# Reliable Decision-Making for Multi-Agent LLM Systems

**Xian Yeow Lee, Shunichi Akatsuka, Lasitha Vidyaratne,
Aman Kumar, Ahmed Farahat, Chetan Gupta**

Industrial A.I. Laboratory
Hitachi America, Ltd.

## Abstract

Large Language Model (LLM)-based multi-agent systems are gaining prominence for their capabilities in reasoning, planning, and generating human-like responses. While much of the focus in multi-agent systems has been on aspects like partial observability, heterogeneous task allocation, and other collaborative paradigms, ensuring reliability and robustness remains a critical challenge, particularly in high-stakes applications. This work frames multi-agent decision-making as a redundancy and fault-tolerance problem, deploying multiple agents to solve the same tasks. Through experiments on benchmark problems—including resource allocation, question answering, topic classification and summarization—we find that simpler strategies, such as Majority Voting and Decentralized architectures, often outperform more complex feedback-based systems in terms of consistency. Feedback mechanisms, while sophisticated, risk error propagation and destabilization, highlighting the value of independent decision-making and aggregation. These findings emphasize the importance of balancing simplicity and reliability when designing robust multi-LLM systems for industrial and mission-critical applications.

## Introduction

The deployment of Large Language Model (LLM)-based systems has rapidly evolved, extending beyond individual task-solving capabilities to forming collaborative multi-agent systems. By leveraging LLMs' natural language understanding, reasoning, and planning capabilities, multi-agent systems have found applications in various domains, including autonomous coordination in logistics (Lang et al. 2008), collaborative robotics in manufacturing (Guo and Zhang 2009), and intelligent decision-support in healthcare (Shakshuki and Reid 2015). These systems promise to address challenges that single-agent solutions struggle to tackle, such as handling distributed tasks, dynamic environments, and complex inter-agent dependencies.

While much of the focus in developing these systems has been on enhancing performance metrics such as accuracy or task efficiency, the practical deployment of multi-agent LLM systems, particularly in industrial settings, demands a greater emphasis on reliability. In high-stakes applications like supply chain optimization, resource allocation, and emergency response, reliability is often more critical than peak performance (Xu and Saleh 2021). For instance, industries may accept a model achieving 80% accuracy, provided it maintains this performance consistently under expected operating conditions. Variability and inconsistency in system outputs, even at higher performance peaks, can lead to disruptions and costly errors in industrial workflows (Ge et al. 2017).

The transition from single-agent to multi-agent paradigms, such as ensemble systems of agentic AI systems, introduces additional layers of complexity and uncertainty (Hu, Lu, and Clune 2024). Multi-agent systems aggregate decisions from multiple models, which raises questions about how aggregation strategies influence system reliability. As such, redundancy and fault tolerance become pivotal, especially when multiple agents are deployed to solve the same problem to enhance robustness. Aggregation strategies must be carefully designed to ensure that combining the outputs of multiple agents leads to consistent and reliable decisions, rather than amplifying inconsistencies (Gupta and Vaidya 2020; Awad et al. 2017).

In this paper, we explore these challenges on multiple problems involving LLM agents, specifically 1) resource allocation, 2) question answering, 3) topic classification and 4) summarization, as a preliminary study. The resource allocation problem is a critical and widely applicable domain in industrial operations. Meanwhile, the question-answering, topic classification and text summarization problems are classical tasks in natural language processing. We evaluate various multi-agent aggregation strategies and aim to identify approaches that enhance reliability without sacrificing system performance in terms of accuracy. Through our empirical experiments, we anecdotally demonstrate that increased architectural complexity does not necessarily improve reliability, emphasizing the importance of balancing sophistication with robustness. We hope our findings will provide insights into the design of reliable multi-agent LLM-based systems and contributes to the growing body of research on deploying multi-agent LLM-based systems in industrial environments.

## Related work

As AI systems continue to rise, so do the complexity and challenges of deploying them to manage a wide range of

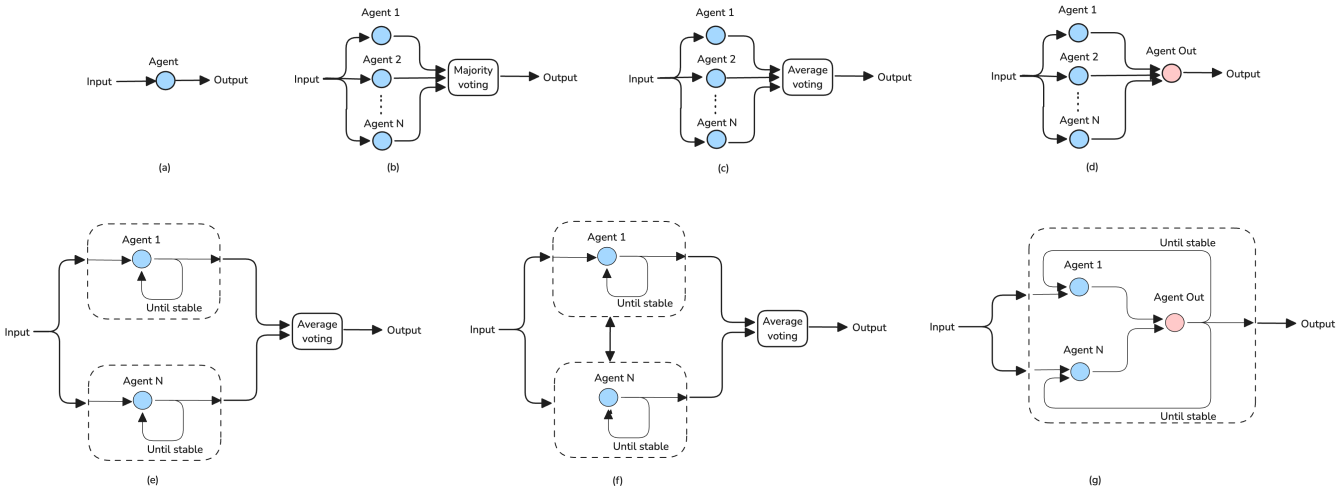



Figure 1: Aggregation strategy of (a) Single Agent, (b) Majority Voting, (c) Average Voting, (d) Spoke & Wheel, (e) Decentralized, (f) Decentralized (Feedback), (g) Spoke & Wheel (Feedback).

tasks. To tackle this, researchers have focused on multi-agents systems where each AI agent interacts and collaborates with each other, utilizing their specialized abilities towards an objective or tasks that are beyond the capacity of a single-agent. Earlier works had focused on creating solutions using a single agent with the help of strategies that split the task into smaller and less complex tasks (Chen et al. 2022; Yao et al. 2024; Wang et al. 2024; Jin and Lu 2023; Long 2023; Besta et al. 2024). On the other hand, tasks that require deep thought and innovation has also been demonstrated to be effectively performed by the collaboration between multiple agents (Dohan et al. 2022; Jinxin et al. 2023; Zhang et al. 2023; Park et al. 2023). Recent studies on stimulating interactive environment (Park et al. 2023; Jinxin et al. 2023), roleplaying (Zhang et al. 2023), and reasoning (Du et al. 2023; Liang et al. 2023), highlight the immense potential of multi-agent systems in tackling complex real-world challenges.

