Highlights

A Comprehensive Survey on Multi-Agent Cooperative Decision-Making: Scenarios, Approaches, Challenges and Perspectives

Weiqiang Jin, Hongyang Du, Biao Zhao, Xingwu Tian, Bohang Shi, Guang Yang

- Provides a comprehensive survey of multi-agent decision-making methods.
- Analyzes key simulation environments for multi-agent reinforcement learning.
- Investigate decision-making approaches, including MARL and large language models.
- Identifies challenges and future research directions in multi-agent collaboration.
- Reviews real-world applications in transportation, aerial systems, and automation.

A Comprehensive Survey on Multi-Agent Cooperative Decision-Making: Scenarios, Approaches, Challenges and Perspectives

Weiqiang Jin[a], Hongyang Du[b], Biao Zhao[a,*], Xingwu Tian[a], Bohang Shi[a], Guang Yang[c,d,e,f,*]

[a]School of Information and Communications Engineering, Xi'an Jiaotong University, Innovation Harbour, Xi'an, 710049, Shaanxi, China
[b]Department of Electrical and Electronic Engineering, The University of Hong Kong (HKU), Hong Kong, Hong Kong, China
[c]Bioengineering Department and Imperial-X, Imperial College London, London, W12 7SL, UK
[d]National Heart and Lung Institute, Imperial College London, London, SW7 2AZ, UK
[e]Cardiovascular Research Centre, Royal Brompton Hospital, London, SW3 6NP, UK
[f]School of Biomedical Engineering & Imaging Sciences, King's College London, London, WC2R 2LS, UK

Abstract

With the rapid development of artificial intelligence, intelligent decision-making techniques have gradually surpassed human levels in various human-machine competitions, especially in complex multi-agent cooperative task scenarios. Multi-agent cooperative decision-making involves multiple agents working together to complete established tasks and achieve specific objectives. These techniques are widely applicable in real-world scenarios such as autonomous driving, drone navigation, disaster rescue, and simulated military confrontations. This paper begins with a comprehensive survey of the leading simulation environments and platforms used for multi-agent cooperative decision-making. Specifically, we provide an in-depth analysis for these simulation environments from various perspectives, including task formats, reward allocation, and the underlying technologies employed. Subsequently, we provide a comprehensive overview of the mainstream intelligent decision-making approaches, algorithms and models for multi-agent systems (MAS). These approaches can be broadly categorized into five types: rule-based (primarily fuzzy logic), game theory-based, evolutionary algorithms-based, deep multi-agent reinforcement learning (MARL)-based, and large language models (LLMs) reasoning-based. Given the significant advantages of MARL and LLMs-based decision-making methods over the traditional rule, game theory, and evolutionary algorithms, this paper focuses on these multi-agent methods utilizing MARL and LLMs-based techniques. We provide an in-depth discussion of these approaches, highlighting their methodology taxonomies, advantages, and drawbacks. Further, several prominent research directions in the future and potential challenges of multi-agent cooperative decision-making are also detailed.

Keywords:
Intelligent decision-making, Multi-agent systems, Multi-agent cooperative environments, Multi-agent reinforcement learning, Large language models.

1. Introduction

1.1. Research Backgrounds of Multi-Agent Decision-Making

With the continuous advancement of science and technology, intelligent decision-making technology has made rapid progress. These technologies have gradually surpassed human capabilities in various human-machine game competitions, even exceeding the top human levels. Over the past few decades, especially following the successful application of Deep Q-Networks (DQN) [**?** **?**] in the Arita game and the victories of AlphaGo and AlphaZero [**?** **?**] over top human opponents, these landmark achievements have significantly propelled the advancement of intelligent decision-making research.

To meet the growing complexity of real-world applications and the increasing demand for more sophisticated, reliable, and efficient intelligent systems, multi-

*Corresponding authors: Biao Zhao and Guang Yang.
Email addresses: weiqiangjin@stu.xjtu.edu.cn (Weiqiang Jin), duhy@hku.hk (Hongyang Du), biaozhao@xjtu.edu.cn (Biao Zhao), txw_xjtu@163.com (Xingwu Tian), Bh_567@stu.xjtu.edu.cn (Bohang Shi), g.yang@imperial.ac.uk (Guang Yang)

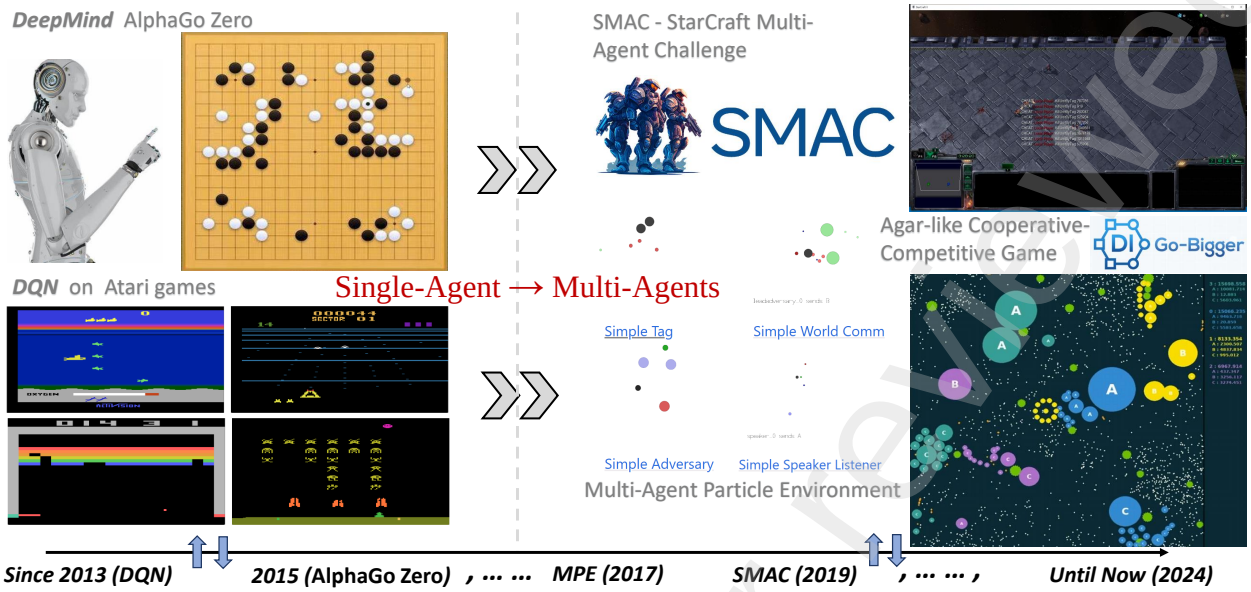


Figure 1: An overview of the evolution of scenarios and methods in decision-making from single-agent to multi-agent systems.

agent cooperative decision-making has rapidly evolved from simple single-agent scenarios [? ? ? ?]. Multi-agent cooperative decision-making is a crucial subfield within machine learning (ML) [?] and artificial intelligence (AI) [?]. It involves multiple interacting agents working together to complete established tasks across diverse well-designed dynamic simulated environments and various complex real-world systems.

As depicted in Figure 1, the evolution research progress from single-agent to multi-agent decision-making systems, along with methodological comparisons, highlights that this rapidly advancing field is a crucial step toward achieving human-level AI and the Artificial General Intelligence (AGI) age. Multi-agent cooperative decision-making has a wide range of practical applications and many fundamental theoretical works across various domains. The service scenarios are extensive, encompassing smart agriculture management [? ?], intelligent collaborative robots [? ? ? ?], self-driving collaborative obstacle avoidance [? ? ?], autonomous navigation [? ? ?] as well as joint rescue tasks [? ?]. Correspondingly, considering the rapid pace of technological advancement and the multi-faceted needs of the real world, in this work, we focus on the comprehensive study of multi-agent cooperative decision-making.

1.2. Overview of Previous Multi-Agent Surveys

Concurrent with the fast-paced advancements in multi-agent cooperative decision-making, there has been a marked increase in systematic literature reviews in this domain [? ? ? ?]. These reviews have covered a wide range of topics, from theoretical innovations to practical applications, providing a comprehensive overview of the state-of-the-art.

Ning et al. [?] provided a comprehensive overview of the evolution, challenges, and applications of multi-agent reinforcement learning (MARL)-based intelligent agents, including its practical implementation aspects. Gronauer et al. [?] provided an overview of recent developments in multi-agent deep reinforcement learning, focusing on training schemes, emergent agent behaviors, and the unique challenges of the multi-agent domain, while also discussing future research directions. Yang et al. [?] explored the utility theory application in AI robotics, focusing on how utility AI models can guide decision-making and cooperation in multi-agent/robot systems. Orr et al. [?] reviewed recent advancements in MARL, particularly its applications in multi-robot systems, while discussing current challenges and potential future applications. Du et al. [?] provided a systematic overview of multi-agent deep reinforcement learning for MAS, focusing on its challenges, methodologies, and applications. Pamul et al. [?] provided a comprehensive analysis of the application of MARL in connected and automated vehicles (CAVs), identifying current developments, existing research directions, and challenges. Hernandez-Leal et al. [?] provided a comprehensive overview of approaches to addressing opponent-

induced non-stationarity in multi-agent learning, categorizing algorithms into a new framework and offering insights into their effectiveness across different environments. The survey by Zhu et al. [**?**] provided a systematic classification and analysis of MARL systems that incorporate communication, encompassing recent advanced Comm-MARL research and identifying key dimensions that influence the design and development of these multi-agent systems.

1.3. Motivations of the Current Survey

However, despite the growing body of work in this field, existing surveys often have noticeable limitations [**? ? ? ?**]. Specifically, our thorough investigation reveals that most current reviews and surveys share several common and significant significant drawbacks and limitations:

- **Limited Research Scope:** Previous literature reviews [**? ?**] predominantly remain within the primary framework of reinforcement learning and have not broken through theoretical limitations, resulting in a lack of comprehensive coverage.
- **Neglect of Environments:** Previous literature reviews [**? ? ?**] have largely concentrated on methodological and algorithmic advancements, frequently overlooking the essential role of simulation environments and platforms in multi-agent intelligent decision-making.
- **Under-emphasis of Project Implementation:** Prior surveys [**? ? ?**] often focus on theoretical models and overlook detailed implementation aspects, including code-bases and project architectures. This gap limits readers' ability to fully understand and apply the findings.

To address the aforementioned limitations and challenges, we recognize the need for more systematic and comprehensive reviews in the multi-agent intelligent decision-making field.

Firstly, current reviews overly emphasize deep reinforcement learning and fail to adequately consider other potentially effective intelligent decision-making methods [**? ? ? ?**]. Secondly, with the rapid development of large language models (LLMs), their potential in natural language processing, knowledge representation, and complex decision-making has become increasingly apparent. However, current surveys have largely overlooked their integration. Additionally, existing reviews often neglect the critical role of simulation environments in the development of multi-agent systems. However, simulation environments are not merely auxiliary tools but are an integral part of the MAS development and evaluation process. The agents' learning and decision-making processes are influenced and constrained by these environments, making it equally important to understand and develop these environments as it is to focus on the algorithms themselves. Finally, the lack of attention to practical implementation details in current reviews has resulted in a disconnect between theory and practice. This survey will delve into the specifics of project implementation, including code structures, system architecture, and the challenges encountered during development, to enhance research reproducibility and facilitate the effective translation of theoretical research into practical applications.

Building on the motivations outlined earlier, this survey extends beyond the scope of previous reviews, which were often limited to specific areas of discussion. We treat multi-agent environments as equally important components, alongside the methods and techniques, and provide a thorough introduction to the most advanced algorithms and simulation environments. Moreover, we categorize various multi-agent cooperative decision-making methods from a more fundamental implementation perspective. In summary, this survey seeks to provide a more comprehensive and practical framework for research in multi-agent cooperative decision-making, thereby advancing the continuous development of this critical field.

1.4. The Survey Overview / Contents Organization

As depicted in Figure 2, we have structured the survey to reflect our research approach, with each main and sub-branch corresponding to a specific part: First, in Section 1, we introduce the research background of multi-agent cooperative decision-making, discuss the drawbacks of previous surveys, and outline the organizational structure of this survey. Given that MARL and LLMs-based intelligent decision-making methods demonstrate significant advantages and future potential, our primary attentions are placed on Deep MARL-based and LLMs-based methods due to their superior ability to manage dynamic and uncertain environments. In Section 2, we then delve into mainstream intelligent decision-making approaches, algorithms, and models. We categorize these approaches, with a continued focus on MARL-based and LLMs-based methods, discussing their methodologies, advantages, and limitations. Following this, in Section 3, we provide an in-depth analysis of the leading simulation environments and platforms for multi-agent cooperative decision-making, again focusing on Deep MARL-based and LLMs-based

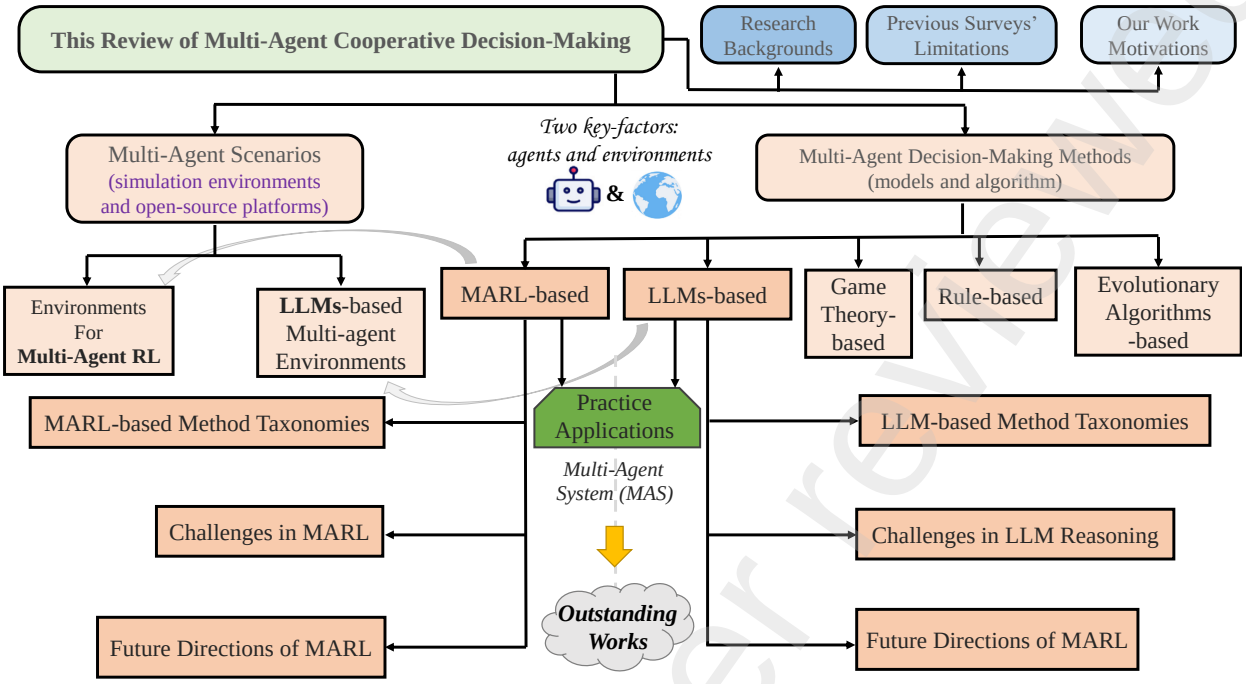


Figure 2: Illustration of our systematic review of multi-agent intelligent decision-making research. Compared to previous reviews, we have incorporated comprehensive introduction and analysis, with each segment corresponding to a specific chapter in the survey.

methods. Furthermore, in Section 4, we discuss the practical applications of multi-agent decision-making systems, such as autonomous driving, UAV navigation, and collaborative robotics. Finally, in Sections 5 and 6, we explore the potential challenges and future research directions of multi-agent cooperative decision-making.

1.5. How to read this survey?

This survey caters to a diverse readership, each with varying levels of expertise and interest in different aspects of multi-agent decision-making systems. To help readers efficiently find the content that interests them, we offer the following guide, providing direction based on different topics:

- For those interested in rule (fuzzy logic)-based, game theory-based, and evolutionary algorithm-based decision-making research, please refer to Section 2.2.1, 2.2.2, and 2.2.3. This section provides a comprehensive analysis of the rule and game-based methods in multi-agent systems, detailing their corresponding technological taxonomies, features, and limitations.
- For those interested in MARL-based decision-making research, please refer to Section 2.3. This section provides a comprehensive analysis of the deep MARL-based methods in multi-agent systems, detailing their corresponding technological taxonomies, advantages, and limitations.
- If you are focused on decision-making based on LLMs, Section 2.4 will offer you an in-depth exploration, with the corresponding technological taxonomies, advantages, and limitations. This part discusses the unique capabilities of LLMs in multi-agent environments and their potential applications, especially in reasoning and decision-making.
- For readers focused on the well-known simulation environments of MAS, we suggest reading Section 3, which primarily covers an introduction to MARL-based Simulation Environments (Section 3.1) and LLMs Reasoning-based Simulation Environments (Section 3.2).
- If your interest lies in the practical applications of multi-agent decision-making systems, Section 4 will be of particular relevance. This section offers a detailed discussion of how these systems are applied across various fields, such as autonomous driving, UAV navigation, and collaborative robotics.

- If you are interested in the challenges and problems faced by existing multi-agent decision-making methods, Section 5 provides an in-depth discussion, exploring the limitations of current approaches and unresolved issues in the field, offering insights into these challenges.
- Finally, if you wish to learn about future research directions and the prospects for multi-agent decision-making technique, we recommend reading Section 6. This section looks ahead to future research trends and potential breakthroughs, exploring key directions that could drive the field forward.

2. Multi-Agent Decision-Making Taxonomies

This section discusses the taxonomies of decision-making in multi-agent systems and their related techniques. The multi-agent cooperative decision-making methods can be broadly classified into five categories: rule-based (primarily fuzzy logic), game theory-based, evolutionary algorithms-based methods, MARL-based approaches, and LLMs-based methods [**?**]. Although these rule-based, game theory-based, and evolutionary algorithms-based solutions demonstrate a degree of effectiveness, they typically rely heavily on pre-designed strategies and assumptions. This dependence limits their adaptability to changing and complex environments and ill-suited for handling highly dynamic and uncertain scenarios. In contrast, DRL-based and LLMs reasoning-based solutions offer more dynamic and flexible approaches, capable of learning and adapting to new strategies on the fly. Therefore, these methods have significant advantages in dealing with dynamic and uncertain environments. Thus, special research attentions are placed on DRL-based and LLMs-based methods due to their significant advantages in handling dynamic and uncertain environments.

The analysis is conducted from multiple perspectives, including agent interaction dynamics, mainstream paradigms of cooperative decision-making, MARL (multi-agent reinforcement learning), and LLM (large language model)-driven multi-agent systems, aiming to provide a systematic framework and technical foundation for the design and optimization of multi-agent decision-making.

Specifically, Section 2.1 analyzes agent interaction dynamics in MAS, categorizing them into four typical types: fully cooperative, fully competitive, mixed cooperative-competitive, and self-interested, while discussing their impact on overall system behavior. Subsequently, Sections 2.2, 2.3, and 2.4 introduce mainstream paradigms of cooperative decision-making, MARL-based decision-making methods, and LLMs-based multi-agent systems, respectively.

2.1. Agent Interaction Dynamics for Multi-Agent Systems

In multi-agent systems, the nature of interaction among agents can be categorized into distinct relational dynamics, i.e., *agent interaction dynamics*, each influencing the overall system behavior and outcomes. These dynamics are critical to understanding and designing intelligent systems where agents operate within shared environments. Below, we detail the primary types of agent relationships:

1. *Fully Cooperative:* In this scenario setting, all agents have aligned objectives, meaning they share identical reward structures and work towards a common goal. The agents operate with complete cooperation, aiming to maximize collective benefits. This relationship is typical in systems where synergy is essential, and the success of one agent directly contributes to the success of others, ensuring mutual reinforcement of strategies and actions.
2. *Fully Competitive:* This relationship is characterized by a zero-sum game dynamic, where the gain of one agent directly translates into the loss of another. Agents are in direct opposition, with their objectives fundamentally misaligned. This is commonly observed in competitive environments such as robotic competitions, where agents are designed to outperform each other, and success is measured relative to the failure or underperformance of others.
3. *Mixed Cooperative and Competitive:* In most real-world scenarios, agents may engage in both cooperation and competition simultaneously. This type of relationship is evident in team-based environments like robotic soccer, where agents within the same team cooperate to achieve a shared objective (e.g., scoring goals), but simultaneously compete against agents from opposing teams. The complexity of such systems lies in balancing internal cooperation with external competition, often requiring sophisticated strategies to optimize outcomes at both individual and collective levels.
4. *Self-Interested:* In self-interested dynamics, each agent acts primarily to maximize its own utility, with little regard for the impact on others. An agent's actions might incidentally benefit or harm other agents, but these effects are not a concern for

the self-interested agent. This relationship is pertinent in scenarios where agents are designed to prioritize personal gain over collective welfare, often leading to outcomes where the overall system efficiency is not necessarily optimized, as individual agents do not account for the potential externalities of their actions on the environment or other agents.

