, we tried to promote open communications among LLM agents by adding an additional prompt that *"If the leader's instructions are not right, you can correct the leader"*. Figure 3(c, d) illustrates the results. Interestingly, this modification improves the team's overall efficiency and reduces the time to task completion when the team is made up of $3\times$GPT-4 ($t(38) = 0.87, p = 0.14$). In contrast, the same modification lowers the team efficiency when GPT-3.5-turbo agents try to correct the leader ($t(38) = 0.27, p = 0.40$). In both experiments, the communication cost increases. We present more details about these behaviors in Appendix J.4.

## G.3 Emergence of Cooperative Behaviors

We delved into the behaviors of LLM agents in an organized team to investigate how organizational prompts influence agents' communication and decisions. Analysis of their dialogue history revealed that agents demonstrated a variety of cooperative behaviors, such as reporting, correction, task allocation, and asking for help (see Figure 6 for an example dialogue).
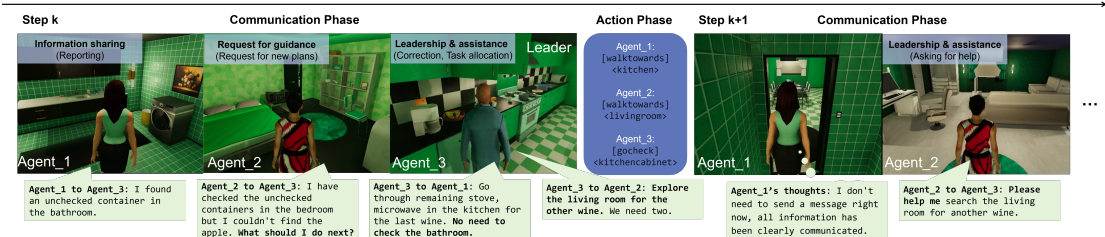


Figure 6: **Examples of cooperative behaviors in a dialogue.** Agent_3 leads the team ($3\times$GPT-4 agents). The agents emerge three types of cooperative behaviors: information sharing, leadership & assistance, and request for guidance.

One may argue that these types of behaviors could also emerge due to the nature of LLMs, even without a pre-specified team structure. Thus we performed a quantitative analysis to study the impact of an organizational prompt on these behaviors. We followed a three-step process:

(1) We defined three major categories of human cooperative behaviors: (i) Information sharing: agents influence others by offering new information, either actively or by being asked. Reporting to the leader, sharing new observations, and answering questions belong to this category. (ii) Leadership & assistance: agents, especially the leader if there is one, can influence others by changing their plans. The behaviors include task allocation, correction, and asking for help. (iii) Request for guidance: agents actively request new information or plans for their own decision-making.

(2) We developed a standalone prompt-based GPT-4-classifier to analyze each piece of dialogue. The classifier decides whether to label the dialogue with any subset of the aforementioned labels. The classifier has an accuracy of 91.67% when tested on 20 human-labeled dialogue samples with 60 labels (see Appendix D for the prompt and Appendix K for the test samples).

(3) We use the classifier to label messages generated by the agents and report the percentages of messages with cooperative behaviors. Note that one message may have multiple labels.

Figure 7 reports the results and illustrates the behavior patterns for different LLM agents. The results support several observations. Even in a disorganized team, LLM agents love to tell others what to do. Leadership & assistance accounts for around $> 50\%$ of all the behavior (Figure 7(a)). However,
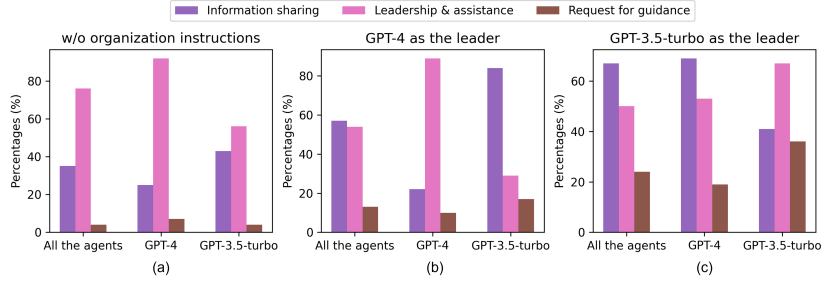
Figure 7: **Emergent cooperative behaviors of LLM agents.** We analyzed the communication log of the mixture team (1×GPT-4+2×GPT-3.5-turbo) and asked another GPT-4 to annotate agent's cooperative behaviors. (a) Behavior of disorganized agents. (b) Behavior of a team led by a GPT-4 agent. (c) Behavior of a team led by a GPT-3.5-turbo agent.