In the context of developing reliable and fault-tolerant systems, substantial research has been conducted on general machine learning and deep learning methodologies (Amin, Iqbal, and Shahbaz 2024; Bouhata et al. 2024; Myllyaho et al. 2022). However, thus far, limited work has been directed toward leveraging LLM-based multi-agent systems for such purposes. For instance, Liu et al. have introduced DyLAN, a dynamic network of LLM agents that optimizes collaboration for task-specific reliability, achieving notable improvements in decision-making tasks (Liu et al. 2024). Chacon-Chamorro et al. have proposed resilience metrics to measure and enhance the adaptability of LLM-augmented agents in cooperative environments, highlighting their ability to sustain functionality amid disruptions (Chacon-Chamorro et al. 2024). Additionally, several studies have proposed frameworks to develop multi-agent systems with fault-tolerant mechanisms such as AgentScopre by Gao et al. (Gao et al. 2024), and AgentMonitor by Chan et al. (Chan et al. 2024) that are capable of mitigating risks to improve overall system reliability.

Given the proliferation of studies on multi-agent LLM systems and their potential to be applied in diverse applications, in this work, we focus on the intersection of the fields above. Specifically, we study how the reliability and performance of a system that is composed of multiple LLM agents performing the same task is affected by different aggregation strategies.

# Methods

## Multi-agent architectures

As mentioned above, we focus on studying the effectiveness of various multi-agent architectures in solving a task not just from the aspect of accuracy but also from the aspect of consistency and reliability. As such, we consider a system that has multiple agents performing the same task with the same input and consider various strategies of aggregating the inputs to study which aggregation strategy results in the most consistent and reliable system output. While the aggregation methods may bear similarity with certain ensembling strategies found in literature, we highlight that we measure the effectiveness of ensembling from the aspect of maximizing reliability.

In this work, we compare six different architectures, inspired by architectures commonly studied in multi-agent systems literature, together with a simple baseline of using a single agent (Han et al. 2024; Zhang et al. 2024; Yang et al. 2024):

- **Single Agent (Baseline):** A single decision-making agent using an LLM.
- **Majority Voting:** Multiple agents independently generate responses, and the final response is generated based on majority voting for each agent's response.
- **Averaging:** Multiple agents independently generate responses, and the final response is generated by averaging the responses.

- **Decentralized:** Agents iteratively generate and refine responses until consensus is reached. An averaged/randomly selected plan is used if consensus is not reached within a fixed number of iterations in the resource allocation/QA problems, respectively.
- **Decentralized (Feedback):** Similar to Decentralized, but the agents incorporate feedback from previous iterations of other agents into their next response.
- **Spoke & Wheel:** Agents (spokes) independently generate responses, and a central agent (wheel) combines these into a final response.
- **Spoke & Wheel (Feedback):** Similar to Spoke & Wheel, but the central agent's plan is used as feedback to guide the agents in subsequent iterations.

Figure 1 illustrates a schematic of some of the different configurations we study in this paper.

## Experiments

In the following sections, we briefly discuss the setting of the benchmarks we evaluate the multi-agent architectures on.

### Resource Allocation

In this problem, LLM agents are tasked with allocating limited resources to satisfy demands across multiple regions, based on given demand and resource constraints. Each region has predefined demands for multiple resource types, and the total available resources are constrained. The task for the LLM agents is to maximize the overall satisfaction of regional demands (by meeting each region's demands) while adhering to resource constraints. An example of a simplified problem is shown in Table 1. In this example, we show two types of resource (water and food) demanded by three regions, and the total resources being constrained to 15 units of water and 10 units of food respectively. For this problem where the total demand is equals to the total resources, the optimal allocation is simply for the agent to distribute the resources exactly to each region's demand, without under- or over-allocating the resources to any single region.

| Region | Demand (Water, Food) |
|---|---|
| Region 1 | (5, 3) |
| Region 2 | (4, 2) |
| Region 3 | (6, 5) |
| **Total Resources** | Water: 15, Food: 10 |

Table 1: Example configuration of regions and resource constraints.

### Question Answering

We use the SQuAD 2.0 (Rajpurkar, Jia, and Liang 2018) dataset for the question-answering task. Along with the usual question-answering task, we added a formatting instruction to each question to see the ability of the LLM agents to follow these instructions correctly. The formatting instructions are randomly selected from pre-set instructions. These instructions are designed to be easy to be evaluated with a hard-coded evaluation method. See the Appendix for the list of added instructions.

### Topic Classification

For this experiment, we use the AG's news topic classification dataset (Zhang, Zhao, and LeCun 2015) as the task for the multi-agent LLM system. In this setup, each agent is given as input a new article and is tasked with classifying the article as one of the topics: "World", "Sports", "Business" and "Science/Technology". Rather than performing a numerical classification, we prompted the LLM agent to directly return the predicted topic in text.

### Text Summarization

For the text summarization task, we used the Xsum (Narayan, Cohen, and Lapata 2018) dataset. The Xsum dataset contains 226,711 BBC news articles paired with human-authored summaries. The task is to generate a summary that is similar to the human-authored summary, given one of the articles in the dataset.

## Evaluation Metrics

To evaluate the effectiveness and reliability of different architectures, we use two primary metrics: the performance metric (specific to each task) and a reliability metric that is common to all task.

### Performance Metrics

**Resource Allocation** For the resource allocation problem, the performance metric is the satisfaction, $S(R)$ - for a given region $R$ measures how well the allocated resources meet the demands of the region. For each resource $r$ demanded by region $R$, the satisfaction is capped at 100% and averaged across all resources as follows:

$$\mathrm{S}(R) = \frac{1}{N_R} \sum_{r \in \text{Resources}} \min\left(\frac{\text{Allocation}(R, r)}{\text{Demand}(R, r)}, 1\right), \quad (1)$$

where $N_R$ is the total resources demanded by region $R$.

**Question Answering** For the QA problem, the performance metrics are (1) the correctness of the answer, and (2) the instruction following accuracy. For the correctness metric, we used the F1 scores of the answers that are calculated using the method defined in Rajpurkar, Jia, and Liang (2018). For the instruction following accuracy, we used a hard-coded evaluation method to decide if an answer has correctly followed the instruction.