Overall, these agent interaction dynamics is crucial for the design and analysis of multi-agent systems, as they directly impact the strategies employed by agents and the overall system performance. The complexity of agent interactions in mixed or self-interested relationships often requires advanced coordination mechanisms and incentive structures to manage potential conflicts and ensure desired outcomes.

While the overarching concept of agent interaction dynamics holds some value, this survey focuses more on analyzing the characteristics of these methods from a technical and scientific perspective. Therefore, distinguishing relationships is not the primary emphasis of this study. Instead, we will proceed with a more comprehensive taxonomy of Multi-Agent Decision-making Systems.

2.2. Mainstream Paradigms of Multi-Agent Cooperative Decision-Making

In multi-agent cooperative decision-making, several mainstream paradigms exist, each leveraging different techniques to tackle challenges associated with coordination, learning, adaptability, and optimization among autonomous agents. These paradigms utilize diverse approaches, including rule-based (primarily fuzzy logic) systems [**? ? ? ? ? ?**], game theory-based [**? ? ? ? ? ?**], evolutionary algorithms-based [**? ? ? ? ? ?**], MARL-based [**? ? ? ? ? ? ?**], and LLMs-based [**? ? ? ?**] multi-agent decision-making systems. Each of these methods has distinct strengths and applications, depending on the problem context and the complexity of interactions between agents. For a comprehensive overview, please refer to Table 1, which provides a detailed classification of these paradigms.

*2.2.1. **Rule-Based (Primarily Fuzzy Logic)***

Rule-based decision-making, particularly fuzzy logic, has been widely adopted in multi-agent systems (MAS) due to its ability to handle uncertainty, imprecise data, and dynamic environments [**? ? ? ?**]. Fuzzy logic enables agents to make adaptive, human-like decisions by mapping inputs to linguistic rules rather than strict mathematical models.

Miki et al. [**?**] presented a rule-based multi-agent control algorithm that utilizes local information instead of absolute coordinates, making it more practical for real-world applications. Charaf et al. [**?**] introduced a rule-based multi-agent system to address coordination challenges, such as controllability and observability, in distributed testing environments. Yarahmadi et al. [**?**] reviewed the applications of multi-agent systems in Cyber-Physical Systems (CPS) and the Internet of Things (IoT), proposing a combination of learning and rule-based reasoning to improve decision-making in MAS. Marti et al. [**?**] presented an expert rule-based system using multi-agent technology to support traffic management during weather-related issues. Daeichian et al. [**?**] used fuzzy logic in combination with Q-learning and game theory to control traffic lights autonomously. Wu et al. [**?**] introduced a fuzzy-theoretic game framework that integrates fuzzy logic with game theory to handle uncertainty in utility values during multi-agent decision making. Nekhai et al. [**?**] devised a cybersecurity management model for agricultural enterprises using a multi-agent system (MAS) based on fuzzy logical reasoning. Ramezani et al. [**?**] applied fuzzy logic to multi-agent decision-making in soccer robot teams, combining cooperative and non-cooperative game strategies. Zhang et al. [**?**] introduced a new online method for optimal coordination control in multi-agent differential games, combining fuzzy logic, and adaptive dynamic programming. Ren et al. [**?**] presented a fuzzy logic-based approach for partner selection in multi-agent systems, emphasizing flexibility and adaptability in dynamic environments. Gu et al. [**?**] introduced a cooperative reinforcement learning algorithm for multi-agent systems using a leader-follower framework, modeled as a Stackelberg game. Schwartz et al. [**?**] introduced a multi-agent fuzzy actor-critic learning algorithm for differential games. Harmati et al. [**?**] proposed a game-theoretic model for coordinating multiple robots in target tracking, using a semi-cooperative Stackelberg equilibrium and a fuzzy inference system for high-level cost tuning. Khuen et al. [**?**] introduced an Adaptive Fuzzy Logic (AFL) approach for multi-agent systems with negotiation capabilities, focusing on resource allocation. Yan et al. [**?**] proposed a graphical game-based adaptive fuzzy optimal bipartite containment control scheme for high-order nonlinear multi-agent systems (MASs). Vicerra et al. [**?**] proposed a multi-agent robot system using a pure fuzzy logic-based artificial intelligence model. Gu et al. [**?**] presented a fuzzy logic-based policy gradient multi-agent reinforcement learning algorithm for leader-follower systems, where fuzzy logic

Table 1: Representative Methods in Mainstream Paradigms of Multi-Agent Cooperative Decision-Making.

Paradigm	Representative Methods and Key References
Rule-Based (Primarily Fuzzy Logic)	Miki et al. [**?**], Yarahmadi et al. [**?**], Wu et al. [**?**], Zhang et al. [**?**], Ren et al. [**?**], Gu et al. [**?**], Schwartz et al. [**?**], Harmati et al. [**?**], Khuen et al. [**?**], Yan et al. [**?**], Vicerra et al. [**?**], Gu et al. [**?**], Maruyama et al. [**?**], Peng et al. [**?**], Yang et al. [**?**]
Game Theory-based	Wang et al. [**?**], Guo et al. [**?**], Schwung et al. [**?**], Wang et al. [**?**], Lin et al. [**?**], Wang et al. [**?**], Wang et al. [**?**], Lanctot et al. [**?**], Guo et al. [**?**], Zhang et al. [**?**], Kong et al. [**?**], Wang et al. [**?**], Dong et al. [**?**], Nguyen et al. [**?**], Schwung et al. [**?**], Khan et al. [**?**]
Evolutionary Algorithms-based	Liu et al. [**?**], Xu et al. [**?**], Daan et al. [**?**], Franciszek et al. [**?**], Larry et al. [**?**], Daan et al. [**?**], Liu et al. [**?**], Yuan et al. [**?**], Dong et al. [**?**], Chen et al. [**?**], Zhang et al. [**?**]
MARL-based	Wai et al. [**?**], Hu et al. [**?**], Son et al. [**?**], Yu et al. [**?**], Rashid et al. [**?**], Rashid et al. [**?**], Sunehag et al. [**?**], Huang et al. [**?**], Xu et al. [**?**], Yun et al. [**?**], Mao et al. [**?**], Kraemer et al. [**?**], Kouzeghar et al. [**?**], Gao et al. [**?**], Liu et al. [**?**], Qi et al. [**?**], Vinyals et al. [**?**], Lu et al. [**?**], Chu et al. [**?**], et al. [**?**], Kurach et al. [**?**], Lv et al. [**?**], Radac et al. [**?**], Wang et al. [**?**], Liu et al. [**?**]
LLMs-based	Mordatch et al. [**?**], Zhang et al. [**?**], Xu et al. [**?**], Li et al. [**?**], Wang et al. [**?**], Zhao et al. [**?**], Hou et al. [**?**], Puig et al. [**?** **?**], Gao et al. [**?**], Xiao et al. [**?**], Wang et al. [**?**], Wu et al. [**?**], Wen et al. [**?**], Chen et al. [**?**], Liu et al. [**?**], Chen et al. [**?**], Hong et al. [**?** **?**], XAgent Team [**?**], Wang et al. [**?** **?**], Zheng et al. [**?**], Zhang et al. [**?** **?**], Cao et al. [**?**]

controllers act as policies. Maruyama et al. [**?**] extended the classical framework for reasoning about distributed knowledge, incorporating fuzzy logic to handle uncertainty and degrees of certainty within multi-agent systems. Peng et al. [**?**] proposed a two-layer coordination model for multi-agent systems using fuzzy reinforcement learning. Yang et al. [**?**] presented a multi-agent reinforcement learning algorithm with fuzzy policy to address control challenges in cooperative multi-agent systems, particularly for autonomous robotic formations.

Overall, fuzzy logic remains a foundational approach for rule-based decision-making in MAS, offering interpretability and robustness in uncertain environments. In the future, fuzzy logic will be further integrated with LLMs, hierarchical decision architectures, and multi-agent planning, enabling more precise and adaptive decision-making in complex real-world scenarios.

*2.2.2. **Game theory-based***

Game theory provides a structured framework for analyzing strategic interactions in multi-agent systems. It enables agents to make rational decisions in cooperative, competitive, or mixed scenarios through equilibrium-based optimization [**?** **?**]. Traditional methods such as Nash equilibrium and Stackelberg games form the foundation, while modern approaches integrate reinforcement learning and Bayesian inference to enhance adaptability in dynamic environments.

Wang et al. [**?**] provided a broad discussion on game-theoretic approaches in multi-agent systems, covering cooperative and non-cooperative scenarios. Guo et al. [**?**] applied game theory to multi-agent path planning, leveraging Nash equilibrium to optimize navigation and obstacle avoidance. Zhang et al. [**?**] developed a distributed control algorithm that ensures optimal coverage while maintaining network connectivity.

Beyond fundamental decision-making, game theory has been applied in communication networks and energy systems. Wang et al. [**?**] utilized game-theoretic learning to enhance resource allocation in wireless networks while countering adversarial actions like jamming. Lin et al. [**?**] introduced potential game theory to optimize distributed energy management in microgrids, where agents autonomously coordinate power distribution. Dong et al. [**?**] further extended this approach using a hierarchical Stackelberg model for energy trading, balancing incentives between microgrids and individual agents.

Incorporating machine learning with game theory has also led to advances in multi-agent optimization. Schwung et al. [**?**] combined potential game theory with reinforcement learning for adaptive production scheduling, while Wang et al. [**?**] designed a

Nash equilibrium-based fault-tolerant control strategy for multi-agent systems. Additionally, game-theoretic methods have been explored for distributed computing, as shown by Khan et al. [**?**], who developed a replica placement strategy to minimize data access delays in distributed systems.

Overall, game theory remains a cornerstone of multi-agent decision-making, offering well-defined theoretical guarantees while enabling dynamic adaptation through hybrid approaches. Future research will likely focus on integrating game theory with deep learning and large language models to enhance strategic reasoning in high-dimensional, uncertain environments.

2.2.3. ***Evolutionary Algorithms-based***

Evolutionary algorithms (EAs) provide a bio-inspired approach to optimization in multi-agent systems by leveraging principles such as natural selection, mutation, and recombination [**?** **?** **?**]. By allowing agents to evolve their strategies iteratively, EAs are particularly effective for problems requiring continuous learning, large-scale coordination, and self-organized behavior.

Liu et al. [**?**] introduced the Multi-Agent Genetic Algorithm (MAGA), where agents interact through competition and cooperation to optimize global solutions. Xu et al. [**?**] extended this idea to hardware-based multi-agent systems, using nanoclusters as physical agents to achieve large-scale parallel computation. Daan et al. [**?**] explored the role of evolutionary strategies in dynamic environments such as financial markets, smart grids, and robotics, demonstrating how adaptive algorithms can handle real-world uncertainties.

Franciszek et al. [**?**] proposed a self-optimization model integrating cellular automata and game theory, simulating competitive evolutionary interactions among agents. Larry et al. [**?**] analyzed the trade-offs between mutation and recombination, showing that mutation can sometimes outperform traditional recombination strategies in evolutionary computing. To further enhance adaptability, Daan et al. [**?**] introduced Deep Neuroevolution (DNE), applying coevolutionary techniques to complex multi-agent scenarios, including Atari games.

Recent studies have focused on scaling evolutionary learning to larger agent populations. Liu et al. [**?**] developed Evolutionary Reinforcement Learning (ERL), a scalable approach that partitions learning into multiple stages, ensuring better adaptability in multi-agent environments. Yuan et al. [**?**] introduced EvoAgent, a framework that extends LLMs-based autonomous agents into multi-agent systems using evolutionary techniques like mutation and selection.

Evolutionary game theory has also been explored to improve cooperative behavior. Dong et al. [**?**] designed a three-strategy decision model, where agents adopt conservative or adaptive strategies based on their interactions with neighbors, fostering long-term cooperation. Chen et al. [**?**] proposed a kinetic decision-making model grounded in rarefied gas dynamics, offering a new perspective on agent evolution using the Boltzmann equation. Zhang et al. [**?**] applied evolutionary game theory to policy optimization, analyzing cooperation strategies among governments, enterprises, and farmers in agricultural water conservation projects.

Overall, evolutionary algorithms provide a robust framework for decentralized decision-making, allowing agents to self-improve and adapt in uncertain environments. In the future, evolutionary algorithms will be further integrated with deep learning, hierarchical evolution, and large-scale multi-agent coordination, enabling more adaptive, autonomous MAS.

2.2.4. ***MARL-based Multi-Agent Systems***

Before introducing the MARL-based multi-agent systems (MAS), we provide a detailed discussion in Appendix A on the key technological comparisons and methodological principles of both DRL-based single-agent systems and MARL-based MAS. This helps readers build the necessary background knowledge for better understanding the following discussions.

Multi-Agent Reinforcement Learning offers a structured framework to tackle decision-making in MAS, where autonomous agents interact with each other and a shared environment. The MAS research in MARL is broadly divided into three paradigms: *Centralized Training with Centralized Execution (CTCE)* [**?** **?**], *Decentralized Training with Decentralized Execution (DTDE)* [**?**], and *Centralized Training with Decentralized Execution (CTDE)* [**?** **?**]. Each paradigm is designed to address specific challenges such as coordination, scalability, and policy optimization, providing tailored solutions for diverse scenarios.

Centralized Training with Centralized Execution The CTCE paradigm [**?** **?**] relies on a central controller that governs all agents by aggregating their observations, actions, and rewards to make joint decisions. While this paradigm enables high levels of coordination, its scalability is limited in large-scale systems. Multi-Agent DQN (MADQN) [**?** **?** **?**] is a representative method, employing parameter-sharing mechanisms to handle cooperative tasks effectively. However, its reliance on centralized control restricts its applicability in

dynamic environments with numerous agents.

Decentralized Training with Decentralized Execution The DTDE paradigm [**?**] emphasizes independent learning and execution, where each agent interacts with the environment individually and updates its policy based solely on local observations and rewards. This paradigm excels in scalability and robustness, especially in scenarios with limited communication. Notable methods include *Independent Q-Learning (IQL)* [**?** **?**] and *Decentralized REINFORCE* [**?**], which allow agents to learn autonomously. Despite its advantages, DTDE faces challenges such as learning non-stationarity, where the environment changes as other agents adapt, and difficulty in addressing the credit assignment problem in cooperative settings.

Centralized Training with Decentralized Execution The CTDE paradigm [**?** **?** **?**] combines the strengths of centralized training and decentralized execution, making it the most prominent paradigm in MARL research. During training, a central controller aggregates information from all agents to optimize their policies, but during execution, each agent operates independently based on its own observations. CTDE addresses key challenges like non-stationarity and scalability, with methods such as *Value Decomposition Networks (VDN)* [**?**] and *QMIX* [**?** **?**] for value-based learning, *Multi-Agent Deep Deterministic Policy Gradient (MADDPG)* [**?**] for actor-critic frameworks, and *Multi-Agent Proximal Policy Optimization (MAPPO)* [**?**] for policy gradient optimization. These approaches are widely applied in complex environments like StarCraft II [**?** **?**] and the Multi-Agent Particle Environment (MPE) [**?** **?**].

Communication-based MARL Algorithms Additionally, communication-based MARL algorithms have emerged to enhance coordination by enabling agents to share critical information during training and execution. Examples include *Attentional Communication (ATOC)* [**?**] and *Targeted Multi-Agent Communication (TarMAC)* [**?**], which use advanced mechanisms to improve the efficiency and effectiveness of inter-agent communication in cooperative tasks.

By structuring MARL methods within these paradigms, researchers provide a clear framework for addressing the diverse challenges of multi-agent decision-making. From autonomous driving fleets to resource allocation systems, MARL continues to push the boundaries of what distributed intelligent systems can achieve [**?**].

*2.2.5. **LLMs-based Multi-Agent Systems***

Although LLMs like GPT [**?** **?** **?**], Llama [**?** **?**], and Gemini [**?**] support very long input contexts, their ability to understand complex inputs still varies. In this context, multi-agent collaboration optimizes task execution through role assignment, enabling better performance through collaboration among multiple agents compared to a single agent. Each agent has an independent workflow, memory, and can seek help from other agents when necessary. LLMs-based Multi-Agent Systems represent a relatively new multi-agent decision-making model that leverages the powerful capabilities of language models, to enhance communication and collaboration between autonomous agents. In an LLMs-based multi-agent system, agents communicate via natural language or symbolic representations, breaking down complex tasks into smaller, more manageable subtasks. One important feature of LLMs-based systems is the hierarchical organization of agents, typically consisting of two levels [**?** **?**]:

1) Global planning agents, responsible for high-level decisions such as task decomposition, resource allocation, and overall strategy management.

2) Local execution agents, which are responsible for executing specific subtasks and providing feedback to the global planning agent. These agents are generally more focused on local tasks but communicate progress and challenges with the global level for adjustments. This decomposition makes distributed problem solving possible, with agents sharing information, strategies, and goals through language, thus advancing task execution together.

For example, frameworks like AutoGen [**?** **?** **?**], Crew AI [**?** **?**], and LangGraph [**?** **?**] provide rich tools for building multi-agent solutions, supporting efficient cooperation and interaction between agents. Through these frameworks, developers can build virtual teams that leverage the strengths of different agents in distributed tasks. Additionally, LLMs-based multi-agent systems possess adaptive re-planning capabilities, enabling them to adjust in dynamic environments. When agents encounter changes or new information, they can quickly update strategies or re-plan tasks using language models, ensuring the system remains aligned with changing goals.

Firstly, LLMs-based multi-agent environments have emerged as pivotal platforms for advancing research in multi-agent collaboration, reasoning, and task execution. For instance, environments such as ThreeDWorld Multi-Agent Transport (TDW-MAT) [**?** **?**], Communicative Watch-And-Help (C-WAH) [**?**], Cuisineworld

[?], and AgentScope [?] offer diverse settings for evaluating and enhancing multi-agent systems in various contexts, from household chores to gaming interactions and beyond. For instance, MindAgent [?] is a novel infrastructure for evaluating planning and coordination capabilities in gaming interactions, leveraging large foundation models (LFMs) to coordinate multi-agent systems, collaborate with human players. Communicative Watch-And-Help (C-WAH) [? ?] is a realistic multi-agent simulation environment and an extension of the Watch-And-Help Challenge platform, VirtualHome-Social. AgentScope [?] is a robust and flexible multi-agent platform designed to empower developers in building advanced multi-agent systems by leveraging the potential of LLMs.