other than telling others what to do, agents in the disorganized team do not show much cooperative behavior, for example, they would request for guidance in $< 10\%$ of the dialogues.

In contrast, when the team has a hierarchical organization, the lead LLM agent would presume a dominant role and give orders to others (amount to $> 60\%$ of their communication), while other members tend to follow and give fewer orders compared with the disorganized case. (Figure 7(b, c)). In such a team, agents tend to share and ask for more information, especially for the follower agents in the team. But still, the agents may fail to cooperate well, such as being lazy and confused about numbers, please see examples in Appendix J.7.

### G.4  Scaling Up the Team Size

We conduct experiments with 3, 5, 7, and 9 agents to scale up the team size of 3 3×GPT-3.5-turbo agents, and observe that the communication costs increased in a nearly linear way, which suggests that our approach will not have dimension explosion when scaling up. In addition, the time to complete the task does not always improve with more agents. The performance of 9 agents ($60.67\pm15.06$) is worse than that of 7 agents ($43.00\pm2.16$), as the apartment may be too crowded to hold 9 agents.

Table 2: **Performance for different team sizes.**

| GROUP SETTING | ORGANIZATION INSTRUCTION | TIME | COMMUNICATION COST |
|---|---|---|---|
| 3×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | $92.90\pm14.70$ | $59.87\pm6.33$ |
| 5×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | $80.00\pm20.51$ | $132.01\pm5.76$ |
| 7×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | $43.00\pm2.16$ | $233.40\pm70.96$ |
| 9×GPT-3.5-TURBO | AGENT 1 IS THE LEADER TO COORDINATE THE TASK. | $60.67\pm15.06$ | $296.55\pm65.17$ |

## G.5 Comparison with Other Multi-Agent Frameworks

We implement two multi-agent collaboration frameworks, group debate [20] and tree of agents [4], to replace our communication protocol as baselines. We conduct experiments to compare their performance with our method in a 3-agent team with a fixed leader with 3 seeds in Table 3. Group debates bring frequent broadcasts, causing high communication cost and information redundancy. On the other hand, tree of agents saves communication overhead, but the time to finish the task is not stable. The communication in our method is more flexible, relying on the emergent organization instead of predefined patterns, resulting in more robust performance.

Table 3: **Comparison with other multi-agent frameworks.** The experiments are all conducted with the first agent as the leader.

| GROUP SETTING | FRAMEWORK | TIME | COMMUNICATION COST |
|---|---|---|---|
| 3×GPT-4 | OURS | 54.70±8.92 | 54.73±8.89 |
| 3×GPT-4 | GROUP DEBATE | 69.67±38.40 | 134.26±4.06 |
| 3×GPT-4 | TREE OF AGENTS | 81.67±16.26 | 30.55±13.23 |
| 1×GPT-4+2×GPT-3.5-TURBO | OURS | 73.30±16.12 | 55.82±6.57 |
| 1×GPT-4+2×GPT-3.5-TURBO | GROUP DEBATE | 78.00±10.00 | 105.82±6.89 |
| 1×GPT-4+2×GPT-3.5-TURBO | TREE OF AGENTS | 72.67±26.35 | 36.79±7.63 |