**Topic Classification** Similar to the Question Answering task above, since this benchmark has four only classes of topics, we used the conventional metric of accuracy in terms of exact word matching to score the final output of the multi-agent LLM system.

**Text Summarization** Following the original paper (Rajpurkar, Jia, and Liang 2018), we use the $F_1$ ROUGE (Lin and Hovy 2003) score to evaluate the similarity between the generated summaries and the human-authored summaries. We calculate the ROUGE-1 and ROUGE-2 scores, which measure the overlap of unigrams and bigrams, respectively, as well as the ROUGE-L, which is calculated based on the longest sequence of words common in two summaries.

### Reliability Metric

Additionally, we define reliability, $\kappa(\tau)$, as a metric that measures the reliability of the multi-agent LLM system to consistently achieve high performance across multiple, identical trials. For a given threshold $\tau$, the reliability is defined as the fraction of trials where the performance metric exceeds a certain threshold:

$$\kappa(\tau) = \frac{\sum_{t=1}^{T} \mathbb{I}\left(\mathrm{S}^{(t)} \geq \tau\right)}{T}, \tag{2}$$

where $T$ is the total number of trials, $\mathbb{I}(\cdot)$ is an indicator function (1 if condition is true, 0 otherwise), and $\mathrm{S}^{(t)}$ is the performance metric in trial $t$.

## Experimental settings

In this section, we explain the parameters specific to each experiment and additional details of our experiments are also shown in the Appendix. We instantiate all agents using GPT-4o-mini with default hyperparameters, set the maximum number of feedback turns to 5 for architectures with feedback mechanisms, and the number of independent trials to 30.

### Resource Allocation

We compare across 3 different scales/complexities (small, medium, large) of resources allocation problems. For each complexity of the problem, we further conduct 3 settings: a) the total resources exactly match the total demand (equal), b) the total resources are less than the total demand (lack) and c) the total resources are greater than the total demand (excess).

Given the stochastic nature of the LLMs, we observed that there are instances when the LLM agent does not respond in the specified template or hallucinates additional regions/resources that are not provided. In those cases, we replace the proposed solution of that specific agent with a default template of allocation, with all allocations set to 0. This is to emulate the scenario of one agent's failure in a multi-agent system and allow us to better study the benefits of having multiple agent in creating a more fault-tolerant system.

### Question Answering

We randomly selected 30 questions from the SQuAD 2.0 test dataset. For each of the questions, we added randomly-sampled formatting instructions as explained in the previous sections. We prompted the agents to generate the answer to the question first and then prompted them to follow the instructions to add the required information in the specific format.

### Topic Classification

In this experiment, we randomly sampled 100 articles from the AG's news test dataset and prompted the agents to classify the news articles into one of the four classes. Since this task, doesn't require a structured output that is required in the resource allocation problem, any output text that is not an exact match to one of the four topic classes is considered to be an incorrect classification.

### Text Summarization

We randomly sampled 50 articles from the test split in the Xsum dataset. For each of the articles, we prompted the agents to make a one-line concise summary of the article. The prompts used to generate the summaries are shown in the Appendix.

## Results

### Resource Allocation

Figure 2 presents the reliability of different aggregation mechanisms across a range of satisfaction thresholds for the medium-scale allocation problem under three resource settings: equal, lack, and excess. Comparing these figures, we observe a general trend where reliability in achieving high satisfaction thresholds decreases as the availability of resources shifts from excess to scarcity. This observation validates the proposed framework, as the challenge of resource allocation naturally increases when resources are limited, requiring a higher degree of effective allocation.

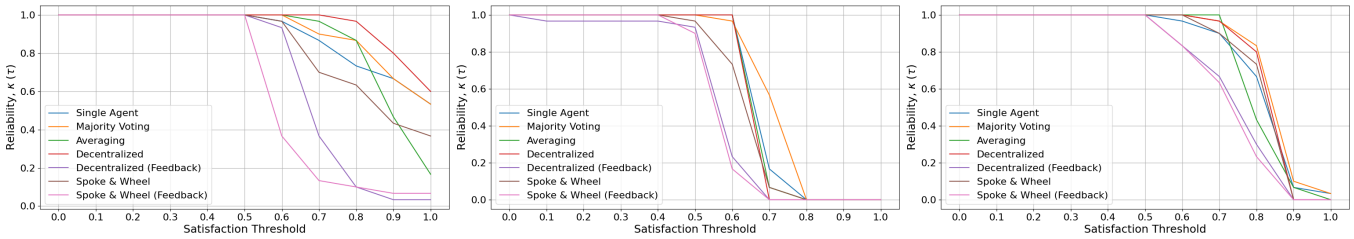
More interestingly, across all settings, aggregation strategies such as Majority voting and Decentralized approaches consistently form the Pareto front of reliability curves. This indicates that these strategies are more reliable in achieving higher satisfaction thresholds. A greater Pareto front of the reliability curve reflects the effectiveness of an aggregation strategy, as it demonstrates a method's ability to consistently achieve higher satisfaction thresholds. In contrast, strategies incorporating feedback mechanisms rank the lowest in reliability across all settings, while the remaining strategies lie in between.

To quantitatively assess whether these observations hold true and generalize across other resource allocation problems of varying complexities and configurations, we rank the aggregation strategies by computing the area under the reliability curves (AURC). An ideal aggregation strategy would achieve an AURC of 1 in scenarios where resources meet or exceed demand, signifying its ability to consistently satisfy resource requirements across all trials. Table 2 summarizes the computed AUC values for all experiments.

For small-scale problems, no single method significantly outperforms others. However, as the problem complexity increases to medium and large scales, majority voting emerges as a leading aggregation strategy. One exception is observed in the Medium-Equal configuration, where the Decentralized strategy achieves a slightly higher AURC, with majority voting ranking second. Reliability curves for additional experiments are included in the Appendix for reference.