Meanwhile, LLMs-based multi-agent systems have broad applications and great prospects [? ?]. They can collaborate in robotic teams to perform complex tasks, such as product assembly or joint research, ensuring seamless interaction and cooperation [? ? ?]. In autonomous driving, LLMs help vehicles communicate, sharing real-time data and navigation strategies to achieve coordinated actions. Moreover, LLMs can support agents (such as drones) in disaster response, transmitting critical information to help systems efficiently respond to crises. *Wu et al.* [?] proposed AutoGen, an open-source framework for developing next-generation LLM applications through multi-agent conversations, allowing customizable agent interactions and integration of LLMs, human inputs, and tools. *Xiao et al.* [?] introduced Chain-of-Experts (CoE), a multi-agent framework that enhances reasoning in complex operations research (OR) problems using LLMs, with domain-specific roles and a conductor for guidance. *Chen et al.* [?] presented AgentVerse, a multi-agent framework inspired by human group dynamics, dynamically adjusting agent roles and composition to enhance complex task-solving across various domains. *Chen et al.* [?] developed AutoAgents, a framework that adaptively generates and coordinates multiple specialized agents for efficient task completion. *Liu et al.* [?] proposed Dynamic LLM-Agent Network (DyLAN), a framework that enhances LLM-agent collaboration through dynamic interactions based on task requirements. *Zhang et al.* [?] introduced CoELA, a Cooperative Embodied Language Agent framework that leverages LLMs to enhance multi-agent cooperation in complex, decentralized environments. *Hong et al.* [? ?] proposed MetaGPT, a meta-programming framework that enhances LLMs-based multi-agent system collaboration using Standard Operating Procedures (SOPs). *XAgent Team* [?] developed XAgent, an open-source, LLM-driven autonomous agent framework for solving complex tasks using a dual-loop architecture for task planning and execution. *Zheng et al.* [?] introduced PlanAgent, a closed-loop motion planning framework for autonomous driving using multi-modal LLMs to generate hierarchical driving commands. *Wang et al.* [? ?] developed LangGraph, a library for building stateful, multi-actor applications with LLMs, offering fine-grained control over workflows and state management. *Zhang et al.* [? ?] introduced CrewAI, an open-source framework for coordinating AI agents in role-playing and autonomous operations, with a modular design for efficient collaboration. *Hou et al.* [?] proposed CoAct , a hierarchical multi-agent system leveraging LLMs for collaborative task execution. It features a global planning agent for task decomposition and strategy management, and a local execution agent for subtask implementation, feedback collection, and adaptive replanning, ensuring alignment with overarching goals.

Despite the strong capabilities of LLMs in small to medium-sized multi-agent systems, scalability remains an open issue, particularly in maintaining coherent communication between large numbers of agents in large environments. As the number of agents increases, the complexity of coordinating their behaviors through language also intensifies. Finding a balance between agent autonomy and effective collaboration is a significant challenge. Additionally, LLMs are often seen as black-box models, meaning understanding the reasoning process behind an agent's decision-making can be difficult. The lack of transparency poses challenges for trust and debugging.

In summary, LLMs-based multi-agent systems hold great potential in a variety of applications, offering an advanced way to model and solve complex decision-making problems that require high levels of coordination, adaptability, and communication between agents. By optimizing task decomposition, collaboration, and feedback mechanisms, LLMs bring unprecedented efficiency and flexibility to multi-agent systems.

2.3. MARL-based Multi-Agent Decision-Making Taxonomies

In multi-agent systems, where multiple autonomous agents interact with a shared environment and often with each other, the complexity of decision-making increases significantly. To achieve optimal performance, agents need to learn not only how to act individually but also how to coordinate with others. One of the central challenges in MARL-based multi-agent systems is determining how much information should be shared among

agents during different phases of learning and deployment. The MARL research is typically structured into three primary paradigms: CTCE [**?** **?** **?** **?** **?**], DTDE [**?** **?**], and CTDE [**?** **?** **?**]. As illustrated in Figure 3, we will next explain the principles and differences of the three methods in conjunction with this conceptual framework diagram.

2.3.1. Centralized Training with Decentralized Execution (CTDE)

As shown in the left of Figure 3, CTDE is a hybrid MARL approach that combines the strengths of both centralized and decentralized systems [**?**]. In CTDE, each agent possesses its own policy network, which is trained under the guidance of a central controller. This approach is characterized by a two-phase process: *centralized training* followed by *decentralized execution*.

1. *Centralized Training (Phase 1):* During the training phase, the central controller collects data from all agents, including their observations, actions, and rewards. This centralized data aggregation allows the controller to oversee the learning process and facilitate the training of each agent's policy network.
2. *Decentralized Execution (Phase 2):* Once the training is complete, the central controller's involvement ceases, and each agent operates independently. At execution, agents make decisions based on their own observations using their trained policy networks.

In some communication-constrained scenarios, agents often cannot share or fully share their observations of the environment. Instead, they must make decisions independently based on their own local observations and policies, which limits the applicability of fully centralized methods. To overcome this challenge, Kraemer et al. [**?** **?**] proposed the CTDE learning paradigm. The CTDE agents have access to global environmental state information and the observations of other agents during the training phase, allowing them to learn a joint policy together. However, during the execution phase, each agent relies solely on its own observations and the trained policy to make independent decisions. It combines the advantages of fully decentralized and fully centralized methods, effectively mitigating issues such as learning non-stationarity and the curse of dimensionality, making it the dominant paradigm in current MARL solutions.

Under CTDE, MARL algorithms can primarily be categorized into three types based on their technical implementations: value function decomposition-based algorithms, actor-critic-based algorithms, and algorithms based on policy gradient methods, such as proximal policy optimization (PPO).

1. Value Decomposition-based Algorithms Value decomposition-based algorithms mainly address the challenge of estimating the joint state-action value function (Q-value) in multi-agent systems, which is difficult due to the high dimensionality of the joint action space. Instead of directly estimating this joint value function, these algorithms decompose it into more manageable individual state-action value functions (Q-value) for each agent. During execution, each agent selects its action based on its own value function. In training, the joint value function is computed from individual value functions, and the temporal difference error of the joint value guides the learning of the individual functions. A key principle these algorithms must satisfy is the Individual-Global-Max (IGM) principle, ensuring that the actions maximizing the joint value are consistent with those maximizing individual values. Different algorithms use various methods to approximate or satisfy the IGM principle.

Value Decomposition Networks (VDN) [**?**] is one of the earliest value decomposition-based algorithms in CTDE-based MARL models. VDN simplifies the estimation of the joint state-action value function by assuming that it can be represented as the sum of the individual state-action value functions of all agents. It means that the joint value function is obtained by simply adding up the individual value functions, which does not take into account the varying contributions of each agent's Q-value. However, the assumption made by VDN is a sufficient but not necessary condition for satisfying the IGM principle, which can limit its applicability. Additionally, VDN does not utilize global state information during training, further restricting its effectiveness in more complex environments.

To address this issue, Rashid et al. [**?**] proposed the QMIX algorithm within the CTDE paradigm. QMIX assumes a monotonic nonlinear relationship between the joint state-action value function and the individual state-action value functions of agents. To implement this, QMIX introduces a mixing network that computes the joint state-action value function based on the individual Q-values of all agents. This mixing network is designed with non-negative parameters to ensure that the monotonicity assumption is met. QMIX has been successfully applied in various scenarios and is considered one of the most successful value decomposition algorithms to date. By enforcing a monotonic relationship between the joint action Q-values and individual Q-

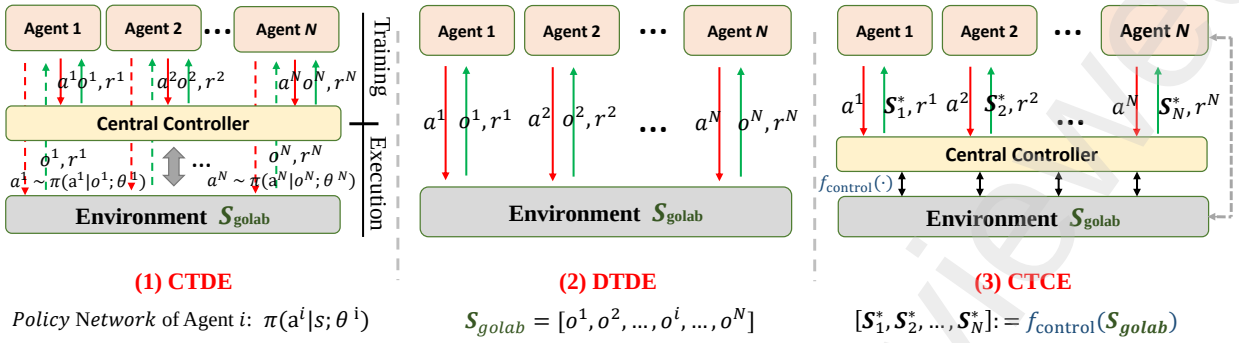


Figure 3: The paradigms visualization of CTDE (left), DTDE (centre), and CTCE (right), consisting of three crucial elements: agent (i.e., algorithm or model), environment, central controller (Optional).

values, QMIX simplifies the policy decomposition process, facilitating decentralized decision-making. However, the monotonicity assumption, while sufficient for ensuring the Individual-Global-Max (IGM) principle, is not a necessary condition. This limitation restricts the algorithm's applicability in situations where an agent's optimal action depends on the actions of other agents.

Weighted QMIX [**?**] builds upon QMIX and addresses this limitation by introducing a novel weighting mechanism during the projection of Q-values, which is widely used for cooperative MARL scenarios. In QMIX, the projection of Q-learning targets into the representable space is done with equal weighting across all joint actions, which can lead to suboptimal policy representations, even if the true optimal Q-values (Q*) are known. To overcome this, Weighted QMIX introduces two weighting schemes-Centrally-Weighted (CW) QMIX and Optimistically-Weighted (OW) QMIX-that place greater emphasis on the better joint actions during this projection process. The weighting schemes ensure that the correct maximal action is recovered for any set of joint action Q-values, effectively improving the algorithm's ability to learn optimal policies. These schemes in Weighted QMIX enhances the representational capacity of QMIX, demonstrating improved results on both predator-prey scenarios of Multi-Agent Particle Environment (MPE) [**?**] and the challenging StarCraft II benchmarks [**? ? ?**].

Since then, numerous methods building on value function decomposition have been developed. QPLEX [**?**] introduces a novel duplex dueling network architecture for multi-agent Q-learning, designed to nonlinearly decompose the joint state-action value function while embedding the IGM principle directly into the network structure. Fast-QMIX [**?**] enhances the original QMIX by dynamically assigning virtual weighted Q-values with an additional network, improving convergence speed, stability, and overall performance in cooperative multi-agent scenarios. QTRAN [**?**] introduces a more flexible factorization method that overcomes the structural limitations of QMIX, where the joint Q-value is constrained to be a monotonic function of the individual Q-values, thereby imposing a specific structural form on the factorization. Specifically, QTRAN introduces a necessary and sufficient condition for the IGM principle and incorporates two additional loss terms into the loss function to constrain the training of individual Q-value functions, ensuring they satisfy this IGM principle.

2. Actor-Critic-based Algorithms: Actor-Critic-based algorithms [**? ? ?**] represent a foundational class of methods within the CTDE paradigm, offering a flexible and effective approach for tackling the challenges of multi-agent environments. These algorithms combine the strengths of policy optimization (actor) with value estimation (critic), allowing agents to learn robust and adaptive strategies in both cooperative and competitive settings. By leveraging a centralized critic during training, Actor-Critic-based methods [**? ? ?**] address key issues such as environmental non-stationarity and credit assignment, enabling effective policy optimization in dynamic and complex multi-agent scenarios. Below, we discuss several prominent Actor-Critic-based approaches and their contributions to advancing MARL.

MADDPG [**?**] is a typical Actor-Critic-based CTDE approach specifically designed to address the challenges of multi-agent environments, where agents engage in both cooperative and competitive interactions. Traditional reinforcement learning algorithms, such as Q-learning and policy gradient methods, struggle in multi-agent settings due to issues like non-stationarity-where the environment constantly changes as other agents

learn-and increased variance with the growing number of agents. MADDPG adapts the actor-critic framework by incorporating a centralized critic during training, which has access to the actions and observations of all agents. This centralized critic helps mitigate the non-stationarity problem by learning a more stable value function that considers the joint action space. During execution, however, each agent independently follows its policy (actor) based on local observations, enabling decentralized decision-making. It allows each agent to successfully learn and execute complex coordination strategies, outperforming existing methods in both cooperative and competitive multi-agent environments. To address the computational challenges of continuous action spaces, Li et al. [**?**] extend the MADDPG algorithm to Multi-Agent Mutual Information Maximization Deep Deterministic Policy Gradient (M3DDPG) by incorporating a minimax approach to enhance robustness in multi-agent environments. M3DDPG introduce Multi-Agent Adversarial Learning (MAAL), which efficiently solves the minimax formulation, ensuring agents can generalize even when opponents' policies change and leading to significant improvements over existing baselines in mixed cooperative-competitive scenarios.

Counterfactual Multi-Agent Policy Gradient (COMA) [**?**] is a cooperative algorithm based on the Actor-Critic framework that uses centralized learning to address the credit assignment problem in multi-agent settings. COMA employs a centralized critic to compute advantage functions for each agent, using counterfactual baselines to reduce policy dependencies among agents and improve learning efficiency. Each agent has its own policy network, but the shared centralized critic evaluates joint Q-values by considering the collective state and action information of all agents. This approach minimizes the negative impacts of policy dependencies and allows for a more comprehensive assessment of each agent's behavior, enhancing overall policy optimization.

3. Proximal Policy Optimization-based Algorithms: Proximal Policy Optimization (PPO) [**?**] is a widely used CTDE reinforcement learning algorithm that has been adapted and extended to address challenges in MARL. Within the CTDE paradigm, PPO and its multi-agent variants have shown remarkable effectiveness in balancing policy optimization efficiency and stability. PPO was introduced by Schulman et al. [**?**] as an efficient policy gradient algorithm designed to improve upon the trust region policy optimization (TRPO) framework [**?**]. PPO employs a clipped surrogate objective function that simplifies the trust region constraint in TRPO, allowing for stable updates without overly restrictive computational overhead. The key innovation of PPO lies in its ability to control the magnitude of policy updates through the clipping mechanism, which ensures that policies do not deviate excessively from their previous versions.

In MARL, Multi-Agent PPO (MAPPO) [**?**] extends PPO to the centralized critic paradigm. MAPPO uses a centralized value function (critic) that evaluates joint states and actions during training, while agents execute independently using their decentralized policies. MAPPO has demonstrated superior performance in various cooperative and competitive multi-agent environments, such as the StarCraft II [**? ?**] and Multi-Agent Particle Environment (MPE) [**? ? ?**] benchmarks. The centralized critic allows for improved credit assignment and non-stationarity handling during training, while the decentralized execution ensures scalability. While MAPPO leverages parameter sharing among agents, this assumption may not hold in heterogeneous-agent systems where agents differ in capabilities, objectives, or action spaces.

To address this, Kuba et al. [**?**] proposed Heterogeneous-Agent Trust Region Policy Optimization (HATRPO) and Heterogeneous-Agent Proximal Policy Optimization (HAPPO). These algorithms remove the parameter-sharing assumption, allowing for individualized policy networks for each agent. HATRPO builds upon TRPO by introducing a sequential update scheme, where only one agent updates its policy at a time while the policies of other agents remain fixed. This approach ensures monotonic improvement in joint policies, as it approximates the Nash equilibrium under certain conditions, such as full observability and deterministic environments. HAPPO extends PPO in a similar vein, replacing parameter sharing with individualized policies. Like HATRPO, HAPPO employs a sequential update mechanism, but it retains the computational efficiency and practical simplicity of PPO's clipped objective function.

Both HATRPO and HAPPO utilize a sequential update process where one agent updates its policy while others remain fixed. This prevents conflicts during policy optimization and ensures theoretical convergence to a stable joint policy. Moreover, HATRPO and HAPPO provide monotonic improvement guarantees under specific conditions. By removing the parameter-sharing constraint, these algorithms enable agents to learn tailored policies that account for their unique roles and capabilities. Both algorithms perform competitively in benchmark tasks, demonstrating their ability to scale to high-dimensional state-action spaces while maintaining robust coordination among agents.

PPO-based algorithms, including MAPPO [**?**], HATRPO [**?**], and HAPPO [**?**], have revolutionized multi-agent reinforcement learning by combining the stability of PPO with the coordination benefits of centralized critics. These algorithms have proven effective across a wide array of cooperative and competitive MARL tasks, offering strong performance and scalability.

3. Other Categories of Algorithms within the CTDE Paradigm: In addition to the well-established categories of Value Decomposition-based, Actor-Critic-based, and Proximal Policy Optimization (PPO)-based algorithms, the MARL research has seen significant advancements through innovative optimizations and enhancements within CTDE paradigm that are not confined to these traditional classifications. These approaches aim to address the inherent challenges of multi-agent environments, such as non-stationarity and limited communication, to improve overall cooperation and policy learning efficiency.

For example, Centralized Advising and Decentralized Pruning (CADP) is a novel framework introduced by Zhou et al. [**?**] to address limitations in the CTDE paradigm. CADP enhances the training process by allowing agents to explicitly communicate and exchange advice during centralized training, thus improving joint-policy exploration. To maintain decentralized execution, CADP incorporates a smooth model pruning mechanism that gradually restricts agent communication without compromising their cooperative capabilities, demonstrating its superior performance on multi-agent StarCraft II SMAC and Google Research Football benchmarks. CommNet [**?**] introduces a neural model where multiple agents learn to communicate continuously and collaboratively through a shared communication channel, optimizing their performance on fully cooperative tasks. The method allows agents to develop their own communication protocols during training, leading to improved coordination and task-solving capabilities. Mao et al. [**?**] introduced a novel Meta-MARL framework by integrating game-theoretical meta-learning with MARL algorithms using the CTDE's framework, such as the Actor-Critic-based COMA [**?**]. This framework offers initialization-dependent convergence guarantees and significantly improves convergence rates by addressing related tasks collectively. Yun et al. [**?**] proposed a novel approach called Quantum Meta Multi-Agent Reinforcement Learning (QM2ARL), achieving high rewards, fast convergence, and quick adaptation in dynamic environments. QM2ARL leverages the unique dual-dimensional trainability of Quantum Neural Networks (QNNs) to enhance MARL. Liu et al. [**?**] proposed the Learning before Interaction (LBI) framework, which integrates a language-guided simulator into the multi-agent reinforcement learning pipeline to address complex decision-making problems. By leveraging a generative world model with dynamics and reward components, LBI generates trial-and-error experiences to improve policy learning, demonstrating superior performance and generalization on the StarCraft Multi-Agent Challenge benchmark [**? ?**].

2.3.2. Decentralized Training with Decentralized Execution (DTDE)

As shown in the centre of Figure 3, DTDE represents a fully decentralized mechanism where each agent interacts independently with the environment and updates its own policy based on its own observations and rewards [**?**]. In this framework, each agent trains and operates completely independently, relying only on its own observations and rewards to update its strategy. DTDE is particularly suited for environments with limited communication or no global coordination, offering strong scalability and robustness [**?**].

The core idea behind DTDE is the independence of agents [**?**]. Each agent interacts with its environment and learns without requiring information from others. This makes DTDE scalable, but it also introduces challenges such as non-stationarity, where the environment appears to change as other agents adapt their strategies. This characteristic makes DTDE a valuable and challenging area of research. The theoretical foundation of DTDE is often based on Decentralized Partially Observable Markov Decision Processes (Dec-POMDPs). As described by Amato et al. [**? ?**], a Dec-POMDP models a decentralized decision-making environment where agents operate independently with limited observations while aiming to maximize a shared reward. The decentralized nature of DTDE requires each agent to learn optimal strategies based on local information only.

First and foremost, one of the earliest DTDE approaches is Independent Q-Learning (IQL) by et al. [**?**]. Here, each agent applies Q-learning independently, maintaining its own Q-function and updating it based on local observations and rewards. However, IQL faces several challenges, such as the non-stationary nature of the environment caused by other agents learning simultaneously. It also struggles with credit assignment, where it is hard to determine how an individual agent contributes to the team's success. To address these issues, several extensions of IQL have been proposed:

- **Distributed Q-Learning** [**?**] optimistically assumes other agents always take optimal actions, fo-

cusing on learning from high-reward interactions. While effective in deterministic settings, it can be overly optimistic in environments with randomness.

- **Hysteretic Q-Learning** [**?**] By introducing two learning rates—one for positive updates and another, smaller rate for negative updates—hysteretic Q-learning balances optimism with robustness in stochastic environments.

- **Lenient Q-Learning** [**?**] dynamically adjusts how lenient the agent is in updating its values, depending on how frequently specific state-action pairs are encountered. It allows for more exploration in the early stages of learning while focusing on optimization later.

As MARL problems become more complex, DTDE methods have been extended to deep learning. Deep Q-Networks [**? ? ? ?**] and Deep Recurrent Q-Networks [**?**] have been adapted for decentralized settings, enabling agents to handle high-dimensional state and action spaces. Independent DRQN (IDRQN) [**?**], for example, combines DRQN with independent learning, but its asynchronous experience replay can cause instability. To solve this, Concurrent Experience Replay Trajectories (CERTs) [**?**] synchronize experience replay among agents, reducing non-stationarity and improving learning efficiency. Other advancements include Deep Hysteretic DRQN (Dec-HDRQN) [**?**], which combines hysteretic updates with deep neural networks and uses concurrent buffers to handle decentralized data. These methods have shown robust performance in partially observable environments.