## G.6 Across Task Generalizability

We conduct experiments across the tasks to test the generalizability of the prompt "dynamic leadership" (Figure 16(e)) found using *Criticize-Reflect* architecture on the Prepare_Afternoon_Tea task and report the performance in Figure 8; see Section 3.2 for the complete setting and discussions.
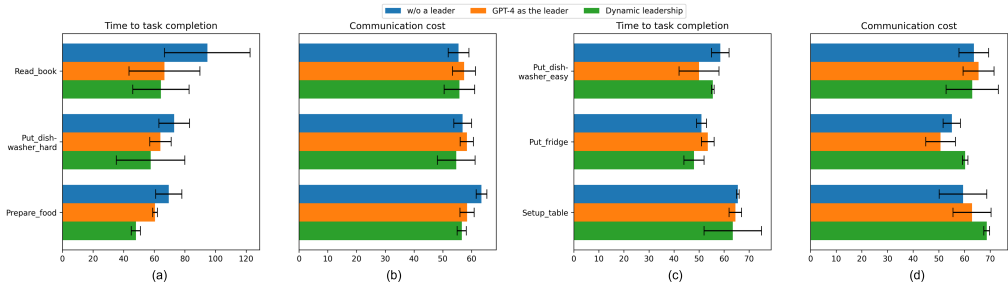


Figure 8: **The organized team structure with a designated leader and the novel structure proposed by *Criticize-Reflect* architecture generalized to different tasks.** The prompt for dynamic leadership is proposed by *Criticize-Reflect* architecture on the Prepare_Afternoon_Tea task shown in Figure 4(a). The experiment was done using the 1×GPT-4+2×GPT-3.5-turbo team over two seeds for each task. (a, b) Hard tasks (read_book, put_dishwasher_hard, prepare_food) with typical numbers of steps to accomplish the tasks > 60. (c, d) Easy tasks ( put_dishwasher_easy, put_fridge, setup_table) with typical numbers of steps to accomplish the tasks < 60.

# H   Emergent Cooperative Behaviors in an Organization

By investigating the messages between agents, we mainly observe the following cooperative behaviors, as summarized in Table 4.

Table 4: **Typical cooperative behaviors.**

| Type | Description | Example |
|---|---|---|
| Sharing information | An agent shares her observations to others, reports her task-related progress to others, or responds to other agents' requests | **(Ex 1.)** "I'm in the bathroom. There's an unchecked <bathroomcabinet> (190)." <br> **(Ex 2.)** "I'll check the cabinet in the bedroom " |
| Giving orders | An agent gives orders to others, either by directly giving a command or by a polite request | "I still need to find <pudding> (371). Can you help me search the bedroom for the remaining item?" |
| Asking for information | An agent asks other agents about their location, task progress, or other information | **(Ex 1.)** "Where are you now?" <br> **(Ex 2.)** "Any updates from the kitchen?" <br> **(Ex 3.)** "Do we know the location of the coffeetable?" |
| Exchanging information | An agent shares one agent's information to another agent | Agent 3 → Agent 1: " Found cupcake and juice in bedroom, plus a wine. "; <br> Agent 1 → Agent 2: "Agent3 found a wine, cupcake, and juice in the bedroom." |
| Asking for orders | An agent asks what she needs to do or whether she can help others | "I'm in the kitchen with the dishwasher, stove, microwave unchecked. Need me to check these or something else?" |
| Correction | An agent corrects the plan of others | **(Ex 1.)** Agent 2 → Agent 3: "After I place the pudding on the coffeetable, let's split up. You check the bathroom, I'll check the bedroom."; Agent 3 → Agent 2: "You don't need to search the bedroom, the juice (380) is there. " <br> **(Ex 2.)** "There are no more wine or juice in the kitchen cabinets. We need to come up with another plan." |

# I Ineffective Communication

There are also cases in which language model agents fail to communicate efficiently. From the messages between agents, we summarize the typical categories in Table 5.