Table 2: Summary of Area Under Reliability Curves values for different aggregation methods, for the Resource Allocation, Question Answering, Classification and Text Summarization tasks.

| | | | Single Agent | Majority Voting | Averaging | Decentralized | Feedback Decentralized | Spoke & Wheel | Feedback Spoke & Wheel |
|---|---|---|---|---|---|---|---|---|---|
| Resource Allocation | Small | Equal | 0.999 | **1.000** | 0.999 | **1.000** | 0.999 | **1.000** | 0.988 |
| | | Lack | 0.680 | 0.688 | 0.654 | 0.699 | 0.688 | 0.702 | **0.706** |
| | | Excess | **0.709** | 0.669 | 0.705 | 0.665 | 0.702 | 0.669 | 0.666 |
| | Medium | Equal | 0.897 | 0.915 | 0.890 | **0.961** | 0.695 | 0.835 | 0.627 |
| | | Lack | 0.689 | **0.691** | 0.668 | 0.673 | 0.534 | 0.623 | 0.532 |
| | | Excess | 0.802 | **0.826** | 0.801 | 0.816 | 0.729 | 0.805 | 0.722 |
| | Large | Equal | 0.640 | **0.711** | 0.626 | 0.623 | 0.531 | 0.567 | 0.447 |
| | | Lack | 0.564 | **0.671** | 0.593 | 0.574 | 0.532 | 0.559 | 0.424 |
| | | Excess | 0.672 | **0.816** | 0.682 | 0.648 | 0.582 | 0.579 | 0.450 |
| Question Answering | | Correctness | 0.722 | – | – | **0.732** | 0.725 | 0.729 | 0.670 |
| | | Instruction Following | 0.824 | – | – | **0.838** | 0.811 | 0.801 | 0.744 |
| Classification | | Accuracy | 0.831 | **0.833** | – | **0.833** | 0.826 | 0.704 | 0.686 |
| Text Summarization | | ROUGE-1 | **0.290** | – | – | 0.289 | 0.289 | 0.284 | 0.287 |
| | | ROUGE-2 | 0.089 | – | – | 0.087 | 0.089 | 0.087 | **0.090** |
| | | ROUGE-L | **0.219** | – | – | 0.217 | 0.218 | 0.213 | 0.218 |




(a) Demand = Resources (b) Demand > Resources (c) Demand < Resources

Figure 2: [Allocation] Reliability of different multi-agent aggregation mechanisms to consistently meet a satisfaction threshold for medium-scale problems with (a) equal demand and resources, (b) demand greater than resources, and (c) demand less than resources.

## Question Answering

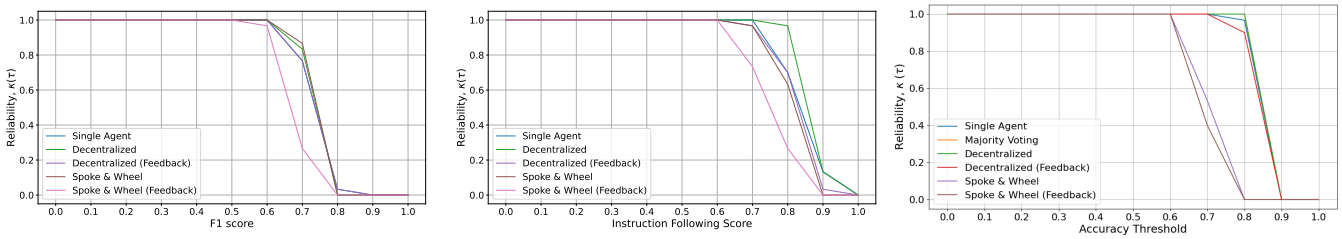
Figure 3(a) and (b) presents the reliability of the correctness (F1 score) and the instruction following, for different aggregation mechanisms. Note that in these experiments, we do not compare the majority voting or average voting strategies as that requires further processing of each agent's text output in order to decode the final output and we leave this study as a potential future research. Similar to the results for the resource allocation problem, aggregation strategies such as Decentralized and Spoke &Wheel consistently achieve higher reliability given a threshold, and feedback mechanisms get relatively lower reliability. Additionally, Table 2 presents the area under the reliability curve for various aggregation strategies. Decentralized approach achieve highest value among all the methods.

## Topic Classification

Figure 3(c) and the results in Table 2 presents a similar trend as well when we observed the results of the experiment. In these set of experiments, we excluded Average Voting strategy, but included the Majority Voting as it is relatively straightforward to take a majority of a predicted class, but non-trivial to take an average of the predicted class without access to predicted class probabilities. From Figure 3(c), we can see that Decentralized, Majority Voting (occluded in the figure) and Single Agent architectures mainly forms the forefront of the Pareto, while the Spoke & Wheel architecture performs the worse, both in terms of reliability and accuracy threshold. Analyzing the AURC of the different method, once again, we see that Majority Voting and Decentralized architectures yields the best AURCs.

## Text Summarization

Similar to the Question Answering experiments, we exclude the study of Majority Voting and Average Voting from these experiments. Figure 4 presents the reliability of the summary qualities measured with the R1, R2, and RL ROUGE scores. We observe that the scores are agnostic to aggregation strategies, unlike in other problems. Specifically, the Decentralized approach did not improve the score, and the feedback




(a) F1 score (b) Instruction following (c) Classification Accuracy

Figure 3: [QA]/[Topic Classification] Reliability of different aggregation methods to consistently achieve (a) F1 score thresholds, and (b) instruction following score thresholds, and (c) classification accuracy thresholds.

methods did not degrade the scores compared to the naive Single Agent approach.

## Discussion

While majority voting is a well-established aggregation strategy in ensemble learning for machine learning models, the findings of this study are unique for several reasons. First, unlike traditional strategies that primarily focus on improving accuracy in predictions or decisions, our work emphasizes reliability—the ability to consistently perform well, particularly at high-performance thresholds. This shift represents a significant departure from conventional evaluation metrics in aggregation studies and places a spotlight on consistency as a critical factor in multi-agent collaboration.

More importantly, while many recent approaches have proposed and leveraged feedback mechanisms—such as iterative refinement, cascading refinement, or self-critic approaches—to enhance performance, our results challenge these assumptions. Interestingly, our preliminary findings demonstrate that simpler architectures, such as Majority Voting or Decentralized aggregation, outperform more complex feedback-based mechanisms in terms of reliability on most of the experiments, with the exception of text summarization. We hypothesize that this counterintuitive outcome arises from the inherent stochasticity of LLM-based agents and their connectivity within the system architecture. In multi-agent systems with highly connected architectures or feedback loops, any erroneous or suboptimal output from an agent risks being propagated to other agents. This amplification of errors can destabilize the entire system, especially when feedback cycles propagate inaccuracies over multiple iterations.