In the DTDE paradigm, policy gradient methods offer an alternative to value-based approaches, particularly for scenarios involving continuous action spaces [**?**]. Several policy gradient DTDE methods have been proposed:

- **Decentralized REINFORCE** [**?**] independently optimizes each agent's policy using gradient ascent based on rewards observed during episodes. While simple, it is less sample-efficient.

- **Independent Actor-Critic (IAC)** [**?**] Combining value estimation (critic) and policy optimization (actor), IAC enables agents to learn faster and update more frequently, improving sample efficiency.

- **Independent Proximal Policy Optimization (IPPO)** [**? ? ?**] Extending Proximal Policy Optimization (PPO) to decentralized settings, IPPO improves policy stability by limiting how much policies can change between updates.

Despite its advantages, DTDE still faces significant challenges [**? ?**]: 1. *Non-Stationarity*: As other agents learn and adapt, the environment appears dynamic and unstable to each agent, making convergence difficult; 2. *Credit Assignment*: It is hard to determine how each agent's actions contribute to the team's overall reward in cooperative tasks; 3. *Trade-Offs Between Scalability and Performance*: While DTDE scales well, its performance may be limited in tasks requiring high levels of coordination. To overcome these challenges, future research could focus on improving communication strategies during training and more robust strategies for dynamic environments.

In conclusion, the DTDE paradigm provides a powerful framework for solving distributed decision-making problems, balancing scalability, independence, and efficiency. It has been successfully applied in areas such as autonomous driving, distributed energy management, and swarm robotics. As research continues, DTDE is expected to play a larger role in real-world multi-agent systems, especially in scenarios requiring robust, independent learning.

2.3.3. Centralized Training with Centralized Execution (CTCE)

As shown in the right of Figure 3, Centralized Training with Centralized Execution (CTCE) stands out as a fully centralized mechanism to MARL decision-making, where all agents transmit their information to a central controller [**? ? ? ?**]. This central controller has access to the observations, actions, and rewards of all agents. The agents themselves do not possess policy networks and are not responsible for making decisions. Instead, they simply execute the directives issued by the central controller [**? ?**].

Multi-Agent DQN [**?**] is a classic example of the CTCE paradigm, where DQN is combined with a parameter-sharing mechanism to address tasks in multi-agent environments. Gupta et al. [**?**] firstly extends three single-agent DRL algorithms (DQN [**? ?**], TRPO, and A3C) to multi-agent systems, resulting in Multi-Agent-DQN, Multi-Agent-TRPO, and Multi-Agent-A3C. These approaches were designed to learn cooperative policies in complex, partially observable environments without requiring explicit communication between agents. The DQN algorithm based on multi-agent settings, also known as PS-DQN (Parameter Sharing DQN), effectively utilizes curriculum learning to handle increasing task complexity. By starting with

fewer agents and gradually increasing the number, the model scales well to more complex scenarios. Further, this foundational work has led to numerous enhancements and variants based on Multi-Agent DQN, each designed to address specific challenges in multi-agent systems, such as CoRe [**?**], MARL-DQN [**?**], and [**?**]. CoRe [**?**] introduces a counterfactual reward mechanism into MARL to address the credit assignment problem in cooperative settings. By computing the difference in global rewards when an agent hypothetically changes its action while others keep theirs fixed, CoRe enhances the standard DQN framework, significantly improving learning efficiency and performance in cooperative tasks. MARL-DQN [**?**] optimizes energy efficiency and resource allocation in NOMA wireless systems by using MARL framework combined with Deep Q-Networks. By combining MARL with DQN, it dynamically adjusts power and time allocation to minimize energy consumption while ensuring quality of service, outperforming traditional methods in terms of efficiency and performance. Hafiz et al. [**?**] proposed a simplified and efficient multi-agent DQN-based multi-agent system (MAS) that addresses the challenges of complexity, resource demands, and training difficulties inherent in more advanced MARL frameworks. The work introduced a shared state and reward system while maintaining agent-specific actions, which streamlines the experience replay process. The significant improvements in tasks such as Cartpole-v1[1], LunarLander-v2[2], and Maze Traversal[3] from OpenAI Gym [**?**] demonstrates the model's effectiveness and superiority.

2.3.4. Addition Taxonomies and Efforts of Communication-based MARL Algorithms

As outlined above, three primary paradigms—CTCE, DTDE, and CTDE—have emerged in the MARL domain to tackle the challenges associated with training and execution in multi-agent systems. Each of these paradigms has its strengths and limitations, yet all face inherent difficulties in handling communication among agents, which is critical for effective collaboration and decision-making.

Specifically, the CTCE paradigm, while providing a fully integrated framework for learning and execution, struggles with scalability as the system size grows. The DTDE paradigm, on the other hand, allows for independent agent training and execution, but often lacks the necessary coordination required for global task optimization. The CTDE paradigm has emerged as a widely adopted approach due to its ability to leverage centralized information during training to learn effective policies, while enabling decentralized execution to operate efficiently in distributed environments. However, even in CTDE, the communication between agents during execution is a bottleneck, prompting researchers to focus on improving communication strategies to enhance system performance.

Communication-based MARL algorithms have made significant progress in overcoming these challenges. From the perspective of communication protocols and languages, communication-based MARL methods can be categorized into three types: broadcasting communication, targeted communication, and networked communication, as shown in Figure 4. From the technical angle, we provide an overview of these communication-based MARL advancements, categorizing the algorithms into three main groups based on their focus: (1) value function-based Communication-based MARL, (2) policy search-based Communication-based MARL, and (3) Communication-based MARL algorithms designed to improve communication efficiency. These approaches represent the forefront of research in enabling agents to effectively share information, coordinate actions, and optimize performance in complex environments. Here, we provide a detailed introduction to these approaches.

Value Function-Based Communication-based MARL: For Value Function-Based Communication-based MARL Algorithms, several notable works include Differentiable Inter-Agent Learning (DIAL) [**?**] and Deep Distributed Recurrent Q-Networks (DDRQN) [**?** **?**]. Among them, DIAL facilitates effective collaboration and optimization of joint action policies by enabling the exchange of gradients of Q-functions between agents. On the other hand, DDRQN leverages recurrent neural networks to address partially observable environments, allowing agents to share critical Q-values or hidden states and make adaptive decisions in dynamic settings.

Policy Search-Based Communication-based MARL: For Policy Search-Based Communication-based MARL Algorithms, significant progress has been made with approaches such as Communication Networks (CommNet) [**?**], Bidirectional Coordinated Network (BiCNet) [**?** **?**], Multi-Agent Distributed MADDPG (MD-MADDPG) [**?** **?**], Intrinsic A3C [**?** **?**

[1]Cartpole-v1 game: `https://www.gymlibrary.dev/environments/classic_control/cart_pole/`.

[2]LunarLander-v2 game: `https://www.gymlibrary.dev/environments/box2d/lunar_lander/` and `https://github.com/topics/lunarlander-v2`.

[3]Maze Traversal game: `https://github.com/vision-mini/MazeSolverLLM`.

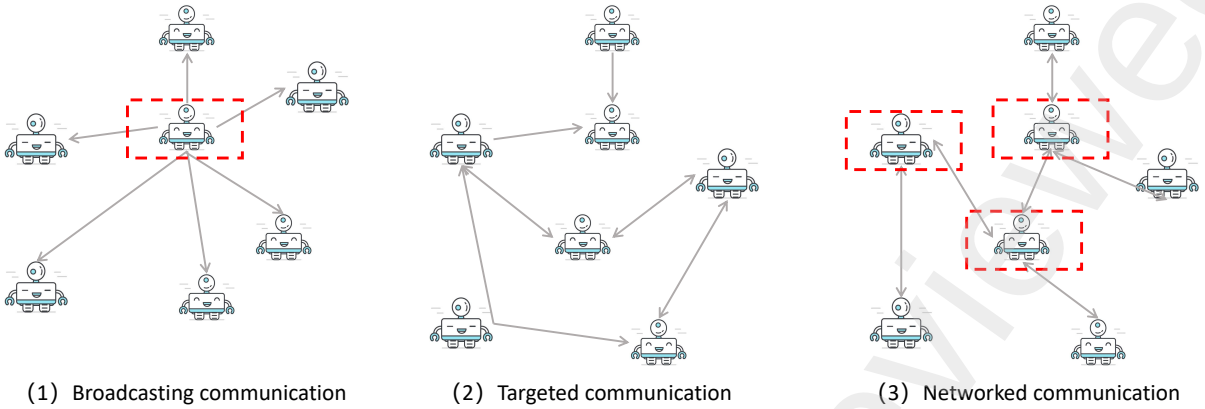


Figure 4: A schematic representation of three distinct communication methods among agents, with arrows indicating the direction of message transmission. (a) *Broadcasting communication*: The activated agent transmits messages to all other agents within the communication network. (b) *Targeted communication*: Agents selectively communicate with specific target agents based on a supervisory mechanism that regulates the timing, content, and recipients of the messages. (c) *Networked communication*: Agents engage in localized interactions with their neighboring agents within the network.

], and Multi-Agent Communication and Coordination (MACC) [**?** **?**]. Among them, CommNet [**?**] proposes a centralized but differentiable communication framework where agents share encoded signals to form a global context, improving policy decisions. BiCNet [**?** **?**] enhances coordination among agents by employing bidirectional recurrent layers, making it suitable for complex tasks. MD-MADDPG [**?** **?**] combines centralized training and decentralized execution, enabling agents to exchange critical state-action information during training for robust policy learning. Intrinsic A3C [**?** **?**] introduces intrinsic motivation to encourage effective exploration in sparse-reward scenarios, with agents sharing intrinsic rewards through communication to boost performance. Finally, Multi-Agent Communication and Coordination (MACC) [**?** **?**] focuses on adaptive communication mechanisms, providing stable and secure coordination to enhance training and execution in dynamic multi-agent environments.

Communication-based MARL Enhancing Communication Efficiency: For algorithms aimed at enhancing communication efficiency, several outstanding approaches include Attentional Communication (ATOC) [**?**], Targeted Multi-Agent Communication (TarMAC) [**?**], Inter-Agent Centralized Communication (IC3Net) [**?**]. Attentional Communication (ATOC) [**?**] employs an attention mechanism to dynamically determine when communication is necessary, achieving a balance between efficiency and coordination. Targeted Multi-Agent Communication (TarMAC) [**?**] introduces targeted attention mechanisms to direct messages to relevant teammates, minimizing redundant communication, and improving overall performance. Inter-Agent Centralized Communication (IC3Net) [**?**] incorporates a gating mechanism that allows agents to learn when and how to communicate, optimizing both the frequency and quality of interactions.

These research advances in Communication-based MARL methods demonstrate significant strides in enabling agents to share information and achieve coordinated decision-making in MAS. These advancements will pave the way for deploying MARL in real-world scenarios where efficient and effective communication is essential.

2.4. LLMs-based Multi-Agent System Taxonomies

The field of LLMs-based multi-agent systems has seen significant advancements, with researchers exploring various aspects of these systems to enhance their capabilities and applications [**?** **?**]. A comprehensive taxonomy can help categorize and understand the different dimensions of LLMs-based multi-agent systems, including architectural design, application domains, evaluation methods, and future research directions.

2.4.1. Architectural Design

The design of architectures for LLMs-based multi-agent systems is a critical component in harnessing the full potential of LLMs to enhance the capabilities of autonomous agents. Architectural design encompasses the

framework and mechanisms that enable agents to interact, adapt, and make decisions in complex and dynamic environments. This section explores two primary levels of autonomy within these systems: Adaptive Autonomy and Self-Organizing Autonomy.

- **Adaptive Autonomy:** [**? ? ? ?**] Adaptive autonomy refers to systems where agents can adjust their behavior within a predefined framework. These agents are designed to operate within the constraints set by the system architects but can adapt their actions based on the specific requirements of the task at hand. For example, in a task-specific adaptation scenario, an agent might adjust its search strategy in an information retrieval task based on the relevance of the results. In a context-aware adaptation scenario, an agent might change its communication style based on the social context of the interaction. This level of autonomy is crucial for agents that need to operate in dynamic environments where the task requirements can change over time.
- **Self-Organizing Autonomy:** [**? ? ? ? ? ?**] Self-organizing autonomy represents a higher level of autonomy where agents can dynamically adapt their behavior without predefined structures. This allows for more flexible and context-aware interactions among agents. For instance, in dynamic task allocation, agents can assign tasks to each other based on the current state of the environment and their individual skills. Emergent behavior is another key feature at this level, where agents can form coalitions or develop new strategies to solve complex problems. This level of autonomy is essential for multi-agent systems that need to operate in highly dynamic and unpredictable environments.

2.4.2. Applications

In the **social sciences** [**? ? ?**], LLMs-based agents have been used to simulate various social phenomena, providing insights into human behavior and social dynamics.

- *1) Economic Agents:* [**? ?**] LLMs can be used to model economic agents, similar to how economists use the concept of homo economicus. Experiments have shown that LLMs can produce results qualitatively similar to those of traditional economic models, making them a promising tool for exploring new social science insights. For example, in market simulation, LLMs can predict market trends and the impact of economic policies. In behavioral economics, LLMs can model individual and group decision-making processes, providing a more nuanced understanding of economic behavior.
- *2) Social Network Simulation:* [**? ? ? ?**] The Social-network Simulation System (S3) uses LLMs-based agents to simulate social networks, accurately replicating individual attitudes, emotions, and behaviors. This system can model the propagation of information, attitudes, and emotions at the population level, providing valuable insights into social dynamics. For example, it can simulate how information spreads through social networks and identify influential nodes, or model the evolution of social norms and behaviors over time.
- *3) User Behavior Analysis:* [**? ? ?**] LLMs are employed for user simulation in recommender systems, demonstrating superiority over baseline simulation systems. They can generate reliable user behaviors, improving the accuracy of recommendations. For example, in personalized recommendations, LLMs can generate user profiles and behaviors to optimize recommendation algorithms. In user engagement, LLMs can simulate user interactions to optimize user retention and engagement.

In the **natural sciences** [**? ? ?**], LLMs-based agents have been used to simulate complex systems and processes, providing insights into natural phenomena and scientific theories.

- *1) Macroeconomic Simulation:* LLMs-based agents are used for macroeconomic simulation, making realistic decisions and reproducing classic macroeconomic phenomena. These agents can simulate the impact of economic policies on the macroeconomy, providing a more accurate and dynamic model of economic behavior. For example, they can simulate the interactions between different economic sectors and their impact on the overall economy, helping policymakers make more informed decisions.
- *2) Generative Agent-Based Modeling:* This approach couples mechanistic models with generative artificial intelligence to unveil social system dynamics, such as norm diffusion and social dynamics. By combining the strengths of both approaches, researchers can model complex social systems and predict their behavior over time. For example, they can model the spread of diseases in

a population, the impact of environmental changes on ecosystems, or the evolution of social norms in a community.

In **engineering** [**?** **?** **?** **?**], LLMs-based agents have been used to develop and optimize complex systems, improving efficiency and performance.

- *1) Software Development:* LLMs-based agents are used for software development, facilitating sophisticated interactions and decision-making in a wide range of contexts. These agents can assist in code generation, bug detection, and system optimization, improving the productivity and quality of software development. For example, they can generate code snippets based on natural language descriptions, detect bugs in code, and suggest optimizations to improve performance.
- *2) Multi-Robot Systems:* LLMs-based multi-agent systems are used to simulate complex real-world environments effectively, enabling interactions among diverse agents to solve various tasks. These systems can coordinate the actions of multiple robots, optimizing their behavior to achieve common goals. For example, they can be used in search and rescue operations, where multiple robots need to coordinate their actions to locate and rescue victims.

3. Simulation Environments of Multi-Agent Decision-Making

First and foremost, the designs and implementations of multi-agent cooperative simulation environments are crucial in the historical research of multi-agent decision-making, which are widely utilized in practical applications and production. These simulation environments form the foundation for conducting efficient and effective studies in multi-agent cooperative decision-making. Specifically, a dynamic multi-agent cooperative decision-making environment refers to predetermined scenarios and platforms where multiple agents collaborate to solve problems, complete tasks, and achieve goals. Such environments provide not only a platform for testing and validating various intelligent decision-making algorithms but also help us better understand the behaviors and interactions of agents in dynamic settings. By simulating these interactions, researchers can gain insights into how agents coordinate and adapt to changing conditions, thereby improving the robustness and efficiency of multi-agent systems in real-world applications. Consequently, the importance of these simulation environments cannot be overstated. They serve as a testing ground for theoretical models, allowing researchers to observe the practical implications of their intelligent algorithms. Additionally, these platforms help in identifying potential issues and refining strategies before deployment in actual scenarios, ensuring that the agents are well-prepared to handle the complexities of real-world environments. In Table 2, a wide range of simulated environments is listed. Next, we will delve into these environments one by one, emphasizing their significance and features for future development.

3.1. MARL-based Simulation Environments

This section provides an overview of several widely-used simulation environments designed for MARL. These platforms, such as Multi-Agent Particle Environment [**?** **?** **?**], and PettingZoo [**?**], offer diverse scenarios and functionalities for exploring cooperative and competitive agent interactions in both simple and complex tasks.

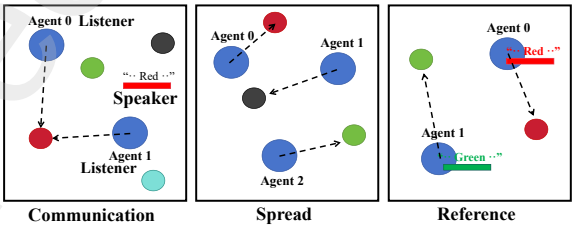


Figure 5: Typical Scenarios in Multi-Agent Particle Environment (MPE).

3.1.1. Several Widely-used Environments on MARL

Multi-Agent Particle Environment (MPE) [**?** **?** **?**] is a versatile and widely-used MARL platform designed for research in both cooperative and competitive settings. Developed by OpenAI, it is primarily known for being the testing environment of the MADDPG algorithm [**?**]. MPE is a time-discrete, space-continuous 2D platform designed for evaluating MARL algorithms.

Figure 5, initially derived from Malloy et al. [**?**], illustrates various scenarios within the Multi-Agent Particle Environment (MPE), including tasks such as adversarial interactions, cooperative crypto, object pushing, and team-based world navigation. Compatible with the widely-used Gym interface, it supports a variety of tasks ranging from fully cooperative to mixed cooperative-competitive scenarios, such as `simple_adversary`, `simple_crypto`, `simple_spread`, `simple_speaker_listener`,

Table 2: Diverse MARL-based and LLMs-based Simulated Environments for Multi-Agent Systems.

Categories	Multi-Agent System Environments
MARL-based	Multi-Agent Particle Environment (MPE) [**? ? ?**], PettingZoo [**?**], SMAC [**?**], SMAC-v2 [**?**], GFootball [**?**], Gym-Microrts [**?**], MAgent [**?**], Dexterous Hands [**? ?**], OpenAI Gym [**?**], Gym-MiniGrid [**?**][4], Melting Pot [**?**][5], Capture The Flag[6] [**?**], VillagerAgent [**?**], Minecraft [**? ? ?**], Unity ML-Agents [**?**], SUMO[7] [**?**], Hanabi Learning [**? ?**], Predator-Prey [**? ?**]
LLMs-based	TDW-MAT [**? ?**], C-WAH [**?**], Cuisineworld [**?**], AgentScope [**?**], RoCoBench [**?**], Generative Agents [**? ?**], SocialAI school [**? ?**], Welfare Diplomacy [**?**]

`and simple_world_comm`[8]. Each scenario highlights distinct cooperative and competitive dynamics among agents. MPE allows agents to interact and strategize within a visually simplistic UI where particles represent different entities. MPE is a open-source platform that widely adopted in the multi-agent system research, enabling extensive customization and contributing to its role as a standard tool for studying complex multi-agent dynamics.

Overall, MPE is a pivotal resource in the MARL community, offering a well-rounded platform for experimentation and algorithm comparison. Its design and functionality have made it an indispensable tool for researchers seeking to push the boundaries of what is possible in multi-agent systems.