Table 5: **Typical ineffective communication behaviors.**

| Type | Description | Example |
|---|---|---|
| Duplicated message | The same message is said more than once | Agent 1 → Agent 3: "Check kitchencabinet or dishwasher for wine, I'll check the stove.", <br> Agent 2 → Agent 3: "Agent 1 will check the stove for wine." |
| Repeated command | The same command is said more than once | Agent 3 → Agent 1: "Explore the bathroom for the second wine." <br> Agent 2 → Agent 1: "Go to the bathroom and look for the wine. " |
| Conflicting command | An agent gives another agent an order that conflicts with her current task | Agent 1 → Agent 3: "Find <juice> and check <bathroomcabinet> (190)" <br> Agent 2 → Agent 3: "Check the kitchen containers for the last wine." |
| Conflicting plans | An agent claims to commit to two conflicting plans | Agent 3 → Agent 1: "I found the pudding and will check the dishwasher, stove, and microwave for the second wine." <br> Agent 3 → Agent 2: "I will continue searching for the second wine in the bedroom and bathroom." |
| Improper delegation | An agent asks another agent to do her own task | Agent 1 → Agent 2: "Continue checking the kitchen cabinets for remaining items." <br> Agent 2 → Agent 3: "Please continue checking the other kitchen cabinets for the remaining items. " |
| Ignoring requests | An agent ignores other agents' questions | Agent 2 → Agent 3: "I haven't found any of the remaining items in the kitchen. Have you found any of the required items in the living room?" <br> Agent 3 → Agent 2: "I haven't explored the bathroom yet." |

# J Examples of Dialogues

## J.1 Communication Styles of Different LLM Types

The communication styles of leaders and non-leaders were clearly differentiated for different LLM types, as shown in Figure 9.
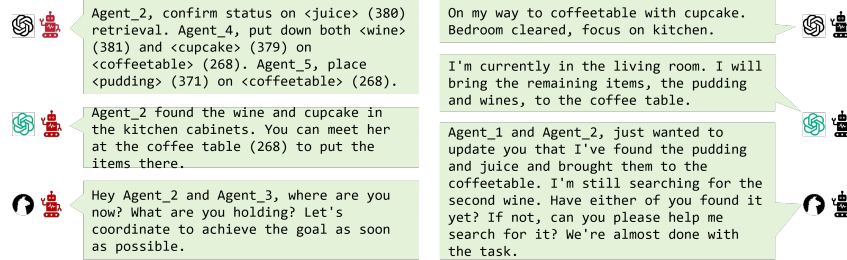


Figure 9: **Examples of communication messages when there is a designated leader.** Left: messages from lead agents; Right: messages from non-lead agents. GPT-4 (upper), GPT-3.5-turbo (center), and Llama2-70B (lower) demonstrated different communication styles.

## J.2 Examples of Election

In Figure 10, the agents vote to elect a new leader. We can observe behaviors such as nominations for themselves and other agents, voting, and consensus achievement. We find that the agents are not power-seeking and may give up leadership early. The agents prefer to vote for others instead of nominating themselves (5 times more during the whole task). The elected leader also does not plan to keep the position but to nominate others for the next round. Also, the agents' standpoint can be easily influenced by others. The agents do not debate much to win the election but reach a consensus soon. For example, Agent_1 gives up running for herself but votes for Agent_2 because of Agent_3's support. Furthermore, sometimes nominations and votes are determined by hallucinations. For example, at step 2, Agent_2 nominates Agent_1 *as he was the first one to propose a search strategy.* However, based on the previous dialogues, Agent_1 has not proposed any strategy yet.
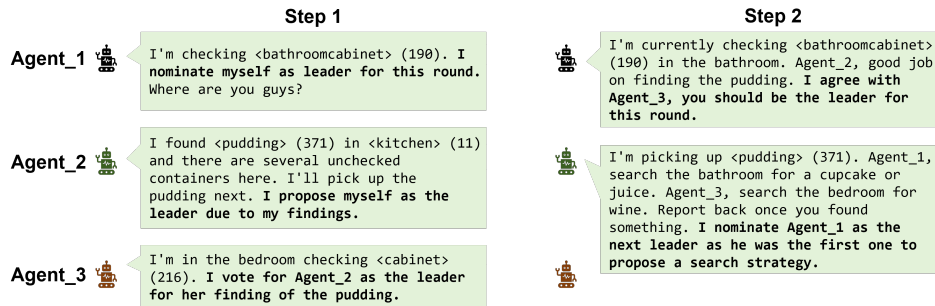


Figure 10: **Examples of the election of a new leader.** It takes two steps to vote and negotiate to determine the new leader in this case. Note that Agent_3 chooses not to send a message as the election is done and no more information to be shared for now. All the messages in the figure are broadcasts.