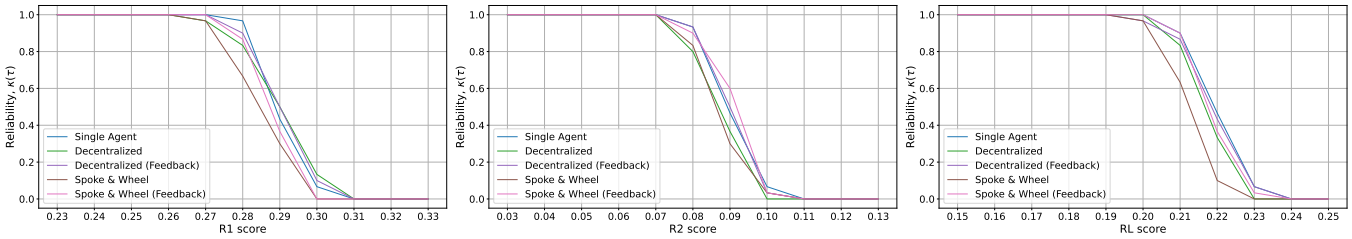
In contrast, having multiple decentralized, independent agents,coupled with an aggregation mechanism that combines their decisions without direct inter-agent influence, introduces redundancy that can enhance reliability. This conclusion aligns with conventional principles of building robust systems through redundancy, where duplicating components ensures continued functionality even in the presence of component failures. By reducing the likelihood of error propagation, decentralized strategies mitigate the risks associated with over-connected feedback-based systems. This insight highlights the potential limitations of feedback-based strategies in having multi-agent collaborate to complete tasks, where achieving consistent high reliability is critical. Our findings emphasize that simplicity in aggregation design can sometimes yield better results, particularly for scenarios where consistent reliability, rather than just accuracy, is paramount.

Interestingly, for text summarization tasks, our experiments showed very little difference between the aggregation methods. Whether we used Majority Voting, Decentralized, or Feedback-based approaches, the results were almost the same. We posit that this is because of how summarization is currently evaluated, mainly using ROUGE scores. These scores compare the generated summaries to reference summaries written by humans, focusing on how much the words overlap. However, ROUGE may not fully capture other important aspects like how well the summary conveys the meaning or flows naturally. Because of this, it's hard to see the impact of different aggregation methods in tasks such as summarization, where the generated output is not easily evaluated objectively. This highlights how the choice of evaluation metric can affect the results, especially for tasks where quality is more subjective or harder to measure.

### Additional considerations

Based on the results shown, we can see that Majority Voting and Decentralized architectures are strong candidates that can result in reliable multi-agent systems. However, one major drawback of majority / average voting is that they're only naively applicable to scenarios where the outputs are numerical or where a majority could be counted. In situations where the output of multiple agents are lexically different but semantically similar, an aggregation strategy which can mimic the properties of majority voting while maintaining the reliability of the system will be an interesting avenue for future work.

Additionally, while this study evaluates three datasets across three distinct tasks, we acknowledge that the results are not exhaustive. For instance, the performance of individual agents within the system could likely be enhanced through additional fine-tuning of prompts, the introduction of stricter guardrails, or methods to enforce more structured outputs. Despite these potential improvements, we anticipate that our primary observations will remain valid: improving the reliability of individual agents is expected to directly enhance the reliability of the overall system. This underscores the importance of focusing on both individual




(a) ROUGE-1 (b) ROUGE-2 (c) ROUGE-L

Figure 4: [Summarization] Reliability of different aggregation methods to consistently achieve (a) ROUGE-1, (b) ROUGE-2, and (c) ROUGE-L score thresholds.

agent optimization and robust aggregation strategies when designing dependable multi-agent systems.

## Conclusion

Our study highlights the critical role of redundancy and fault tolerance in designing reliable multi-agent systems, particularly in contexts where consistent performance is paramount. Through experiments on multiple resource allocation problems, we observe that simple strategies, such as majority voting, exhibit the best consistency, reliably maintaining performance above a certain threshold over a set number of trials. This finding underscores the value of straightforward yet effective approaches in achieving robustness, even when more complex architectures might appear promising. Future work will extend the exploration of multi-agent architectures to broader benchmarks and domains, including cases where both inputs and outputs are potentially unstructured, and strategies such as majority voting are intractable to implement. By addressing these challenges, we aim to further refine and expand the applicability of LLM-driven multi-agent systems, paving the way for more versatile and dependable solutions.

## References


Amin, A. A.; Iqbal, M. S.; and Shahbaz, M. H. 2024. Development of intelligent fault-tolerant control systems with machine learning, deep learning, and transfer learning algorithms: a review. *Expert Systems with Applications*, 238: 121956.

Awad, E.; Booth, R.; Tohmé, F.; and Rahwan, I. 2017. Judgement aggregation in multi-agent argumentation. *Journal of Logic and Computation*, 27(1): 227–259.

Besta, M.; Blach, N.; Kubicek, A.; Gerstenberger, R.; Podstawski, M.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Niewiadomski, H.; Nyczyk, P.; et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 17682–17690.

Bouhata, D.; Moumen, H.; Mazari, J. A.; and Bounceur, A. 2024. Byzantine fault tolerance in distributed machine learning: a survey. *Journal of Experimental & Theoretical Artificial Intelligence*, 1–59.

Chacon-Chamorro, M.; Giraldo, L. F.; Quijano, N.; Vargas-Panesso, V.; González, C.; Pinzón, J. S.; Manrique, R.; Ríos, M.; Fonseca, Y.; Gómez-Barrera, D.; et al. 2024. Cooperative resilience in artificial intelligence multiagent systems. *arXiv preprint arXiv:2409.13187*.

Chan, C.-M.; Yu, J.; Chen, W.; Jiang, C.; Liu, X.; Shi, W.; Liu, Z.; Xue, W.; and Guo, Y. 2024. AgentMonitor: A Plug-and-Play Framework for Predictive and Secure Multi-Agent Systems. *arXiv preprint arXiv:2408.14972*.

Chen, W.; Ma, X.; Wang, X.; and Cohen, W. W. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Dohan, D.; Xu, W.; Lewkowycz, A.; Austin, J.; Bieber, D.; Lopes, R. G.; Wu, Y.; Michalewski, H.; Saurous, R. A.; Sohl-Dickstein, J.; et al. 2022. Language model cascades. *arXiv preprint arXiv:2207.10342*.

Du, Y.; Li, S.; Torralba, A.; Tenenbaum, J. B.; and Mordatch, I. 2023. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*.

Gao, D.; Li, Z.; Pan, X.; Kuang, W.; Ma, Z.; Qian, B.; Wei, F.; Zhang, W.; Xie, Y.; Chen, D.; et al. 2024. Agentscope: A flexible yet robust multi-agent platform. *arXiv preprint arXiv:2402.14034*.

Ge, Z.; Song, Z.; Ding, S. X.; and Huang, B. 2017. Data mining and analytics in the process industry: The role of machine learning. *Ieee Access*, 5: 20590–20616.