StarCraft Multi-Agent Challenge (SMAC)[9] [**?**] is a widely-used benchmark for MARL that focuses on decentralized micromanagement tasks in the popular real-time strategy game StarCraft II[10]. In SMAC, multiple agents control individual units and must learn to cooperate and coordinate actions based on local, partial observations. The agents face complex challenges, including coordinating combat techniques like focus fire, kiting, and positioning, while the opponent is controlled by the built-in StarCraft II AI. SMAC emphasizes problems such as partial observability, decentralized decision-making, and multi-agent credit assignment. The environment is structured to simulate real-world scenarios where agents must learn to collaborate without full knowledge of the global state. Agents' observations are restricted to a limited field of view, forcing them to rely on local information for decision-making. As shown in Figure 6, these multi-agent cooperative decision-making environments are respectively `2s vs 3z`, `5m vs 6m`, `6h vs 8z`, `MMM2`, where the inside numbers represent the number of units and the letters represent the unit types in general. In recent years, SMAC has become a standard benchmark for evaluating MARL algorithms, offering a rigorous and challenging environment for advancing the field.

Figure 6: Several Typical Scenarios in StarCraft Multi-Agent Challenge (SMAC).

StarCraft Multi-Agent Challenge 2 (SMACv2)[11] [**? ? ?**] However, SMAC [**?**] has limitations, including insufficient stochasticity and partial observability, which allows agents to perform well with simple open-loop policies. To address these shortcomings, SMACv2 introduces *procedural content generation (PCG)*, randomizing team compositions and agent positions, ensuring agents face novel, diverse scenarios. Several multi-agent decision-making scenarios are depicted in Figure 7, which are from Benjamin et al. [**?**

[8] Multi-Agent Particle Environment: `https://github.com/openai/multiagent-particle-envs/tree/master/multiagent/scenarios`.

[9] StarCraft Multi-Agent Challenge (SMAC): `https://github.com/oxwhirl/smac`.

[10] StarCraft II: `https://starcraft2.blizzard.com/`.

[11] StarCraft Multi-Agent Challenge 2 (SMACv2): `https://github.com/oxwhirl/smacv2`.

]. This requires more sophisticated, closed-loop policies that condition on both ally and enemy information. Additionally, SMACv2 includes the *Extended Partial Observability Challenge (EPO)*, where enemy observations are masked stochastically, forcing agents to adapt to incomplete information and communicate more effectively. SMACv2 thus represents a major evolution of the original benchmark, addressing key gaps such as the lack of stochasticity and meaningful partial observability. These changes make SMACv2 a more challenging environment, requiring agents to generalize across varied settings and improve coordination, communication, and decentralized decision-making. Overall, SMACv2 provides a more rigorous testbed for advancing the field of cooperative MARL.

(a) Terran (b) Protoss

Figure 7: Several scenarios from SMACv2 showing agents battling the built-in AI.

Google Research Football Environment (GFootball) [**?**] is a state-of-the-art multi-agent simulation environment developed by the Google Research Brain Team. It is specifically designed for reinforcement learning research and is built on top of the open-source football game, GamePlay Football. GFootball is compatible with the OpenAI Gym API, making it a versatile tool not only for training intelligent agents but also for allowing players to interact with the built-in AI or trained agents using a keyboard or game controller. GFootball features an advanced, physics-based 3D football simulator where agents can be trained to play football, offering a challenging yet highly customizable platform for testing novel reinforcement learning algorithms and ideas. GFootball is tailored for multi-agent experiments and multiplayer scenarios, enabling the exploration of more complex interactions and strategies. GFootball supports various scenarios, including full-game simulations with varying difficulty levels, as well as simpler tasks in the Football Academy that focus on specific skills like passing or scoring.

Moreover, training agents for the "*Football Benchmark*" can be quite challenging. To help researchers efficiently test and iterate on new ideas, researchers provide a toolset called "Football Academy", as illustrated in Figure 8, which includes a series of scenarios with varying levels of difficulty. These scenarios range from simple setups, such as a single player scoring against an open goal (e.g., approaching an open goal, scoring in an open goal, or scoring while running), to more complex team-based setups, where a controlled team must break through specific defensive formations (e.g., scoring while running against a goalkeeper, passing and shooting against a goalkeeper, and 3v1 against a goalkeeper). Additionally, the toolset covers common situations in football matches, such as corner kicks, simple counterattacks, and complex counterattacks. Lastly, as an famous open-source GitHub project[12], it offers a unique opportunity for researchers and pushes the boundaries of AI research in a reproducible and scalable manner.

Figure 8: Typical examples of Training Scenarios in Football Academy.

Unity Machine Learning-Agents Toolkit[13] [**?**] is an open-source platform designed to enable games and simulations to serve as environments for training intelligent agents. Built on Unity's powerful game engine, it supports a wide range of AI and machine learning methods, including reinforcement learning, imitation learning, and neuroevolution, through an intuitive Python API. The platform includes state-of-the-art algorithm implementations (based on PyTorch), allowing researchers and developers to train agents for 2D, 3D, and VR/AR applications.

ML-Agents is particularly useful for training NPC behaviors in diverse scenarios, automated testing of game builds, and evaluating game design decisions. It features a highly flexible simulation environment with

[12]Google Research Football: `https://github.com/google-research/football`.

[13]Unity ML-Agents Toolkit: `https://github.com/Unity-Technologies/ml-agents`.

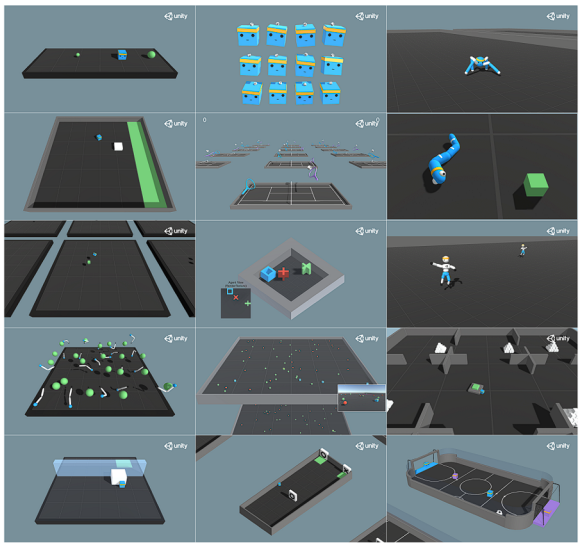


Figure 9: Typical Training Scenarios in Unity Machine Learning-Agents Toolkit (released version: v0.11). From Left-to-right, up-to-down: (a) Basic, (b) 3DBall, (c) Crawler, (d) Push Block, (e) Tennis, (f) Worm, (g) Bouncer, (h) Grid World, (i) Walker, (j) Reacher, (k) Food Collector, (l) Pyramids, (m) Wall Jump, (n) Hallway, (o) Soccer Twos [**?**].

realistic visuals, physics-driven interactions, and rich task complexity. By integrating tools for creating custom environments and supporting multi-agent and adversarial settings, the toolkit bridges the gap between AI research and practical applications in game development.

As seen from Figure 9, it depicts several typical multi-agent environments from the previous work of Juliani et al. [**?**]. The platform also provides key components such as a Python API, Unity SDK, and pre-built environments, enabling users to customize and evaluate their algorithms in Unity's interactive and visually rich settings. With its versatility and accessibility, Unity ML-Agents Toolkit has become an indispensable resource for both AI researchers and game developers, driving innovation in artificial intelligence and simulation-based learning.

Gym-Microrts[14] [**?**] (pronounced "*Gym-micro-RTS*") is a fast and affordable reinforcement learning (RL) platform designed to facilitate research in full-game Real-Time Strategy (RTS) games. Unlike traditional RTS research that demands extensive computational resources, Gym-µRTS allows training advanced agents using limited hardware, such as a single GPU and CPU setup, within reasonable timeframes. Figure 10 showcases a match between our best-trained agent (top-left) and CoacAI (bottom-right), the 2020 µRTS AI competition champion. The agent employs an efficient strategy, starting with resource harvesting and a worker rush to damage the enemy base, transitioning into mid-game combat unit production to secure victory.

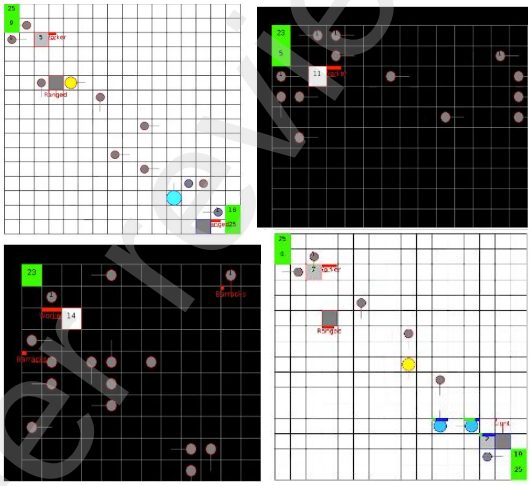


Figure 10: Screenshot of our best-trained agent (top-left) playing against CoacAI (bottom-right), the 2020 μRTS AI competition champion [**?**].

The platform offers a simplified RTS environment that captures the core challenges of RTS games, including combinatorial action spaces, real-time decision-making, and partial observability. Gym-µRTS employs a low-level action space, enabling fine-grained control over individual units without AI assistance, which poses unique challenges and opportunities for RL algorithms. It supports Proximal Policy Optimization (PPO) and incorporates techniques like invalid action masking, action composition, and diverse training opponents to enhance training efficiency and agent performance.

Gym-µRTS has demonstrated its effectiveness by producing state-of-the-art DRL agents capable of defeating top competition bots, such as CoacAI. The platform is open-source and provides all necessary tools for researchers to explore and advance RL techniques in RTS games, making it a valuable resource for both AI researchers and gaming enthusiasts.

MAgent[15] [**?**] is an open-source platform specifically designed to support large-scale MARL research, with a focus on exploring Artificial Collective Intelligence (ACI). Unlike traditional MARL platforms, MAgent excels in handling scenarios involving hundreds to

[14] Gym-Microrts: `https://github.com/kered9/gym-microrts`.

[15] MAgent: `https://github.com/geek-ai/MAgent`.

millions of agents, making it ideal for studying complex interactions and emergent behaviors in large populations.

For instance, as shown in Figure 11, the "*Pursuit*" scenario is a classic example designed to showcase the emergent cooperative behaviors of agents in a predator-prey environment. In this setup, predators work together to capture preys while the preys attempt to evade capture. Each predator receives rewards for successfully attacking a prey, while preys are penalized if caught. Over time, predators learn to form collaborative strategies, such as surrounding and trapping preys, highlighting the emergence of local cooperation.

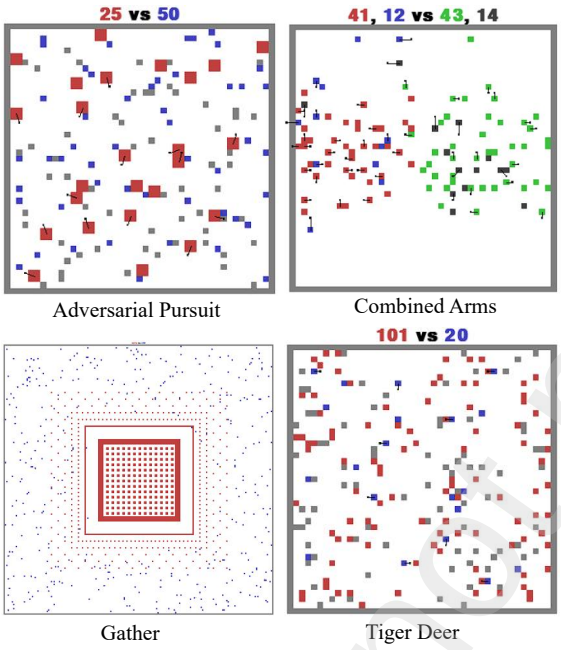


Figure 11: Illustrations of one of the typical running example in MAgent, called "*Pursuit*" [**?**].

The platform is based on a grid-world model where agents can perform actions such as moving, turning, or attacking, while perceiving both local and global information. Through a flexible Python interface, researchers can easily customize the state space, action space, and reward mechanisms, enabling the rapid creation of complex cooperative or competitive environments. MAgent comes with several built-in scenarios, such as pursuit, resource gathering, and team-based battles, which highlight emergent social behaviors like cooperative strategies, competitive dynamics, and resource monopolization.

MAgent is highly scalable, leveraging GPU-based parallelism to simulate large-scale interactions efficiently. It also provides intuitive visualization tools for real-time observation of agent behaviors, facilitating analysis and debugging. Its flexibility and scalability make MAgent a powerful tool for MARL research, enabling the study of large-scale agent interactions, emergent behaviors, and the dynamics of artificial societies.

3.2. LLMs Reasoning-based Simulation Environments

LLMs-based multi-agent systems have become an essential tool for enhancing the collaboration, reasoning, and decision-making capabilities of autonomous agents [**?**]. By integrating LLMs with simulation platforms, researchers can create complex test environments to explore the interactions of multi-agent systems in various tasks and scenarios. These simulation environments not only provide rich dynamic testing scenarios but also promote the widespread application of LLMs in task planning, coordination, and execution. The following will introduce several widely used simulation platforms for LLM multi-agent systems.

ThreeDWorld Multi-Agent Transport (TDW-MAT)[16] [**?** **?**] is a simulation environment for multi-agent embodied task, which is extended from the ThreeDWorld Transport Challenge [**?**] and is designed for visually-guided task-and-motion planning in physically realistic settings. It operates within the ThreeDWorld (TDW) platform, which offers high-fidelity sensory data, real-time physics-driven interactions, and near-photorealistic rendering. In TDW-MAT, embodied agents are tasked with transporting objects scattered throughout a simulated home environment using containers, emphasizing the need for coordination, physics awareness, and efficient planning. For instance, in the common scenario shown in Figure 12, the agent must transport objects scattered across multiple rooms and place them on the bed (marked with a green bounding box) in the bedroom.

Communicative Watch-And-Help (C-WAH)[17] [**?**] is a realistic multi-agent simulation environment and an extension of the Watch-And-Help Challenge platform, VirtualHome-Social [**?**]. C-WAH places a greater emphasis on cooperation and enhances communication between agents compared to VirtualHome-Social. Built on the VirtualHome-Social, C-WAH simulates common household activities where agents must collaborate to complete tasks such as preparing meals, washing dishes, and setting up a dinner table. As shown in Figure 13,

[16]ThreeDWorld Multi-Agent Transport: `https://github.com/threedworld-mit/tdw`.

[17]Communicative Watch-And-Help: `https://github.com/xavierpuigf/watch_and_help`.

Figure 12: An overview of one of the example task in ThreeDWorld Transport Challenge [**? ?**].

C-WAH supports both symbolic and visual observation modes, allowing agents to perceive their surroundings either through detailed object information or egocentric RGB and depth images.

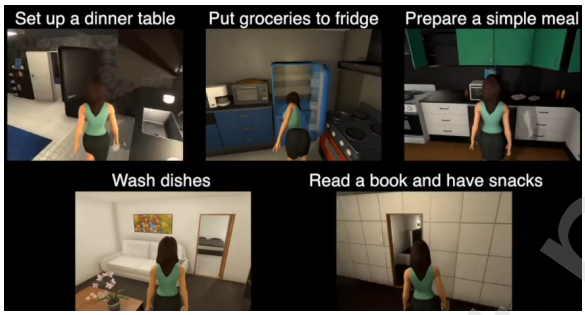


Figure 13: An typical object-moving task leveraging LLMs-based embodied agents within the Communicative Watch-And-Help [**?**].

Cuisineworld[18] [**?**] is a virtual kitchen environment designed to evaluate and enhance multi-agent collaboration and coordination (i.e., the working efficiency) in a gaming context. As shown in Figure 14, in this scenario, multiple agents work together to prepare and complete dish orders within a limited time frame. The tasks range from simple preparations, like chopping ingredients, to complex cooking processes that involve multiple tools and steps. CuisineWorld is equipped with a textual interface, and it supports various levels of difficulty, making it a flexible and robust testbed for assessing the planning and scheduling capabilities of Large Foundation Models (LFMs). The environment also introduces a "Collaboration Score" (CoS) metric to measure the efficiency of agent coordination as task demands increase, providing a comprehensive benchmark for multi-agent system performance in dynamic and cooperative settings.

[18]Cuisineworld: `https://mindagent.github.io/`.

Figure 14: An typical multi-agent cooperative scenario in the CuisineWorld platform [**?**].

AgentScope[19] [**?**] is a innovative, robust and flexible multi-agent platform designed to empower developers in building advanced multi-agent systems by leveraging the potential of LLMs. At its core, the platform employs a process-based message exchange mechanism, simplifying the complexities of agent communication and collaboration. This approach ensures smooth and efficient agent interaction, enabling developers to focus on designing workflows rather than low-level details. The platform stands out for its comprehensive fault-tolerance infrastructure, which includes retry mechanisms, rule-based corrections, and customizable error-handling configurations. AgentScope also excels in multi-modal support, seamlessly integrating text, images, audio, and video into its workflows. By decoupling data storage and transfer, it optimizes memory usage and enhances scalability, making it ideal for applications requiring rich multi-modal interactions. Additionally, its actor-based distributed framework enables efficient parallel execution and supports hybrid deployments, bridging the gap between local and distributed environments with ease.

Moreover, to improve user interaction with multiple agents, AgentScope assigns distinct colors and icons to each agent, as shown in Figure 15, providing clear visual differentiation in both the terminal and web interface. Designed with user accessibility in mind, AgentScope provides intuitive programming tools, including pipelines and message centers, which streamline the development process. Its interactive user interfaces, both terminal- and web-based, allow developers to monitor performance, track costs, and engage with agents effectively. These features position AgentScope as a state-of-the-art platform for advancing multi-agent

[19]AgentScope: `https://github.com/modelscope/agentscope`.

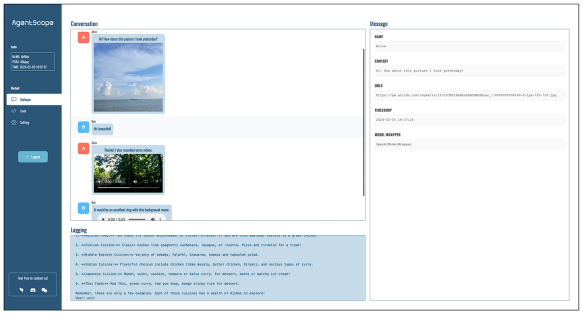


Figure 15: The official multi-modal interaction Web UI page between agents in the AgentScope platform [**?**].

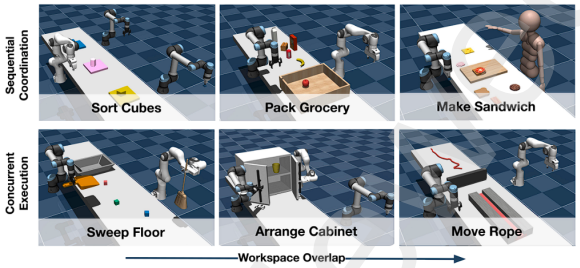


Figure 16: An overview of RoCoBench, a collection of six multi-robot collaboration tasks set in a tabletop manipulation environment. The scenarios encompass a diverse range of collaborative challenges, each demanding distinct communication and coordination strategies between robots, incorporating familiar, intuitive objects designed to align with the semantic understanding capabilities of LLMs [**?**].

systems, combining ease of use with cutting-edge technology.

RoCoBench[20] RoCoBench is a benchmark platform, proposed by *Mandi et al.* [**?**], designed to evaluate and enhance the collaborative capabilities of multi-robot systems powered by LLMs. Built as an extension to the RoCo project[21], RoCoBench provides a realistic simulation environment where robotic agents interact and collaborate to complete complex tasks, as shown in Figure 16, such as sorting packages, assembling components, or preparing a workspace. RoCoBench places a strong emphasis on communication-driven collaboration, integrating both symbolic and visual interaction modes to enable robots to perceive and respond to their environment effectively. Each robot is equipped with LLMs-powered reasoning, facilitating real-time dialogue and coordination. Correspondingly, the benchmark introduces a "Collaboration Efficiency Metric" (CEM) to evaluate the effectiveness of multi-robot teamwork, taking into account factors like task completion time, resource allocation, and the quality of inter-robot communication. RoCoBench serves as a comprehensive platform for evaluating the potential of LLMs in driving dialectic multi-robot collaboration, offering a scalable and flexible environment for developers and researchers alike

Generative Agents[22] *Park et al.* [**?** **?**] introduces Generative Agents, a groundbreaking framework for simulating human behavior in interactive virtual worlds. These agents exhibit realistic individual and collective behaviors by incorporating dynamic memory, self-reflection, and action planning capabilities. The system leverages LLMs to store, retrieve, and synthesize memories into higher-level reasoning, enabling agents to adapt their actions based on personal experiences and evolving environmental changes. As illustrated in Figure 16, they present an interactive sandbox environment called *Smallville*, akin to "The Sims," where 25 distinct virtual agents live, interact, and carry out daily activities. Each agent has a detailed initial profile, including personal traits, relationships, and goals, stored as "seed memories." Agents engage in natural language-based dialogues and demonstrate social behaviors such as hosting events, making new acquaintances, and responding to user interventions. Generative Agents enable interactive applications in fields such as simulating realistic social dynamics for games and training simulations; designing dynamic, non-scripted virtual worlds for interactive systems; and exploring theories and behaviors in a controlled yet realistic virtual setting. The evaluations revealed the critical role of *memory retrieval, self-reflection, and action planning* in achieving coherent agent behaviors. Common issues, such as exaggerated responses and overly formal communication, were identified as areas for improvement. Generative Agents push the boundaries of human behavior simulation, offering a robust framework for creating autonomous, memory-driven virtual agents.