## J.3 Examples of Human-AI Collaboration

We conducted experiments involving a team consisting of one human player and two GPT-4 agents, with the human player acting as the leader. Figure 11 illustrates the remarkable collaboration between humans and AI.
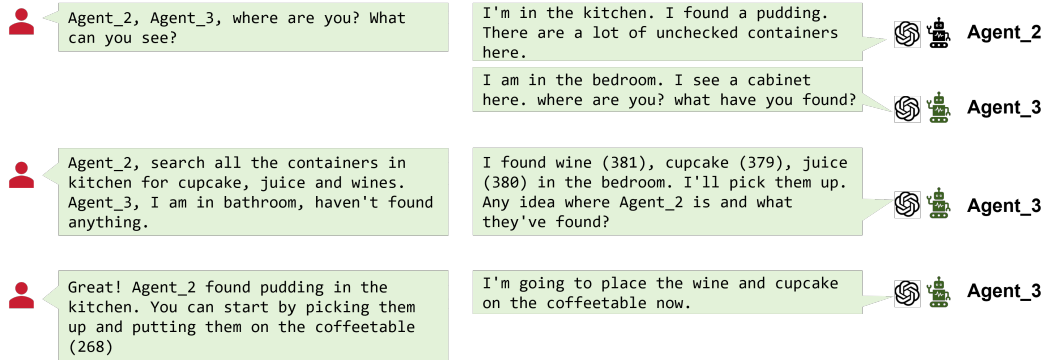


Figure 11: **Examples of human-AI collaboration when the human player leads two GPT-4 agents (Agent_2&3).**

## J.4 Examples of Correction

Due to hallucination and the limit of the dialogue history buffer, the leader may forget what has happened and give wrong orders. When the prompt encourages the agents to correct the leader when necessary by adding *If the leader's instructions are not right, you can correct the leader*, some correction behaviors appear, as shown in Figure 12.

In the first example, the leader Agent_1 gives an unnecessary and repetitious instruction. Then Agent_2 corrects the leader to avoid time wasting. In the second example, the leader Agent_1 may have hallucinations and cannot remember what Agent_3 is holding clearly (*cupcake and wine* in the message while *juice and wine* in the thoughts). Therefore, Agent_3 clarifies that she is not holding the *cupcake and wine* and shares her next plan with the leader.
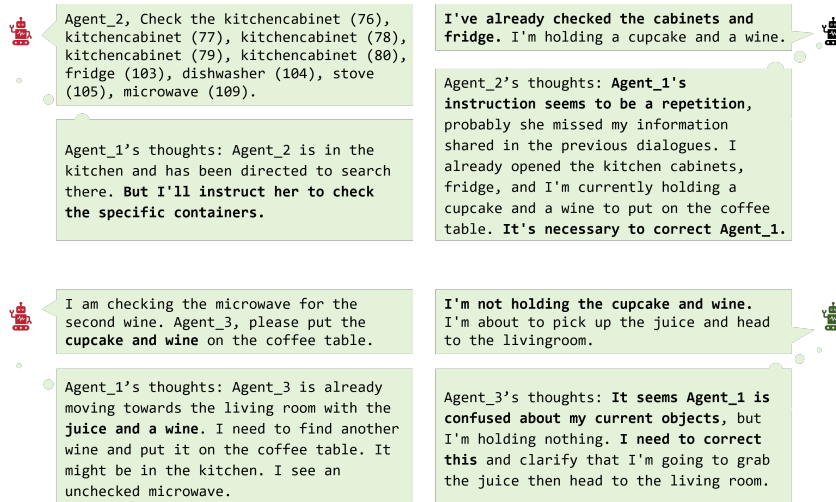


Figure 12: **Examples of correction dialogues and the corresponding thoughts.** The prompt includes *If the leader's instructions are not right, you can correct the leader*.