Guo, Q.; and Zhang, M. 2009. A novel approach for multi-agent-based intelligent manufacturing system. *Information Sciences*, 179(18): 3079–3090.

Gupta, N.; and Vaidya, N. H. 2020. Fault-tolerance in distributed optimization: The case of redundancy. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, 365–374.

Han, S.; Zhang, Q.; Yao, Y.; Jin, W.; Xu, Z.; and He, C. 2024. LLM multi-agent systems: Challenges and open problems. *arXiv preprint arXiv:2402.03578*.

Hu, S.; Lu, C.; and Clune, J. 2024. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*.

Jin, Z.; and Lu, W. 2023. Tab-cot: Zero-shot tabular chain of thought. *arXiv preprint arXiv:2305.17812*.

Jinxin, S.; Jiabao, Z.; Yilei, W.; Xingjiao, W.; Jiawen, L.; and Liang, H. 2023. Cgmi: Configurable general multi-agent interaction framework. *arXiv preprint arXiv:2308.12503*.

Lang, N.; Moonen, H.; Srour, J.; and Zuidwijk, R. 2008. Multi agent systems in logistics: a literature and state-of-the-art review. *ERIM report series research in management Erasmus Research Institute of Management*, (ERS-2008-043-LIS).

Liang, T.; He, Z.; Jiao, W.; Wang, X.; Wang, Y.; Wang, R.; Yang, Y.; Shi, S.; and Tu, Z. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*.

Lin, C.-Y.; and Hovy, E. 2003. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 150–157.

Liu, Z.; Zhang, Y.; Li, P.; Liu, Y.; and Yang, D. 2024. A dynamic LLM-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*.

Long, J. 2023. Large language model guided tree-of-thought. *arXiv preprint arXiv:2305.08291*.

Myllyaho, L.; Raatikainen, M.; Männistö, T.; Nurminen, J. K.; and Mikkonen, T. 2022. On misbehaviour and fault tolerance in machine learning systems. *Journal of Systems and Software*, 183: 111096.

Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In Riloff, E.; Chiang, D.; Hockenmaier, J.; and Tsujii, J., eds., *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1797–1807. Brussels, Belgium: Association for Computational Linguistics.

Park, J. S.; O'Brien, J.; Cai, C. J.; Morris, M. R.; Liang, P.; and Bernstein, M. S. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, 1–22.

Rajpurkar, P.; Jia, R.; and Liang, P. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In Gurevych, I.; and Miyao, Y., eds., *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 784–789. Melbourne, Australia: Association for Computational Linguistics.

Shakshuki, E.; and Reid, M. 2015. Multi-agent system applications in healthcare: current technology and future roadmap. *Procedia Computer Science*, 52: 252–261.

Wang, W.; Dong, L.; Cheng, H.; Liu, X.; Yan, X.; Gao, J.; and Wei, F. 2024. Augmenting language models with long-term memory. *Advances in Neural Information Processing Systems*, 36.

Xu, Z.; and Saleh, J. H. 2021. Machine learning for reliability engineering and safety applications: Review of current status and future opportunities. *Reliability Engineering & System Safety*, 211: 107530.

Yang, Y.; Peng, Q.; Wang, J.; and Zhang, W. 2024. Multi-LLM-Agent Systems: Techniques and Business Perspectives. *arXiv preprint arXiv:2411.14033*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Zhang, G.; Yue, Y.; Sun, X.; Wan, G.; Yu, M.; Fang, J.; Wang, K.; and Cheng, D. 2024. G-designer: Architecting multi-agent communication topologies via graph neural networks. *arXiv preprint arXiv:2410.11782*.

Zhang, J.; Xu, X.; Zhang, N.; Liu, R.; Hooi, B.; and Deng, S. 2023. Exploring collaboration mechanisms for llm agents: A social psychology view. *arXiv preprint arXiv:2310.02124*.

Zhang, X.; Zhao, J. J.; and LeCun, Y. 2015. Character-level Convolutional Networks for Text Classification. In *NIPS*.

# Appendix

## Details on resource allocation problems

The following tables shows the specific numbers we used for the different resource allocation problems. The numbers are selected arbitrarily and are only tune to ensure that the total resource either meet, exceed or is less than the specific demand for each of the setting.

| Region | Demand (Water, Food) |
|---|---|
| **Small Scale - Balanced Resource Allocation** | |
| Region 1 | (8, 6) |
| Region 2 | (12, 7) |
| Region 3 | (6, 10) |
| **Total Resources** | (Water: 26, Food: 23) |
| **Small Scale - Insufficient Resources** | |
| Region 1 | (8, 6) |
| Region 2 | (12, 7) |
| Region 3 | (6, 10) |
| **Total Resources** | (Water: 20, Food: 15) |
| **Small Scale - Excess Resources** | |
| Region 1 | (8, 6) |
| Region 2 | (12, 7) |
| Region 3 | (6, 10) |
| **Total Resources** | (Water: 40, Food: 35) |

Table 3: Comparison of Small Scale Scenarios: Balanced, Insufficient, and Excess Resource Allocation.

| Region | Demand (Water, Food) |
|---|---|
| **Medium Scale - Balanced Resource Allocation** | |
| Region 1 | (10, 8) |
| Region 2 | (15, 5) |
| Region 3 | (5, 12) |
| Region 4 | (8, 3) |
| Region 5 | (6, 10) |
| Region 6 | (12, 6) |
| **Total Resources** | (Water: 56, Food: 44) |
| **Medium Scale - Insufficient Resources** | |
| Region 1 | (10, 8) |
| Region 2 | (15, 5) |
| Region 3 | (5, 12) |
| Region 4 | (8, 3) |
| Region 5 | (6, 10) |
| Region 6 | (12, 6) |
| **Total Resources** | (Water: 40, Food: 30) |
| **Medium Scale - Excess Resources** | |
| Region 1 | (10, 8) |
| Region 2 | (15, 5) |
| Region 3 | (5, 12) |
| Region 4 | (8, 3) |
| Region 5 | (6, 10) |
| Region 6 | (12, 6) |
| **Total Resources** | (Water: 60, Food: 50) |

Table 4: Comparison of Medium Scale Scenarios: Balanced, Insufficient, and Excess Resource Allocation.