SocialAI school[23] *Kovač et al.* [**?** **?**] introduces The SocialAI School, a novel framework designed to explore and develop socio-cognitive abilities in artificial agents. The study emphasizes the importance of socio-cognitive skills as foundational to human intelligence and cultural evolution. As shown in Figure 18, the SocialAI School provides a customizable suite of procedurally generated environments that enable systematic re-

[20] RoCoBench: `https://project-roco.github.io/`.
[21] RoCo Project: `https://project-roco.github.io/`.
[22] Generative Agents: `https://youmingyeh.github.io/cs-book/papers/generative-agents`.
[23] SocialAI school project: `https://sites.google.com/view/socialai-school`.

Figure 17: Generative agents serve as realistic simulations of human behavior, designed for interactive applications. In a sandbox environment inspired by The Sims, twenty-five agents engage in activities such as planning their routines, sharing updates, building relationships, and collaborating on group events, while allowing users to observe and interact with them. [**?** **?**].

search into the socio-cognitive abilities required for artificial agents to interact with and contribute to complex cultures. Built on MiniGrid, it provides procedural environments for RL and LLMs-based agents to study social skills like joint attention, imitation, and scaffolding. Open-source and versatile, it enables diverse research, including generalizing social inferences, role reversal studies, and scaffolded learning. The SocialAI School represents a significant step toward enriching AI systems with socio-cognitive abilities inspired by human development.

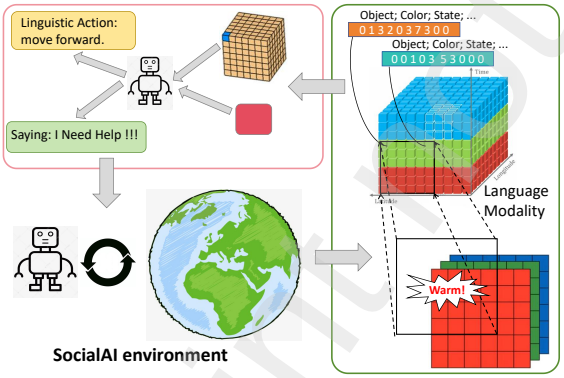


Figure 18: A clear workflow of an agent acting in the SocialAI school environment [**?** **?**].

Welfare Diplomacy[24] [**?**] is an innovative variant of the zero-sum game Diplomacy, designed to evaluate the cooperative capabilities of multi-agent systems. Unlike the original game, which focuses on a single winner, Welfare Diplomacy introduces a general-sum framework where players balance military conquest with investments in domestic welfare. Players accumulate Welfare Points (WPs) throughout the game by prioritizing welfare over military expansion, and their total utility at the end of the game is determined by these points, removing the notion of a single "winner". Welfare Diplomacy enables clearer assessments of cooperation and provides stronger incentives for training cooperative AI agents. Players take on the roles of European powers, negotiating, forming alliances, and strategizing to compete for key supply centers. Orders are submitted and resolved simultaneously, with the goal of controlling a specified number of SCs to achieve victory, emphasizing a balance between cooperation and betrayal. Based on these rules, Welfare Diplomacy implements themselves via an open-source platform, and develops zero-shot baseline agents using advanced language models like GPT-4 [**?** **?**]. Experiments reveal that while these agents achieve high social welfare through mutual demilitarization, they remain vulnerable to exploitation, highlighting room for future improvement.

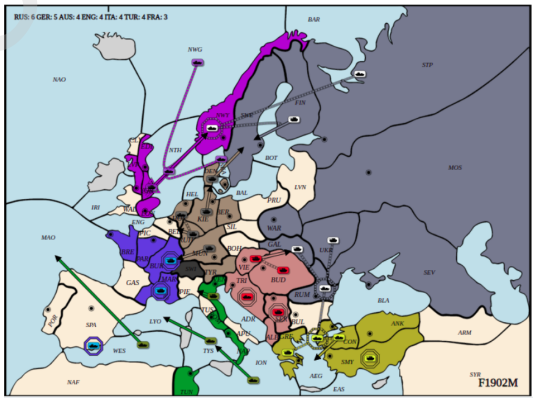


Figure 19: The Balkans in the Diplomacy map in Welfare Diplomacy [**?**].

In summary, these cutting-edge LLMs-powered simulation environments—ranging from task-specific platforms like TDW-MAT [**?** **?**] and CuisineWorld [**?**] to socially-driven frameworks such as Generative Agents [**?**] and the SocialAI School [**?**]—highlight the transformative potential of integrating advanced AI reasoning and multi-agent systems. By fostering research on collaboration, social cognition, and cooperative decision-making, these tools not only advance our understanding of AI's capabilities but also pave the way for practical applications in dynamic, real-world scenarios.

[24] Welfare Diplomacy: `https://github.com/mukobi/welfare-diplomacy`.

4. Practice Applications of Multi-Agent Decision-Making

Multi-agent cooperative decision-making has a wide range of practical applications across various domains. In this section, we delve into the practical applications of multi-agent decision-making, focusing on how advanced methods, particularly multi-agent MARL, are employed to address complex challenges in dynamic and evolving environments. We explore the contributions of advanced multi-agent systems across domains such as agriculture, disaster rescue, military simulations, traffic management, autonomous driving, and multi-robot collaboration. A broad array of applications applications is presented in Table 3. In the following, we will provide a detailed introduction to these applications, highlighting their impact and potential for future advancements.

4.1. MARL-based Intelligent Applications

Recently, a variety of MARL methods have been developed, fostering efficient collaboration, strategic learning, and adaptive problem-solving in multi-agent systems [**?** **?** **?** **?**]. Below, we highlight notable contributions that demonstrate the application of MARL in enhancing multi-agent collaboration and performance

In *smart precious agriculture and continuous pest disease detection*, Seewald et al. [**?**] addressed the challenge of continuous exploration for multi-agent systems with battery constraints by integrating ergodic search methods with energy-aware coverage. In disaster rescue, Qazzaz et al. [**?**] proposed a novel technique using a reinforcement learning multi Q-learning algorithm to optimize UAV connectivity operations in challenging terrain. Samad et al. [**?**] presents a cloud-based multi-agent framework for efficiently managing aerial robots in disaster response scenarios, aiming to optimize rescue efforts by autonomously processing real-time sensory data to locate and assist injured individuals.

In *military confrontations*, Qi et al. [**?**] designed a distributed MARL framework based on the actor-work-learner architecture, addressing the issues of slow sample collection and low training efficiency in MARL within the MaCA [**?**] and SMAC 3D realtime gaming [**?** **?**] military simulation environments. Benke et al. [**?**] proposed a computational model for agent decision-making that incorporates strategic deception, enhancing the representation of deceptive behaviors in multi-agent simulations for military operations research. Sutagundar et al. [**?**] proposed a Context Aware Agent based Military Sensor Network (CAMSN) to enhance multi-sensor image fusion, using node and sink-driven contexts, forming an improved infrastructure for multi-sensor image fusion.

In *efficient limited-bandwidth communication* field, Wang et al. [**?**] proposed a method called IMAC (Informative Multi-Agent Communication) to address the problem of limited-bandwidth communication in MARL.

In the research of *UAV swarm communications against jamming*, Lv et al. [**?**] proposed a MARL-based scheme to optimize relay selection and power allocation. This strategy leverages network topology, channel states, and shared experiences to improve policy exploration and stability, ultimately enhancing anti-jamming performance.

In *UAV pursuit-evasion* [**?** **?** **?**], Kouzeghar [**?**] proposed a decentralized heterogeneous UAV swarm approach for multi-target pursuit using MARL technique and introduced a variant of the MADDPG [**?**] to address pursuit-evasion scenarios in non-stationary environments with random obstacles. Alexopoulos et al. [**?**] addressed the challenge of pursuit-evasion games involving two pursuing and one evading unmanned aerial vehicle (UAV) by introducing a hierarchical decomposition of the game. Luo et al. [**?**] proposed a cooperative maneuver decision-making method for multi-UAV pursuit-evasion scenarios using an improved MARL approach, which incorporates an enhanced CommNet network with a communication mechanism to address multi-agent coordination.

In large-scale traffic signal/flow control, Wang [**?**] proposed a curiosity-inspired algorithm to optimize safe and smooth traffic flow in autonomous vehicle on-ramp merging; Chu et al. [**?**] proposed a fully scalable and decentralized multi-agent deep reinforcement learning algorithm based on the advantage actor-critic (A2C) method.

In *autonomous driving* area, a large number of superior multi-agent decision-making algorithms and models are continuously being explored and devised. Xue et al. [**?**] developed a two-stage system framework for improving Multi-Agent Autonomous Driving Systems (MADS) by enabling agents to recognize and understand the Social Value Orientations (SVOs) of other agents. Liu et al. [**?**] proposed the Personality Modeling Network (PeMN), which includes a cooperation value function and personality parameters to model diverse interactions in highly interactive scenarios, addressing the issue of diverse driving styles in autonomous driving. Wen et al. [**?**] proposed a safe reinforcement learning algorithm called Parallel Constrained Policy Optimization (PCPO) based on the

Table 3: Categorized Applications of MARL and LLMsin Diverse Domains.

Category	Application Area	Works / References
MARL-based MAS	Smart Precious Agriculture & Disaster Rescue	Seewald et al. [?], Qazzaz et al. [?], Samad et al. [?], Boubin et al. [?], Li et al. [?], Mahajan et al. [?]
	Military Confrontations	Qi et al. [?], Benke et al. [?], Sutagundar et al. [?], Vangaru et al. [?], Wang et al. [?], MaCA [?], SMAC [?], SMAC-v2 [?]
	UAV Pursuit-Evasion & Swarm Communications & Navigation	Kouzeghar [?], Alexopoulos et al. [?], Luo et al. [?], Lv et al. [?], Xue et al. [?], Rezwan et al. [?], Baroomi et al. [?]
	Traffic Signal/Flow Control	Wang [?], Chu et al. [?], Aboueleneen et al. [?], Yu et al. [?], Sun et al. [?], Azfar et al. [?], Bokade et al. [?], Kwesiga et al. [?], Zhang et al. [?]
	Autonomous Driving	Xue et al. [?], Liu et al. [?], Wen et al. [?], Jayawardana et al. [?], Liu et al. [?], Formanek et al. [?], Zhang et al. [?], Kotoku et al. [?], Hua et al. [?]
	Multiple Robots Collaborative	Georgios et al. [?], Silva et al. [?], Huang et al. [?], Cena et al. (SMART) [?], Kevin (SCRIMMAGE) [?], Liu et al. [?]
LLMs-based MAS	Multi-Agent Collaboration	Wu et al. (AutoGen) [?], Xiao et al. (CoE) [?], Chen et al. (AgentVerse) [?], Liu et al. (DyLAN) [?], Zhang et al. (CoELA) [?]
	Gaming Interaction	Xu et al. (LLM-Werewolf) [?], Gong et al. (MindAgent) [?], Xie et al. [?], Lin et al. [?], Jia et al. (GameFi) [?], Yin et al. (MIRAGE) [?], Zhang et al. (DVM) [?], Bonorino et al. [?]
	Autonomous Driving	Zheng et al. (PlanAgent) [?], Luo et al. (SenseRAG) [?], Mahmud et al. [?], Peng et al. (LearningFlow) [?], Karagounis et al. [?], Luo et al. [?], Gao et al. [?], Hegde et al. [?]
	Multi-Modal Application	Wang et al. (LangGraph) [? ?], Zhang et al. (CrewAI) [? ?], Zheng et al. (PlanAgent) [?], Wang et al. (MLLM-Tool) [?]

actor-critic architecture to address the issues of unexplainable behaviors and lack of safety guarantees in autonomous driving. Jayawardana et al. [?] proposed enabling socially compatible driving by leveraging human driving data to learn a social preference model, integrating it with reinforcement learning-based AV policy synthesis using Social Value Orientation theory.

In *multiple robots collaborative* fields, Georgios et al. [?] introduces a novel cognitive architecture for large-scale multi-agent Learning from Demonstration (LfD), leveraging Federated Learning (FL) to enable scalable, collaborative, and AI-driven robotic systems in complex environments. Silva et al. [?] address the challenges and limitations in evaluating intelligent collaborative robots for Industry 4.0. The review emphasizes the urgent need for improved evaluation methods and standards to account for the complexities posed by human variability, AI integration, and advanced control systems in industrial environments. Huang et al. [?] presents a multi-agent reinforcement learning approach using the MADDPG algorithm, enhanced with an experience sample optimizer, to train swarm robots for autonomous, collaborative exploration on Mars. This approach outperforms traditional DRL algorithms in efficiency as the number of robots and targets increases. The SMART multi-agent robotic system [?] is a comprehensive and advanced platform designed for executing coordinated robotic tasks. It integrates both hardware components, such as robots and IP-Cameras, and software agents responsible for image processing, path planning, communication, and decision-making. By utilizing Work-Flow Petri Nets for modeling and control, the system effectively ensures coordination and successful task execution even in unstructured environ-

ments.

Furthermore, the well-known project, Simulating Collaborative Robots in a Massive Multi-agent Game Environment (SCRIMMAGE)[25] [**?**], tackles the high costs of field testing robotic systems by offering a flexible and efficient simulation environment specifically designed for mobile robotics research. Unlike many existing simulators that are primarily designed for ground-based systems with high-fidelity multi-body physics models, SCRIMMAGE focuses on simulating large numbers of aerial vehicles, where precise collision detection and complex physics are often unnecessary. SCRIMMAGE is designed to be highly adaptable, with a plugin-based architecture that supports various levels of sensor fidelity, motion models, and network configurations. This flexibility allows the simulation of hundreds of aircraft with low-fidelity models or a smaller number with high-fidelity models on standard consumer hardware. Overall, SCRIMMAGE[26] provides a robust and scalable solution for testing and refining robotic algorithms in a controlled virtual environment, significantly reducing the risks and costs associated with physical testing.

Liu et al. [**?**] proposed the Learning before Interaction (LBI) framework, a novel approach designed to enhance multi-agent decision-making through generative world models. Traditional generative models struggle with trial-and-error reasoning, often failing to produce reliable solutions for complex multi-agent tasks. To address this limitation, LBI integrates a language-guided simulator into the MARL pipeline, enabling agents to learn grounded reasoning through simulated experiences. LBI consists of a world model composed of a dynamics model and a reward model. The dynamics model incorporates a vector quantized variational autoencoder (VQ-VAE) [**?**] for discrete image representation and a causal transformer to autoregressively generate interaction transitions. Meanwhile, the reward model employs a bidirectional transformer trained on expert demonstrations to provide task-specific rewards based on natural language descriptions. LBI further distinguishes itself by generating explainable interaction sequences and reward functions, providing interpretable solutions for multi-agent decision-making problems. By addressing challenges such as the compositional complexity of MARL environments and the scarcity of paired text-image datasets, LBI represents a significant advancement in the field.

Ye et al. [**?**] proposed an adaptive genetic algorithm (AGA) that dynamically adjusts crossover and mutation populations, leveraging the Dubins car model and state-transition strategies to optimize the allocation of heterogeneous UAVs in suppression of enemy air defense missions. Radac *et al.* combine two model-free controller tuning techniques linear virtual reference feedback tuning (VRFT) and nonlinear state-feedback Q-learning as a novel mixed VRFT-Q learning method [**?**]. VRFT is initially employed to determine a stabilizing feedback controller using input-output experimental data within a model reference tracking framework. Subsequently, reinforcement Q-learning is applied in the same framework, utilizing input-state experimental data gathered under perturbed VRFT to ensure effective exploration. Extensive simulations on position control of a two-degrees-of-motion open-loop stable multi input-multi output (MIMO) aerodynamic system (AS) demonstrates the mixed VRFT-Q's significant performance improvement over the Q-learning controllers and the VRFT controllers.

To address the lack of a general metric for quantifying policy differences in MARL problems, Hu et al. [**?**] proposed the Multi-Agent Policy Distance (MAPD), a tool designed to measure policy differences among agents. Additionally, they developed a Multi-Agent Dynamic Parameter Sharing (MADPS) algorithm based on MAPD, demonstrating its effectiveness in enhancing policy diversity and overall performance through extensive experiments. To addresses the challenge of cooperative MARL in scenarios with dynamic team compositions, Wang et al. [**?**] propose using mutual information as an augmented reward to encourage robustness in agent policies across different team configurations. They develop a multi-agent policy iteration algorithm with a fixed marginal distribution and demonstrate its strong zero-shot generalization to dynamic team compositions in complex cooperative tasks. Progressive Mutual Information Collaboration (PMIC)[27] is a novel framework that leverages mutual information (MI) to guide collaboration among agents, thereby enhancing performance in mult-agent cooperative tasks [**?**]. The key innovation of is its dual MI objectives: maximizing MI associated with superior collaborative behaviors and minimizing MI linked to inferior ones, ensuring more effective learning and avoiding sub-optimal collaborations. Wai et al. [**?**] proposes a novel double averaging primal-dual optimization algorithm for MARL, specifically targeting decentralized applications like sensor

[25] SCRIMMAGE Web: `http://www.scrimmagesim.org/`.

[26] SCRIMMAGE project: `https://github.com/gtri/scrimmage`.

[27] PMIC code: `https://github.com/yeshenpy/PMIC`.

networks and swarm robotics. The MARL algorithm enables agents to collaboratively evaluate policies by incorporating neighboring gradient and local reward information, achieving fast finite-time convergence to the optimal solution in decentralized convex-concave saddle-point problems. To address the challenge of sparse rewards in MARL, Kang et al. [**?**] introduce the Dual Preferences-based Multi-Agent Reinforcement Learning (DPM) framework. DPM extends preference-based reinforcement learning (PbRL) by incorporating dual preference types-comparing both trajectories and individual agent contributions-thereby optimizing individual reward functions more effectively. DPM also leverages LLMs to gather preferences, mitigating issues associated with human-based preference collection. Experimental results in the StarCraft Multi-Agent Challenge (SMAC) [**?**] demonstrate that DPM significantly outperforms existing baselines, particularly in sparse reward settings.

Traditional methods like soft attention struggle with scalability and efficiency in LMAS due to the overwhelming number of agent interactions and large observation spaces. To address these challenges of large-scale multi-agent systems (LMAS) involving hundreds of agents, University of Chinese Academy of Sciences [**?**] introduces the Concentration Network (ConcNet), a novel reinforcement learning framework. ConcNet mimics human cognitive processes of concentration by prioritizing and aggregating observations based on motivational indices, such as expected survival time and state value. It allows the system to focus on the most relevant entities, enhancing decision-making efficiency in complex environments. In ConcNet, a novel concentration policy gradient architecture was further proposed, demonstrating its superior performance and scalability in large-scale multi-agent scenarios, such as decentralized collective assault simulations. This research represents a significant advancement in the field, providing a scalable solution for effective decision-making in large-scale multi-agent environments.

In conclusion, MARL-based intelligent applications have shown exceptional adaptability across diverse domains such as autonomous driving, UAV systems, disaster response, and collaborative robotics [**? ? ? ? ? ? ?**]. Key innovations, including communication-enhanced learning [**? ? ? ?**], adaptive policy optimization, and mutual information [**?**] frameworks, have significantly advanced the field. While challenges like sparse rewards and scalability remain, these advancements highlight MARL's potential to address dynamic and complex multi-agent environments effectively, paving the way for further impactful developments.