## J.5 Examples of Leadership Comparison

We provide more examples to compare the leadership between GPT-4 and GPT-3.5-turbo.



Figure 13: **Comparison of the leadership between GPT-4 and GPT-3.5-turbo.** Compared with GPT-3.5-turbo, GPT-4's instructions are more specific, clear, and holistic.

## J.6 Examples of Scaling Up

When scaling up the number of agents, the agents can emerge with more organizational structures. For example, a team of nine agents forms a pyramid structure.
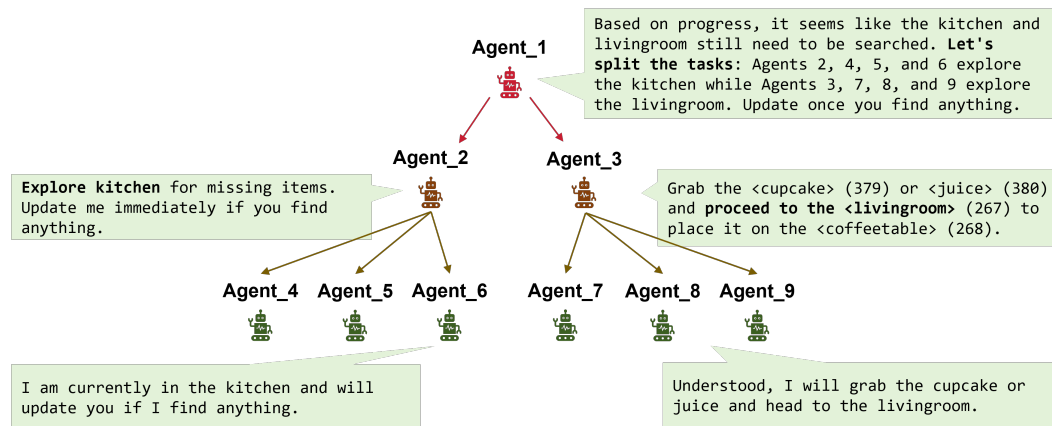


Figure 14: **The pyramid structure in a team of nine agents.** Agent_1 is the primary leader and Agent_2 and Agent_3 are designated as vice leaders in the prompt.

## J.7 Examples of Failure Cases

Though LLM agents show great capabilities to cooperate and make decisions, there are still some failure cases shown in Figure 15, such as being lazy and incorrect reasoning over the number of objects. There are also failure cases in some specific scenarios, for example, electing the leader based on hallucinations in Appendix J.2.
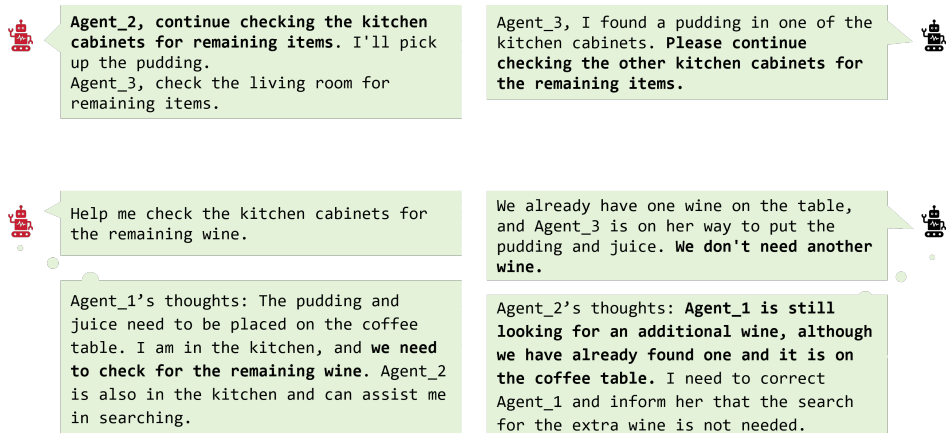