## Prompts

In this section, we provide the details of the prompts we used for prompting each LLM agent.

| Region | Demand (Water, Food, Medicine) |
|---|---|
| **Large Scale - Balanced Resource Allocation** | |
| Region 1 | (5, 3, 2) |
| Region 2 | (7, 5, 3) |
| Region 3 | (4, 6, 5) |
| Region 4 | (8, 4, 3) |
| Region 5 | (6, 7, 4) |
| Region 6 | (10, 5, 6) |
| Region 7 | (3, 2, 1) |
| Region 8 | (9, 8, 7) |
| Region 9 | (7, 6, 5) |
| **Total Resources** | (Water: 59, Food: 46, Medicine: 36) |
| **Large Scale - Insufficient Resources** | |
| Region 1 | (5, 3, 2) |
| Region 2 | (7, 5, 3) |
| Region 3 | (4, 6, 5) |
| Region 4 | (8, 4, 3) |
| Region 5 | (6, 7, 4) |
| Region 6 | (10, 5, 6) |
| Region 7 | (3, 2, 1) |
| Region 8 | (9, 8, 7) |
| Region 9 | (7, 6, 5) |
| **Total Resources** | (Water: 45, Food: 35, Medicine: 25) |
| **Large Scale - Excess Resources** | |
| Region 1 | (5, 3, 2) |
| Region 2 | (7, 5, 3) |
| Region 3 | (4, 6, 5) |
| Region 4 | (8, 4, 3) |
| Region 5 | (6, 7, 4) |
| Region 6 | (10, 5, 6) |
| Region 7 | (3, 2, 1) |
| Region 8 | (9, 8, 7) |
| Region 9 | (7, 6, 5) |
| **Total Resources** | (Water: 80, Food: 65, Medicine: 50) |

Table 5: Comparison of Large Scale Scenarios: Balanced, Insufficient, and Excess Resource Allocation.

### Resource Allocation Prompt

Available resources:
{environment.resources}.
Regions and their demands:
{environment.regions}.

Provide an allocation plan (dictionary) where each resource (e.g., 'water', 'food') is mapped to another (dictionary) region IDs and allocated amounts. Ensure allocations respect the total available resources and your goal is to meet all the demand or as much demand as possible.

As an example, respond directly in the format:
{{'water': {'region1': 1, 'region2': 5, 'region3': 1}, 'food': {'region1': 3, 'region2': 3, 'region3': 3}}}.

Think step-by-step, and ensure all the resources and regions are included in your response.
Do not hallucinate additional regions or resources.
Do not add any additional response and do not respond with JSON format.

### Resource Allocation Prompt for agents with feedback

Available resources:
{environment.resources}.
Regions and their demands:
{environment.regions}.

These are proposed plans from other planners in previous rounds. Use these to the next plan if applicable
{previous plans}.

Provide an allocation plan (dictionary) where each resource (e.g., 'water', 'food') is mapped to another (dictionary) region IDs and allocated amounts. Ensure allocations respect the total available resources and your goal is to meet all the demand or as much demand as possible.

As an example, respond directly in the format:
{{'water': {'region1': 1, 'region2': 5, 'region3': 1}, 'food': {'region1': 3, 'region2': 3, 'region3': 3}}}.

Think step-by-step, and ensure all the resources and regions are included in your response.
Do not hallucinate additional regions or resources.
Do not add any additional response and do not respond with JSON format.

### Topic Classification Prompt

Classify the following news article into one of these topics: World, Sports, Business, Science/Technology.

Article: {article}

Respond with only the topic name (e.g., "World").
Do not add additional explanations or responses.

### Topic Classification Prompt with feedback

Classify the following news article into one of these topics: World, Sports, Business, Science/Technology.

Article: {article}

Previous suggestions and feedback from other agents:
{Feedback}

Based on this feedback and the given article, refine your classification. Respond with only the topic name (e.g., "World"). Do not add additional explanations or responses.

### Question Answering System Prompt

You are an intelligent assistant that answers questions only based on the knowledge provided by the user. Answer in one or several words or a phrase, as short as possible, in brackets []. There could be additional requests from the user shown as "Additional Instruction". Please make sure to follow those instructions. It is possible that the context does not have the information needed to answer the question. In that case, answer NA. We don't need a period at the end of the answer. There could be other assistants working on the same task, and they may have different opinions. Please check their answers if provided, and use them to make your final answer better. You don't have to use others' answers if you don't agree with them.

* Example
Context: Tokyo, Japan's capital, blends tradition and modernity with landmarks like Meiji Shrine and Tokyo Skytree, world-class cuisine, and neighborhoods like Shibuya and Asakusa.
Question: What city is Japan's capital?
Additional Instruction: Add some explanations in less than 15 words.
Answer: [Tokyo] Tokyo is the capital of Japan and it blends tradition and modernity.

**Question Answering Prompt**

Context: {`context`}
Question: {`question`}.
Additional Instruction: {`instruction`}.
Answer:

**Question Answering Prompt with feedback**

Context: {`context`}
Question: {`question`}.
Additional Instruction: {`instruction`}.

These are proposed answers from other assistants in previous rounds. Use these in the next answer if applicable:
Round {`trial`}, Agent {`agent_index`}'s Answer: {`answer[trial][agent_index]`}
## Loop over `trial` and `agent_index`

Answer:

**Text Summarization System Prompt**

You are a professional writer who is able to summarize a given document in one short sentence. Make it concise but informative. There could be other writers working on the same summarization task, and they may have different opinions. Please check their answers if provided, and use it to make your final summarization better, if needed. You don't have to use others' answers if you don't agree with them.

* Example
Document: The Central Bank announced a 0.5% increase in interest rates, bringing the benchmark rate to 5.5%, the highest level in over a decade. The decision aims to curb persistently high inflation, which remains above the target of 2%. Economists warn that higher borrowing costs may slow consumer spending and business investments, potentially impacting economic growth. However, Central Bank Governor Lisa Martinez emphasized that controlling inflation is crucial to stabilizing the economy in the long term. Financial markets responded with mixed reactions, as stock indices dropped while bond yields rose. Analysts expect further rate hikes if inflation does not show signs of easing in the coming months.
Summary: The Central Bank raised interest rates to 5.5% to tackle inflation, sparking concerns over potential economic slowdown.

**Text Summarization Prompt**

Document: {`document`}
Summary:

**Text Summarization Prompt with feedback**

Document: {`document`}

These are proposed summaries from other assistants in previous rounds. Use these in the next summary if applicable:
Round {`trial`}, Agent {`agent_index`}'s Summary: {`summary[trial][agent_index]`}
## Loop over `trial` and `agent_index`
Summary:

## Additional experimental results

In this section, we present the reliability plots for the other experiments that were not shown in the main paper.


Figure 5: [Allocation] Reliability of different multi-agent aggregation mechanisms to consistently meet a satisfaction threshold for small scale problem with equal demand and resources for the small scale problem.