4.2. LLMs reasoning-based Intelligent Applications

To address diverse and complex challenges, a variety of frameworks leveraging LLMs have been developed, enabling advanced reasoning, collaboration, and task execution in multi-agent systems [**? ? ?**]. Below, we highlight notable contributions that demonstrate the application of LLMs in enhancing multi-agent decision-making and coordination.

Wu et al. [**?**] introduced AutoGen, an open-source framework designed to enable the development of next-generation LLM applications through multi-agent conversations. AutoGen allows for customizable agent interactions and the integration of LLMs, human inputs, and tools to collaboratively solve complex tasks. Xiao et al. [**?**] proposed Chain-of-Experts (CoE), a novel multi-agent framework designed to enhance reasoning in complex operations research (OR) problems using LLMs. Chen et al. [**?**] presented AgentVerse, a multi-agent framework designed to facilitate collaboration among autonomous agents, inspired by human group dynamics. AgentVerse dynamically adjusts the composition and roles of agents throughout the problem-solving process, enhancing their ability to tackle complex tasks across various domains, including text understanding, reasoning, coding, and embodied AI. The framework consists of four stages: Expert Recruitment, Collaborative Decision-Making, Action Execution, and Evaluation. Chen et al. [**?**] introduced AutoAgents, a framework capable of adaptively generating and coordinating multiple specialized agents based on different tasks, thereby constructing efficient multi-agent teams to accomplish complex tasks. Liu et al. [**?**] proposed the Dynamic LLM-Agent Network (DyLAN), a framework designed to enhance LLM-agent collaboration by enabling agents to interact dynamically based on task requirements, rather than within a static architecture. Xu et al. [**?**] proposed a novel multi-agent framework that combines LLMs with reinforcement learning to enhance strategic decision-making and communication in the Werewolf game[28], effectively overcoming intrinsic biases and achieving human-level performance. Wen et al. [**?**] introduce the Multi-Agent Transformer (MAT), a novel architecture that frames cooperative MARL as a sequence modeling problem. Experiments on StarCraftII [**? ? ?**], Multi-Agent MuJoCo (MAMuJoCo) [**?**], Dexterous Hands Manipulation [**? ?**], and Google Research Football [**?**] benchmarks demonstrate that it achieves superior performance and

[28] Werewolf game: `https://sites.google.com/view/strategic-language-agents/`.

data efficiency by leveraging modern sequence models in an on-policy learning framework. Wang et al. [**?**] introduced MLLM-Tool[29], a multimodal tool agent system that integrates open-source LLMs with multimodal encoders, enabling it to process visual and auditory inputs for selecting appropriate tools based on ambiguous multimodal instructions. Moreover, they introduced ToolMMBench, a novel benchmark with multi-modal inputs and multi-option solutions, demonstrating its effectiveness in addressing real-world multimodal multi-agent scenarios. Zhang et al. [**?**] introduce CoELA, a Cooperative Embodied Language Agent framework that leverages LLMs to enhance multi-agent cooperation in complex, decentralized environments. CoELA integrates LLMs with cognitive-inspired modules for perception, memory and execution, allowing agents to plan, communicate, and collaborate effectively on long-horizon tasks, outperforming traditional planning-based methods such as Multi-Agent Transformer(MAT) [**?**], and showing promising results in human-agent interaction simulation environments, Communicative Watch-And-Help (C-WAH) [**?**] and ThreeDWorld Multi-Agent Transport (TDW-MAT) [**?** **?**]. Gong et al. [**?**] from Team of Li.FeiFei. introduce MindAgent, a novel infrastructure for evaluating planning and coordination capabilities in gaming interactions, leveraging large foundation models (LFMs) to coordinate multi-agent system (MAS), collaborate with human players, and enable in-context learning. Their team also present "Cuisineworld"[30], a new gaming scenario and benchmark for assessing multi-agent collaboration efficiency. Despite LLMs' success in various collaborative tasks, they struggle with spatial and decentralized decision-making required for flocking. Li et al. [**?**] explored the challenges faced by LLMs in solving multi-agent flocking tasks, where agents strive to stay close, avoid collisions, and maintain a formation. Sun et al. [**?**] proposed Corex, a novel framework that enhances complex reasoning by leveraging multi-model collaboration. Inspired by human cognitive processes, Corex employs three collaborative paradigms-Discuss, Review, and Retrieve-where different LLMs act as autonomous agents to collectively solve complex tasks. Corex empowers LLM agents to "think outside the box" by facilitating collaborative group discussions, effectively mitigating the cognitive biases inherent in individual LLMs. This approach not only enhances performance but also improves cost-effectiveness and annotation efficiency, offering a significant advantage in complex tasks.

Next, we will provide a detailed introduction to some outstanding achievements in the application of LLMs for multi-agent collaborative task execution.

MetaGPT: Existing LLMs-based multi-agent systems often struggle with complex tasks due to logical inconsistencies and cumulative hallucinations, leading to biased results. Hong et al. [**?** **?**] from DeepWisdom[31] proposed MetaGPT[32], an innovative meta-programming framework designed to enhance the collaboration capabilities of LLMs-based multi-agent systems. MetaGPT integrates Standard Operating Procedures (SOPs) commonly used in human workflows, thereby constructing a more efficient and coherent multi-agent collaboration system. MetaGPT employs an assembly-line approach, breaking down complex tasks into multiple subtasks and assigning them to agents with specific domain expertise. These agents collaborate during task execution through clearly defined roles and structured communication interfaces, reducing the risk of information distortion and misunderstanding. In summary, MetaGPT offers a flexible and powerful platform for developing LLMs-based multi-agent systems. Its unique meta-programming framework and rigorous workflow design enable it to excel in handling complex tasks, greatly advancing the field of multi-agent collaboration research.

CoAct: Hou et al. [**?**] proposed CoAct[33], a multi-agent system based on LLMs designed for hierarchical collaboration tasks. The framework consists of six stages: task decomposition, subtask assignment and communication, subtask analysis and execution, feedback collection, progress evaluation, and replanning when necessary. The global planning agent plays a critical role in managing complex tasks. The local execution agent is responsible for executing specific subtasks. This hierarchical framework demonstrates strong adaptability and performance, particularly in complex real-world tasks requiring dynamic replanning and collaborative execution.

AutoGen: Microsoft [**?** **?**] introduced AutoGen[34], a flexible framework for creating and managing multiple autonomous agents to collaboratively complete complex tasks, particularly in programming, planning, and creative writing domains. AutoGen allows users to define distinct agent roles, including specialists, general assistants, and decision-makers, ensuring clear task division and effective coordination. Agents

[29] MLLM-Tool: `https://github.com/MLLM-Tool/MLLM-Tool`.

[30] Cuisineworld: `https://mindagent.github.io/`.

[31] DeepWisdom: `https://www.deepwisdom.ai/`.

[32] MetaGPT: `https://github.com/geekan/MetaGPT`.

[33] CoAct: `https://github.com/dxhou/CoAct`.

[34] AutoGen: `https://github.com/microsoft/autogen`.

interact in a structured conversational environment, exchanging messages to resolve tasks iteratively. AutoGen introduces feedback loops where agents analyze outputs, refine strategies, and optimize task completion autonomously. Notably, it supports integration with various LLMs, offering developers the flexibility to replace APIs without altering code significantly. In summary, AutoGen facilitates scalable, efficient, and robust multi-agent collaboration, demonstrating potential for applications ranging from enhanced ChatGPT systems to real-world industrial workflows.

XAgent: XAgent Team[35] [**?**] developed XAgent[36] is an open-source, LLMs-driven autonomous agent framework designed for solving complex tasks automatically and efficiently. As shown in Figure 20, it employs a dual-loop architecture: the outer loop for high-level task planning and coordination, and the inner loop for executing subtasks. The PlanAgent in the outer loop generates an initial plan by breaking a complex task into manageable subtasks, organizing them into a task queue. It iteratively monitors progress, optimizes plans based on feedback from the inner loop, and continues until all subtasks are completed. The inner loop utilizes ToolAgents, which employ various tools like file editors, Python notebooks, web browsers, and shell interfaces within a secure docker environment to execute subtasks. XAgent emphasizes autonomy, safety, and extensibility, allowing users to add new agents or tools to enhance functionality. Its GUI facilitates user interaction while supporting human collaboration, enabling real-time guidance and assistance for challenging tasks.

PlanAgent: PlanAgent[37], developed by Chinese Academy of Sciences and Li Auto [**?**], introduces a closed-loop motion planning framework for autonomous driving by leveraging multi-modal large language models (MLLMs). The system utilizes MLLM's multi-modal reasoning and commonsense understanding capabilities to generate hierarchical driving commands based on scene information. PlanAgent addresses key limitations of traditional rule-based and learning-based methods, including overfitting in long-tail scenarios and inefficiencies in scene representation. Its novel integration of MLLM-driven reasoning into autonomous driving planning establishes a new benchmark for safety and robustness in real-world applications.

LangGraph: LangChain Inc[38] [**? ?**] introduced

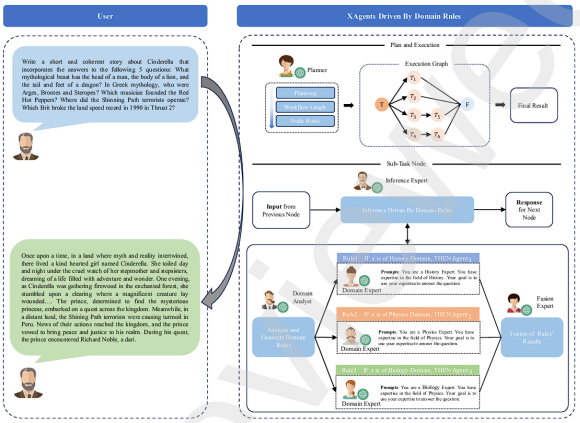


Figure 20: An structure overview of the XAgents framework, highlighting the Task Node as the starting point, the sequence of Sub-Task Nodes forming the Task Execution Graph (TEG), and the Fusion Node integrating outputs for the final result [**?**].

LangGraph[39], a library designed for building stateful, multi-actor applications with LLMs, enabling the creation of complex agent and multi-agent workflows. Inspired by frameworks like Pregel and Apache Beam, LangGraph provides fine-grained control over workflows and state management while offering advanced features like persistence and human-in-the-loop capabilities. LangGraph stands out for its support of iterative workflows with cycles and branching, which differentiates it from DAG-based frameworks. Each graph execution generates a state, dynamically updated by node outputs, enabling reliable and adaptive agent behavior. Its built-in persistence allows workflows to pause and resume at any point, facilitating error recovery and advanced human-agent interactions, including "time travel" to modify past actions. LangGraph integrates seamlessly with LangChain [**? ?**] but functions independently, offering flexibility for various applications, from dialogue agents and recommendation systems to natural language processing and game development. With streaming support, it processes outputs in real-time, making it suitable for tasks requiring immediate feedback. Its low-level architecture and customizable workflows make LangGraph a powerful tool for creating robust, scalable, and interactive LLMs-based systems.

CrewAI: CrewAI[40] [**? ?**] is an open-source framework designed to coordinate AI agents that specialize in role-playing and autonomous operations, enabling efficient collaboration to achieve complex goals. The

[35] XAgent Team: `https://blog.x-agent.net/`.
[36] XAgent: `https://github.com/OpenBMB/XAgent`.
[37] PlanAgent: `http://www.chinasem.cn/planagent`.
[38] LangChain Inc: `https://langchain.ac.cn/langgraph`.
[39] LangGraph: `https://www.langchain.com/langgraph`
[40] CrewAI: `https://github.com/crewAIInc/crewAI`.

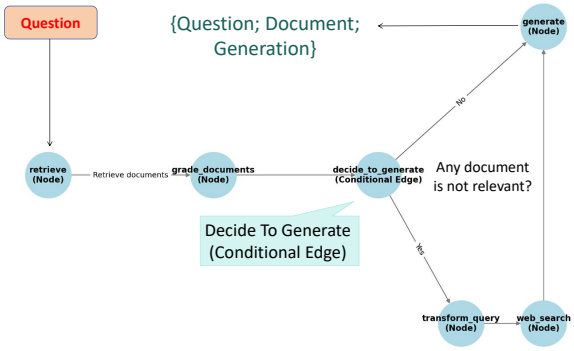


Figure 21: A LangGraph workflow representation demonstrating conditional branching and iterative loops for document retrieval, grading, query transformation, and web search before generating a final output [**? ?**].

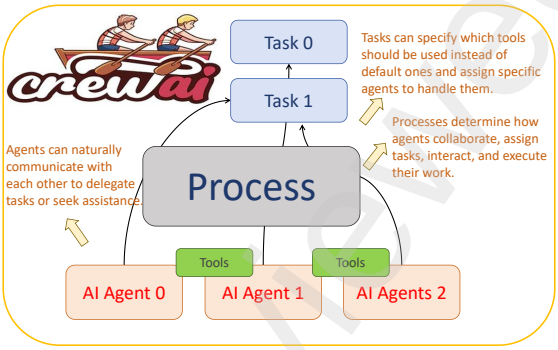


Figure 22: An overview of the processing workflow for the role-playing multi-agent framework, CrewAI [**? ?**].

framework's modular design allows users to create AI teams that operate like real-world teams, with agents assigned specific roles and tasks to ensure clear division of labor and shared objectives. As seen from 22, this framework operates in three primary stages: Agent Creation, where developers define roles with specific goals and tools; Task Management, enabling flexible task assignment and multi-view knowledge enhancement; and Execution and Collaboration, where agents interact in workflows to resolve tasks, with outputs parsed into reusable formats. CrewAI integrates seamlessly with the LangChain ecosystem, leveraging its tools and LLM capabilities, such as OpenAI and Google Gemini. The framework supports real-time decision-making and task adaptation, with future versions planned to include more advanced collaboration processes, such as consensus-driven workflows and autonomous decision-making. Its innovative features, such as role-based design, dynamic rule generation, and modular task workflows, position CrewAI as a robust and scalable framework for multi-agent collaboration across creative and industrial domains. Overall, CrewAI [41] offers a cutting-edge approach to multi-agent systems by integrating role-specific autonomy, flexible workflows, and advanced AI toolsets, making it a versatile framework for collaborative AI applications.

In summary, these frameworks and applications [**? ? ? ?**] highlight the rapid advancements in leveraging LLMs for multi-agent collaboration, reasoning, and task execution. Each system introduces unique innovations—ranging from dynamic agent coordination to enhanced reasoning and human-in-the-loop workflows—demonstrating their potential to tackle complex, real-world challenges across various domains [**? ? ? ?**]. These developments pave the way for more flexible, scalable, and efficient AI-driven solutions.

5. Challenges in MARL-based and LLMs-based approaches

The extension of single-agent decision-making into multi-agent cooperative contexts introduces several challenges, including developing effective training schemes for multiple agents learning and adapting simultaneously, managing increased computational complexity due to the more sophisticated and stochastic environments compared to single-agent settings, and addressing the foundational role of strategic interaction among agents. Additionally, ensuring the scalability of algorithms to handle larger observation and action spaces, facilitating coordination and cooperation among agents to achieve consistent goals, and dealing with non-stationary environments where agents' behaviors and strategies continuously evolve are also inevitable and critical challenges.

Applying multi-agent decision-making techniques to real-world problems, which often involve complex and dynamic interactions, further underscores the need for sophisticated and advanced approaches to effectively adapt these ever-increasing complexities. Multi-agent cooperative decision making significantly surpasses single-agent decision-making in terms of environmental stochasticity, complexity, the difficulty of strategy optimization, and so on. As shown in Figure 23, we present a tree diagram summarizing the existing challenges in MARL-based and LLMs-based multi-agent decision-making approaches.

[41] CrewAI Multi-Agent System platform: `https://www.deeplearning.ai/short-courses/multi-ai-agent-systems-with-crewai/`.

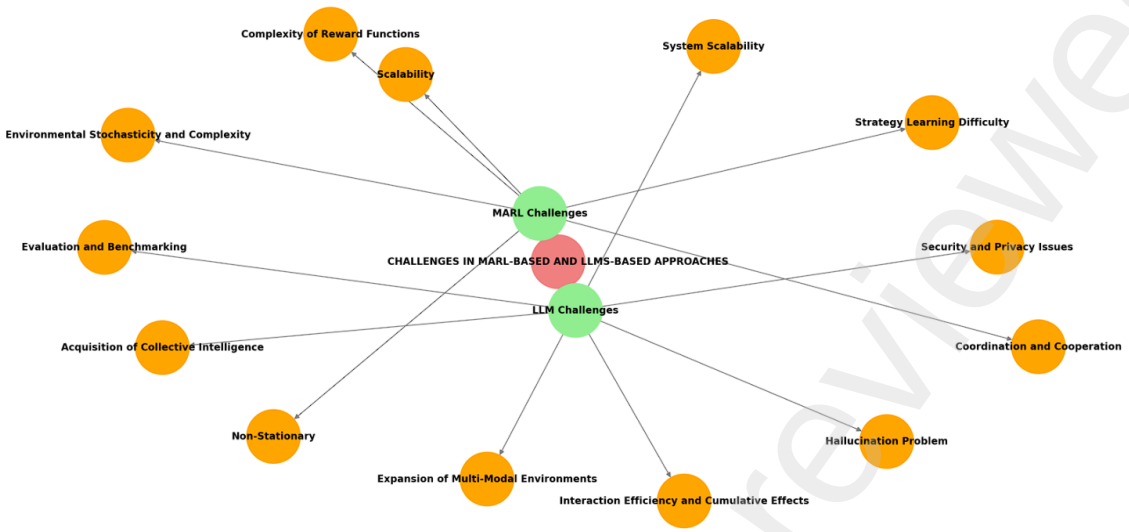


Figure 23: A tree diagram of the challenges in MARL and LLMs-based multi-agent decision-making approaches.

5.1. *Challenges in MARL-based multi-agent systems*

The advancement of MARL remains in its formative stages, with its potential for enabling effective multi-agent coordination and achieving scalability in dynamic environments yet to be fully unlocked [**? ? ? ? ?**]. Challenges such as environmental stochasticity, strategy learning difficulty, non-stationarity, scalability, and reward complexity have emerged as major bottlenecks. This section provides an in-depth analysis of these challenges, exploring the current state, technical limitations, and potential solutions in MARL-based multi-agent systems to enable more robust, efficient, and scalable decision-making frameworks.

1. **Environmental Stochasticity and Complexity:** [**? ?**] In MARL-based decision-making systems, environmental dynamics are influenced not only by external factors but also by the behaviors and decisions of individual agents. This complex interaction results in high levels of stochasticity and complexity in the environment, making prediction and modeling significantly more difficult. For example, in autonomous driving scenarios [**? ? ? ?**], the behavior of each vehicle affects the decisions of surrounding vehicles, thereby increasing the overall complexity of the system;
2. **Strategy Learning Difficulty:** [**? ? ?**] Strategy learning in MARL-based decision-making systems involves multidimensional challenges. Firstly, agents must consider the behaviors of other agents, and this interdependence increases the difficulty of strategy learning. Each agent not only has to optimize its own strategy but also predict and adapt to the strategy changes of others. Additionally, the vast joint action space of multiple agents makes it challenging for any single agent to learn effective joint strategies. The vast joint action space means that each agent needs to explore and learn within a larger decision space, which significantly increases the demands on computational resources and time;
3. **Complexity of Reward Functions:** [**?**] In MARL-based decision-making systems, reward functions become more complex [**? ?**]. The rewards received from the environment in multi-agent cooperative techniques are influenced not only by an individual agent's actions and the environment but also by the actions of other agents, which makes the stable policy learning and modeling process more difficult. In other words, an agent's reward depends not only on its own actions but also on the actions of other agents, making it challenging for the reinforcement learning-based multi-agent decision-making policies to converge. This intricate reward mechanism complicates the design and optimization of reward functions. Agents need to evaluate their behaviors' impact on the overall system through complex interactions to learn effective strategies;
4. **Coordination and Cooperation:** [**? ? ? ?**] Furthermore, in MARL-based decision-making systems, agents need to coordinate and cooperate to achieve common goals. This requires establishing effective communication and coordination mecha-

nisms among agents to ensure that their actions are globally consistent and complementary [**?**]. For example, in disaster rescue scenarios [**? ?**], multiple drones need to coordinate their actions to cover the maximum area and utilize resources most efficiently;

5. **Non-Stationary:** [**? ?**] The environment of MARL-based decision-making systems is non-stationary because each agent's behavior dynamically changes the state of the environment. This non-stationarity increases the difficulty of strategy learning, as agents must continually adapt to changes in the environment and the behaviors of other agents.
6. **Scalability:** [**? ? ?**] Addressing scalability in MARL demands innovative approaches to tackle the exponential growth in complexity as the number of agents increases. Techniques that leverage *knowledge reuse* [**? ? ?**], such as parameter sharing and transfer learning [**? ?**], have proven indispensable. When agents share structural similarities, *information sharing* can streamline the training process, enabling systems to scale more effectively. *Transfer learning*, in particular, allows agents to adapt knowledge from previous tasks to new, related ones, significantly accelerating learning in dynamic environments. Moreover, *curriculum learning* [**? ?**] plays a pivotal role in tackling the increased complexity of training multiple agents. It enables gradual learning by exposing agents to progressively more challenging tasks, thereby improving policy generalization and accelerating convergence. Robustness remains critical for scalability, as learned policies must withstand environmental disturbances. Techniques like *policy ensembles* and *adversarial training* [**? ?**] enhance resilience by fostering diversity and adaptability in policies. The DTDE paradigm addresses these issues but introduces new complexities [**? ?**], such as environmental instability. One promising solution is employing *Independent Deep Q-networks (IDQNs)* [**? ? ?**], which adapt traditional single-agent approaches to multi-agent contexts.