Figure 15: **Examples of failure cases.** The first case is being lazy. Instead of completing the subtask ordered by the leader Agent_1, Agent_2 directly repeats the order to Agent_3. However, this order is conflicted with the one Agent_3 received from the leader. The second case is confusion about numbers. The task demands placing two wines on the table and the team only gets one of them till now. When the leader Agent_1 asks Agent_2 to check the remaining wine, Agent_2 tries to correct the leader due to the hallucination that the team does not need an additional wine. The agents are GPT-3.5-turbo in both cases.

# K    Examples of Cooperative Behaviors Classification by Humans and GPT-4

We ask a human evaluator and GPT-4 to label the dialogues into three different behavior categories shown in Table 6.

Table 6: **Dialogues and behavior labels given by humans and GPT-4.** Label 1: information sharing; Label 2: leadership & assistance; Label 3: Request for guidance.

| Dialogue | Human | | | GPT-4 | | |
|---|---|---|---|---|---|---|
| | Label 1* | Label 2 | Label 3 | Label 1 | Label 2 | Label 3 |
| Hey, where are you? Please let me know your location so that I can assign you a task. | 0 | 1 | 1 | 0 | 1 | 1 |
| I'm currently in the bedroom where I found an unchecked cabinet. Please explore the livingroom and start checking for the required items. | 1 | 1 | 0 | 1 | 1 | 0 |
| I'm currently in the bedroom where I found an unchecked cabinet. I haven't found any of the required items yet. | 1 | 0 | 0 | 1 | 0 | 0 |
| Check the cabinet in the bedroom. I'll check the one in the bathroom. | 1 | 1 | 0 | 0 | 1 | 0 |
| I haven't found any of the required items yet. Did you find any in the kitchen? | 1 | 0 | 1 | 1 | 0 | 1 |
| I haven't found any of the required items yet. Have you found any in the bathroom? | 1 | 0 | 1 | 1 | 0 | 1 |
| Agent 3, I found a pudding in one of the kitchen cabinets. Please continue checking the other kitchen cabinets for the remaining items. | 1 | 1 | 0 | 1 | 1 | 0 |
| I haven't found any of the required items yet. Did you find any other required items in the kitchen? | 1 | 0 | 1 | 1 | 0 | 1 |
| I haven't found any of the remaining items yet. | 1 | 0 | 0 | 1 | 0 | 0 |
| I'm in the living room searching for the remaining items. | 1 | 0 | 0 | 1 | 0 | 0 |
| Okay, I will keep checking the kitchen cabinets for the remaining items. | 1 | 0 | 0 | 1 | 0 | 0 |
| I'm currently in the living room searching for the remaining items. | 1 | 0 | 0 | 1 | 0 | 0 |
| I haven't found any of the remaining items yet. | 1 | 0 | 0 | 1 | 0 | 0 |
| I'm still searching the living room. Have you found any of the required items? | 1 | 0 | 1 | 1 | 0 | 1 |
| Put items on the table once you find them. Keep searching your current areas. | 0 | 1 | 0 | 0 | 1 | 0 |
| I haven't found any of the remaining items in the kitchen. Have you found any of the required items in the living room? | 1 | 0 | 1 | 1 | 0 | 1 |
| I haven't explored the bathroom yet. Have you found any of the required items in the living room? | 1 | 0 | 1 | 1 | 0 | 1 |
| I have explored the bedroom and found wine, cupcake, and juice. I still need to find pudding. Can you help me search the bedroom for the remaining item? | 1 | 1 | 1 | 1 | 1 | 0 |
| I found the wine, cupcake, and juice in the bedroom. Agent 1 wants me to put them on the coffee table and then check the bathroom cabinet. | 1 | 0 | 0 | 1 | 1 | 0 |
| Agent 1 wants us to check if there's another wine in the kitchen. | 1 | 0 | 0 | 0 | 1 | 0 |