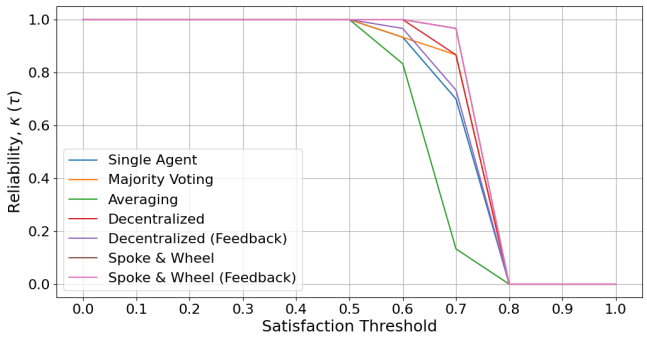



Figure 6: [Allocation] Reliability of different multi-agent aggregation mechanisms to consistently meet a satisfaction threshold for small scale problem with demand greater than resources for the small scale problem.

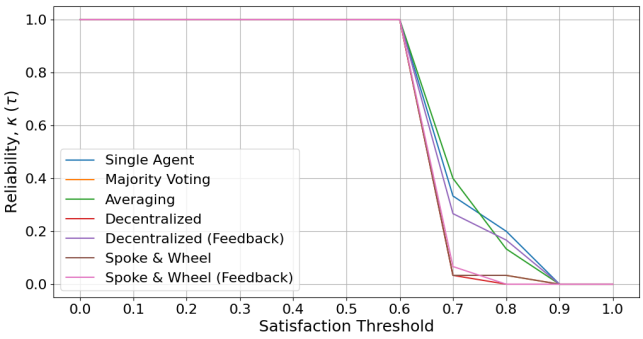



Figure 7: [Allocation] Reliability of different multi-agent aggregation mechanisms to consistently meet a satisfaction threshold for small scale problem with demand less than resources for the small scale problem.

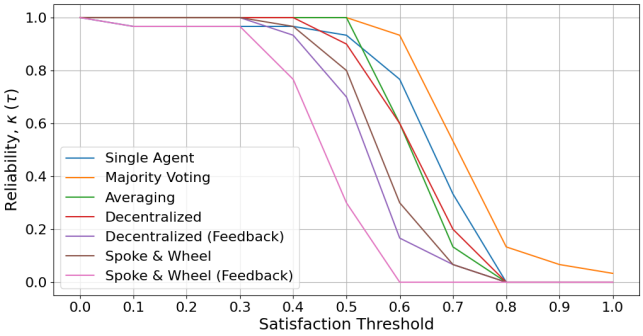



Figure 8: [Allocation] Reliability of different multi-agent aggregation mechanisms to consistently meet a satisfaction threshold for large scale problem with equal demand and resources for the large scale problem.

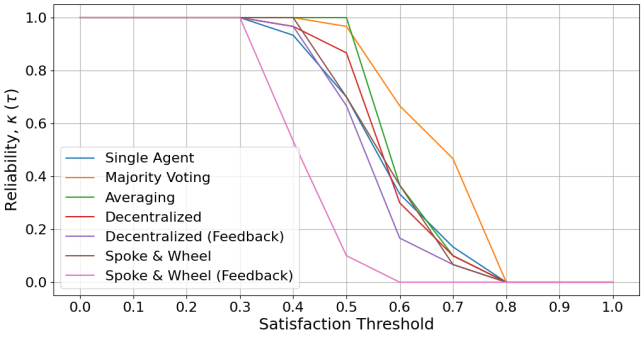



Figure 9: [Allocation] Reliability of different multi-agent aggregation mechanisms to consistently meet a satisfaction threshold for large scale problem with demand greater than resources for the large scale problem.

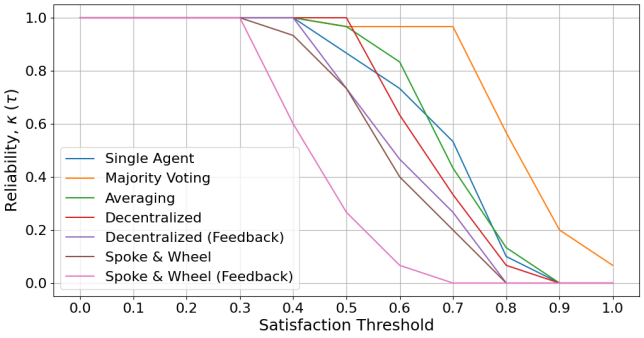



Figure 10: [Allocation] Reliability of different multi-agent aggregation mechanisms to consistently meet a satisfaction threshold for large scale problem with demand less than resources for the large scale problem.

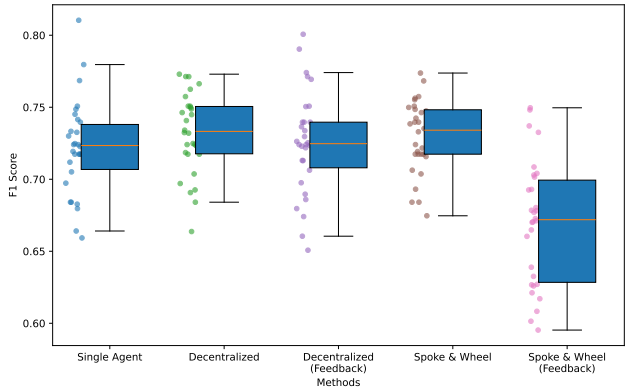



Figure 11: [QA] Comparison of F1 scores achieved by different methods. The box plots represent the distribution of scores across 30 test runs for each method, with the median and interquartile ranges shown. Individual scores for each test run are displayed as scatter points on the left side of each box plot, illustrating the spread of results.

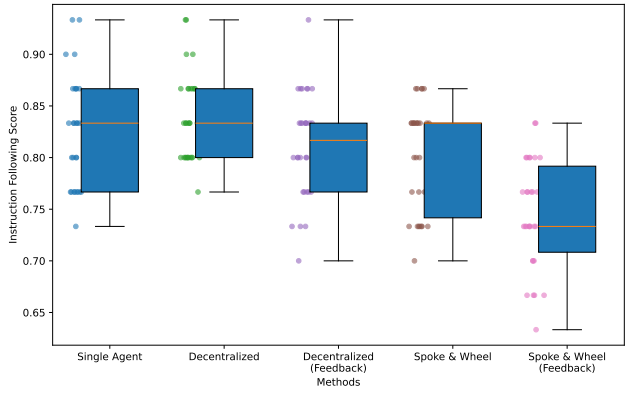



Figure 12: [QA] Comparison of instruction-following scores achieved by different methods. The box plots represent the distribution of scores across 30 test runs for each method, with the median and interquartile ranges shown. Individual scores for each test run are displayed as scatter points on the left side of each box plot, illustrating the spread of results.