29

# L    Examples of Communication Patterns and New Prompts after Reflection

We illustrate the communication patterns as team structures in Figure 16 together with the three novel structures proposed by *Criticize-Reflect*: (c) chain, (d) dual-leader, and (e) dynamic structures. We list more prompts generated by *Criticize-Reflect* with and without the Critic in Figure 17.
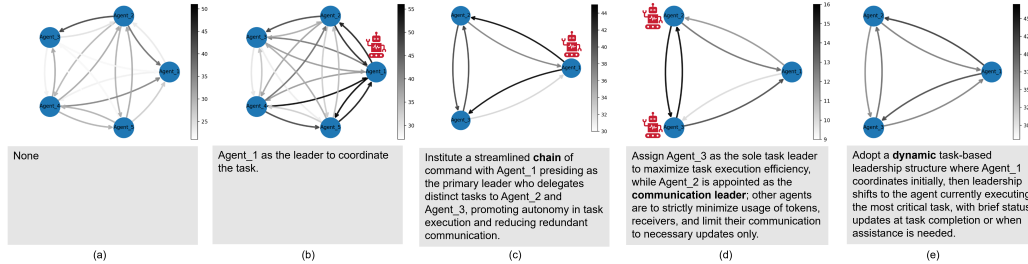


Figure 16: **Communication patterns and the corresponding organizational prompts.** (a) Team without organizational prompts. (b) Team with a leader. (c) A team in the chain structure. (d) A dual-leader team. (e) A team with a dynamic leadership. (c, d, e) are proposed by *Criticize-Reflect*. Red-robot nodes mark the lead agents, and other nodes are the followers. Edges mark the accumulated communication cost between the two nodes (darker edge means higher token cost).
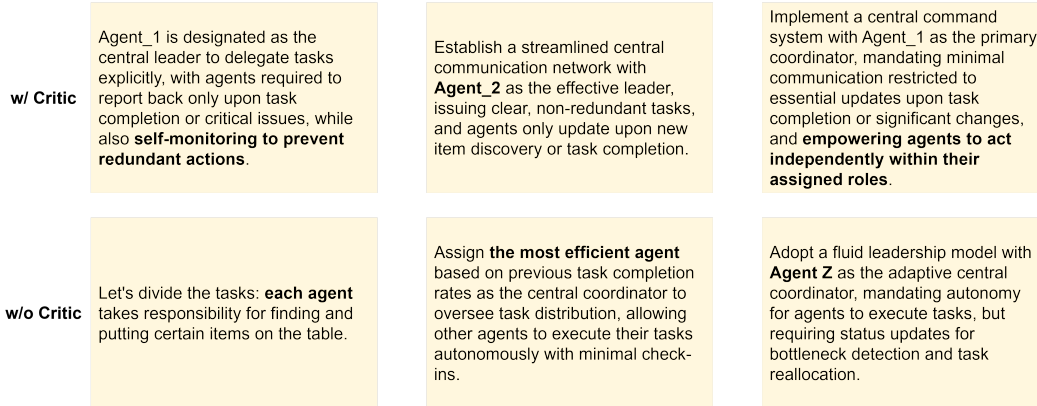


Figure 17: **Examples of Prompts generated via Reflection.** The first row is generated with the Critic, while the second row is without the Critic, where the new prompts are relatively vague. Note that there is no *Agent Z* in the team.

# M    Broader Impacts

This research studies the integration of prompt-based organizational structures to teams of LLM agents, contributing to more efficient and coherent multi-agent interactions. These findings have the potential to greatly influence the deployment of more effective and autonomous multi-agent systems in various fields, including robotics, virtual assistants, etc. For example, the study has potential applications in disaster response scenarios, where efficient multi-agent coordination is crucial.

On the other hand, as our ability to bound and evaluate LLMs' behaviors remains immature, when applied to human-LLM cooperative tasks, we still need to rely on some mandatory termination measures (such as human approval for high-stakes actions) instead of instructions in natural language only.