Overall, the interplay between robustness and scalability in MARL is a key area for future exploration. While existing techniques provide strong foundations, integrating methods like *meta-learning* could offer a way for agents to rapidly adapt to new tasks and environments. Additionally, leveraging recent advances in *graph neural networks* might enhance the scalability of MARL by modeling agent interactions more efficiently. These directions hold promise for tackling the dynamic and large-scale nature of multi-agent environments.

5.2. Challenges in LLMs reasoning-based multi-agent systems

The development of LLMs-based multi-agent systems is still in its early stages, and its advantages in multi-agent collaboration in cross-domain applications have not been fully realized. In this process, technical bottlenecks, design limitations, and imperfect evaluation methods have revealed numerous challenges. This section provides a comprehensive analysis of these challenges, exploring the current status, bottlenecks, and potential breakthrough directions of LLMs-based multi-agent systems in key areas such as multi-modal interaction, system scalability, hallucination control, evaluation, and privacy protection.

1. **Expansion of Multi-Modal Environments:** [**? ? ?**] Current LLMs-based multi-agent systems primarily focus on text processing and generation, particularly excelling in language-based interactions. However, applications in multi-modal environments remain insufficient. Multi-modal environments require agents to handle various inputs from images, audio, video, and physical sensors, while also generating multi-modal outputs, such as descriptions of visual scenes or simulations of physical actions. This cross-modal interaction not only demands stronger model processing capabilities but also requires efficient information fusion between agents. For example, in practical applications, one agent may need to extract visual features from an image and collaborate with other agents through language to accomplish complex tasks, posing new technical challenges. Future research should focus on building unified multi-modal frameworks that enable agents to efficiently understand and collaboratively process various types of data.
2. **Hallucination Problem:** [**? ? ? ?**] The hallucination in LLMs, which involves generating false or inaccurate information, becomes more complex in multi-agent environments. This issue may be triggered within a single agent and further propagated through multi-agent interactions, ultimately negatively impacting the decision-making of the entire system. Because the information flow in multi-agent systems is interconnected, any misjudgment at one node can trigger a chain reaction. This characteristic makes the hallucination problem not only confined to the behavior of individual agents but

also poses challenges to the stability of the entire system. Therefore, addressing this issue requires a dual approach: on one hand, improving model training methods to reduce the probability of hallucinations in individual agents; on the other hand, designing information verification mechanisms and propagation management strategies to minimize the spread of hallucinated information within the agent network.

3. **Acquisition of Collective Intelligence:** [**? ? ?**] Current LLMs-based multi-agent systems rely more on real-time feedback for learning rather than offline data, unlike traditional multi-agent systems [**? ?**]. This real-time learning approach imposes higher requirements on the interactive environment [**? ?**]. Since designing and maintaining a reliable real-time interactive environment is not easy, it limits the scalability of the system. Additionally, existing research mostly focuses on optimizing individual agents, neglecting the potential overall efficiency improvements that could arise from agent collaboration. For example, knowledge sharing and behavioral coordination among agents may create advantages of collective intelligence in certain complex tasks. Future research needs to explore how to fully leverage the potential of collective intelligence by optimizing multi-agent interaction strategies and collaboration mechanisms.
4. **System Scalability:** [**? ? ?**] As the number of agents in LLMs-based multi-agent systems increases, the demand for computational resources grows exponentially, posing challenges in resource-constrained environments. A single LLM agent already requires substantial computational power, and when the system scales to hundreds or thousands of agents, existing hardware and software architectures may not be able to support it. Furthermore, scaling the system introduces new complexities, such as how to efficiently allocate tasks, coordinate, and communicate among numerous agents. Studies have shown that the more agents there are, the more difficult it becomes to coordinate their operations, especially in reducing redundancy and conflicts. Therefore, future work needs to optimize resource utilization through the development of lightweight models and efficient communication protocols, while also exploring the scaling laws for agent expansion.
5. **Evaluation and Benchmarking:** [**? ?**] Current evaluation methods and benchmark tests for LLMs-based multi-agent systems are still incomplete. Most research focuses solely on the performance of individual agents in specific tasks, neglecting the overall system performance in complex scenarios. Evaluating group behavior is more challenging because the dynamics and complexity of multi-agent systems are difficult to measure with a single metric. Additionally, the lack of a unified testing framework and benchmark data is a major obstacle when comparing the capabilities of different LLMs-based multi-agent systems across domains. Future research needs to develop comprehensive evaluation standards and universal benchmark tests, especially in key fields such as scientific experiments, economic analysis, and urban planning, to provide a basis for system performance comparison and improvement.
6. **Interaction Efficiency and Cumulative Effects:** [**? ? ? ?**] The complexity of multi-agent systems leads to prominent issues of low interaction efficiency and cumulative effects. Low interaction efficiency is mainly reflected in the need for generative agents to frequently query models, making the system inefficient in large-scale applications. On the other hand, because the system state highly depends on the results of the previous round, when an error occurs in one round, it may accumulate and propagate to subsequent operations, ultimately degrading the system's overall performance. Future efforts should focus on designing more efficient communication protocols and intermediate result correction mechanisms to reduce interaction costs and the impact of cumulative errors.
7. **Security and Privacy Issues:** [**? ? ? ? ?**] Context sharing within multi-agent systems poses risks of introducing noise and privacy leaks. For example, sensitive information shared between agents (such as identities or locations) may be misused by untrusted nodes, thereby threatening the system's security. Addressing this issue requires a two-pronged approach: first, establishing clear organizational structures to restrict information access permissions; second, introducing more advanced trust management mechanisms, such as distributed trust systems based on consensus algorithms, to enhance the system's security and reliability.

In summary, LLMs-based multi-agent systems face challenges across multiple domains, including multimodal adaptation, scalability, evaluation methods, collective intelligence development, and privacy protection. These challenges not only reveal the current technological limitations but also provide ample space for future research. With advancements in technology and

the deepening of interdisciplinary studies, LLMs-based multi-agent systems are expected to achieve significant breakthroughs both theoretically and in applications.

6. Future Research Prospects and Emerging Trends

Multi-Agent Decision-Making Systems are entering a new era where LLMs are combined with MARL [**?**]. This combination can improve learning efficiency in complex dynamic environments. It also enables better multi-modal information processing, multi-task collaboration, and long-term planning [**? ? ? ? ?**]. In this section, we discuss future prospects and challenges of multi-agent decision-making system (MAS) research from theoretical, technical, application, and ethical perspectives.

6.1. Theoretical Development: From Traditional RL to LLMs-Enhanced MARL Framework

LLMs-enhanced MARL redefines collaboration in multi-agent systems by introducing natural language understanding and reasoning [**? ?**]. Traditional MARL requires agents to learn control strategies in dynamic environments with limited data [**? ? ? ?**]. However, this approach often faces challenges like low sample efficiency, difficult reward design, and poor generalization. LLMs, with their strong reasoning and knowledge representation capabilities, offer solutions [**? ?**]. For example, they can process multi-modal information such as natural language and vision [**? ? ? ?**], helping agents understand tasks and environments more effectively. This improves learning speed and generalization. Furthermore, LLMs can act as reasoning tools, providing additional context and knowledge to optimize long-term planning.

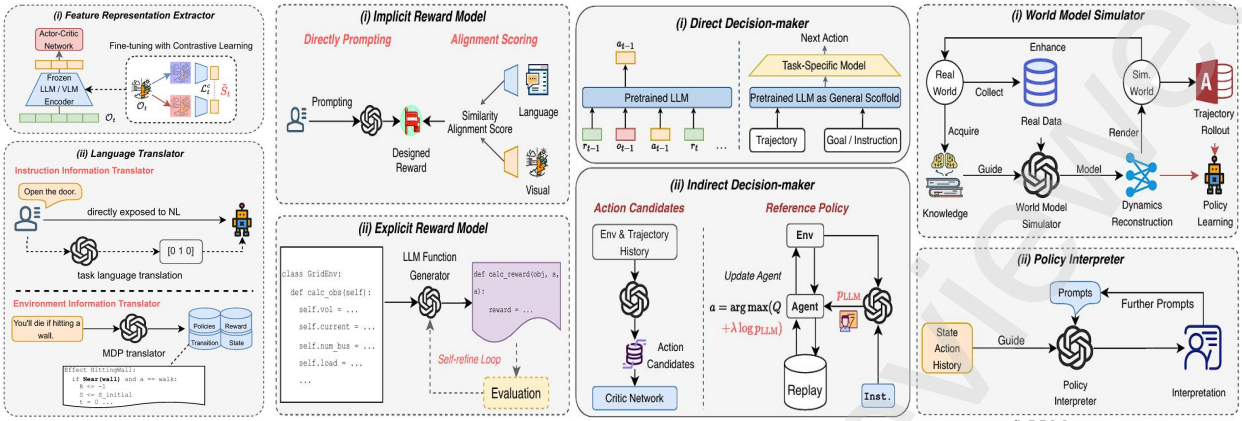
The LLMs-enhanced MARL framework is a groundbreaking integration of LLMs and MARL techniques, which includes roles such as *information processor, reward designer, decision-maker,* and *generator* [**?**]. Figure 24 presents a flowchart illustrating the structure of the LLMs-enhanced MARL framework, highlighting its four key roles. These roles work together to streamline task complexity and improve learning. For instance, LLMs can translate unstructured task descriptions into formal task semantics, reducing learning difficulty. They can also design advanced reward functions to accelerate learning in sparse-reward environments. These roles collectively address the challenges of task complexity, data efficiency, and generalization in MARL [**? ? ?**], while streamlining processes like reward design and policy generation. As shown in Table 4, we summarize recent advancements in LLMs-enhanced MARL methods across these four roles into a comprehensive table for clarity and comparison.

Table 4: Summary of recent studies categorized by the four key roles of LLMs in MARL: Information Processor, Reward Designer, Decision-Maker, and Generator, highlighting their respective contributions and applications.

Method Types **LLM as ...**	**Researchers. / Works. / Refs.**
Information Processor	Poudel et al. (ReCoRe) [**?**], Choi et al. (ConPE) [**?**], Paischer et al. (HELM) [**?**] and (Semantic HELM) [**?**], Radford et al. (CLIP) [**?**], Oord et al. (CPC) [**?**], Michael et al. (CURL) [**?**], Schwarzer et al. (SPR) [**?**]
Reward Designer	Kwon et al. (LLMrewardRL) [**?**], Song et al. (Self-Refined LLM) [**?**], Wu et al. (Read & Reward) [**?**], Carta et al. (GLAM) [**?**], Chu et al. (Lafite-RL) [**?**], Kim et al. (ARP) [**?**], Yu et al. [**?**], Adeniji et al. (LAMP) [**?**], Madaan et al. (Self-Refine) [**?**], Ma et al. (Eureka) [**?**], Xie et al. (Text2Reward) [**?**]
Decision -Maker	Janner et al. (TT-Offline RL) [**?**], Shi et al. (LaMo) [**?**], Li et al. (LLM scaffold) [**?**], Mezghani et al. (text BabyAI) [**?**], Grigsby et al. (AMAGO) [**?**], Zitkovich et al. (RT-2) [**?**], Yao et al. (CALM) [**?**], Hu et al. (instructRL) [**?**], Zhou et al. (LLM4Teach) [**?**]
Generator	Chen et al. (TransDreamer) [**?**], Das et al. (S2E) [**?**], Lin et al. (Dynalang) [**?**] , Robine et al. (TWM) [**?**], Poudel et al. (LanGWM) [**?**], Lin et al. (HomeGrid) [**?**]

6.2. Technical Integration: From Multi-Modal to Multi-Task Optimization

Combining LLMs and MARL significantly improves the ability to handle multi-modal information, multi-task learning, and long-term task planning [**? ? ? ?**]. Traditional MARL often requires separate modules to process visual, textual, or other forms of data. In contrast, LLMs can unify this processing, enabling comprehensive environment understanding. For example, in a robot task involving voice commands and visual inputs, LLMs can process both types of data simultaneously and generate actions directly. Additionally, LLMs provide a distinct advantage in multi-task learning due to their pre-trained knowledge [**? ?**]. Through knowledge transfer, they help agents share experiences across different tasks, improving adaptability [**? ?**]. For long-term planning, LLMs can break down complex tasks



a) LLM as information processor. b) LLM as reward designer. c) LLM as decision-maker d) LLM as generator

Figure 24: Schematic diagram of the LLMs-enhanced MARL framework based on *Cao et al.* [**?**], showcasing its core roles: *information processor* (a), *reward designer* (b), *decision-maker* (c), and *generator* (d).

into subtasks, addressing challenges like the credit assignment problem. This capability is particularly useful in tasks requiring extended reasoning, such as construction tasks in Minecraft. In optimizing reinforcement learning's sample efficiency [**?** **?**], the generative capabilities of LLMs can provide agents with additional virtual samples through high-fidelity environment simulations [**?** **?**]. This not only reduces the cost of real-world learning but also offers high-quality trajectories that serve as valuable references for policy optimization. Furthermore, in sparse reward environments, LLMs can accelerate policy learning by automatically designing reward signals.

6.3. Application Expansion: Driving Intelligent Collaboration in Complex Scenarios

The potential of LLMs-enhanced MARL in practical applications is enormous, especially in scenarios that require complex collaboration and real-time decision-making [**?** **?** **?** **?**]. For example, in the field of autonomous driving [**?** **?** **?**], the integration of LLMs with MARL can simultaneously process sensor data and natural language information (such as traffic regulations, passenger instructions, etc. [**?**]), thereby enhancing the safety and accuracy of decision-making [**?** **?**]. In the field of collaborative robots, LLMs can help multiple robots form a more intuitive communication mechanism, achieving highly complex task division and dynamic adjustment. In addition, in multi-objective optimization tasks such as smart grids and intelligent healthcare, LLMs can provide domain knowledge and optimization suggestions to assist reinforcement learning, design more practical reward functions, and thus improve the overall efficiency of the system. In dynamic and complex environments such as disaster relief [**?**], LLMs can dynamically allocate roles and responsibilities according to task requirements, helping multi-agent systems quickly adapt to changing environments and highly complex task divisions [**?** **?** **?**]. This capability provides a solid technical support for a wide range of applications.

6.4. Human Society Coordination: Balancing Technology and Ethics

The integration of LLMs into MARL opens new avenues for advancing multi-agent systems, while also highlighting exciting research directions in improving technical efficiency and addressing ethical considerations. For instance, enhancing the robustness of LLMs in unfamiliar environments offers the opportunity to develop strategies for minimizing biases and hallucinations, thereby improving decision accuracy. Furthermore, the computational complexity and resource demands of LLMs present a chance to innovate in optimizing inference efficiency and scalability. This is especially relevant in dynamic multi-agent environments where real-time responsiveness is critical.

From an ethical perspective, the incorporation of LLMs calls for advancements in ensuring data privacy, safeguarding against adversarial attacks, and establishing clear accountability frameworks for AI-driven decisions. Sensitive domains such as healthcare and disaster response could particularly benefit from focused research on protecting sensitive information and enhancing system resilience. Additionally, improving the transparency and explainability of LLMs-driven decisions is another promising area for exploration, as it

would increase trust and user confidence in multi-agent systems.

By addressing these areas, future research can maximize the potential of LLMs-enhanced MARL systems, ensuring they are both technically effective and ethically sound in diverse, real-world applications.

Overall, the combination of LLMs and MARL brings new momentum to research and applications in multi-agent systems. By enhancing collaboration through natural language understanding and leveraging large-scale knowledge, these systems can achieve greater efficiency and robustness in complex scenarios. However, fully unlocking their potential requires further exploration in theoretical methods, technological development, and ethical practices. With systematic advancements in these areas, LLMs-enhanced MARL can become the foundation for next-generation intelligent decision-making systems, transforming fields like autonomous driving, collaborative robotics, and healthcare, while shaping the future of AI research.

7. Conclusion

Multi-agent cooperative decision-making has demonstrated remarkable potential in addressing complex tasks through intelligent collaboration and adaptability. In this survey, we systematically review the evolution of multi-agent systems, highlighting the shift from traditional methods, such as rule-based and game-theory approaches, to advanced paradigms like MARL and LLMs. We differentiate these methods by examining their unique capabilities, challenges, and applications in diverse environments, paying particular attention to dynamic and uncertain settings. In addition, we explore the critical role of simulation environments as a bridge between theoretical advancements and real-world implementation, emphasizing their influence on agent interaction, learning, and decision-making. Practical applications of multi-agent systems in domains such as autonomous driving, disaster response, and robotics further underscore their transformative potential. By summarizing advanced multi-agent decision-making methodologies, datasets, benchmarks, and future research directions, this survey aims to provide a comprehensive resource for researchers and practitioners. We hope it inspires future studies to address existing challenges, such as improving inter-agent communication and adaptability, while leveraging the innovative potential of DRL and LLMs-based approaches to advance multi-agent cooperative decision-making.

Acknowledgment

The corresponding authors of this survey are B. Zhao and G.Yang from Xi'an Jiaotong University and Imperial College London. Guang Yang was supported in part by the ERC IMI (101005122), the H2020 (952172), the MRC (MC/PC/21013), the Royal Society (IEC\NSFC\211235), the NVIDIA Academic Hardware Grant Program, the SABER project supported by Boehringer Ingelheim Ltd, NIHR Imperial Biomedical Research Centre (RDA01), Wellcome Leap Dynamic Resilience, UKRI guarantee funding for Horizon Europe MSCA Postdoctoral Fellowships (EP/Z002206/1), and the UKRI Future Leaders Fellowship (MR/V023799/1). The authors would like to thank the editors and anonymous reviewers, who significantly enhanced the quality of the survey.

Declaration of Generative AI and AI-assisted Technologies in the Writing Process

During the preparation of this work, the authors utilized generative AI and AI-assisted technologies for proofreading and enhancing readability and language clarity in certain sections. The authors have carefully reviewed these contents to ensure accuracy and completeness, acknowledging that AI can generate authoritative-sounding output that may be incorrect, incomplete, or biased.

Appendix A. Technological Comparisons between Single-Agent and Multi-Agent (Under Reinforcement Learning)

Here, we discuss a series of technological comparisons of both DRL-based single-agent and MARL-based multi-agent research.

In solving these single-agent sequential decision-making problems, Markov Decision Processes (MDP) is a powerful mathematical modeling framework, especially in uncertain environments. Since the decision-making process of an agent can inherently be modeled as a sequence of decisions, the single-agent decision-making process can be formulated as an typical MDP, similar to a Markov chain.

In contrast to single-agent DRL systems, multi-agent systems under the MARL techniques involve multiple agents interacting within a shared environment. POMDP is a powerful mathematical modeling framework. It is an extension of the MDP framework that is particularly well-suited for modeling decision-making

in environments where the agent does not have full visibility of the entire state space. POMDPs extend MDPs to environments where the agent cannot fully observe the underlying state. Instead, the agent maintains a belief state, which is a probability distribution over the possible states.

Figure A.25 provides a comparative illustration of *Markov Decision Processes (MDP)* and *Partially Observable Markov Decision Processes (POMDP)*, which correspond to *single-agent* and *multi-agent* reinforcement learning paradigms, respectively.

The left side of Figure A.25 depicts an *MDP*, which models *single-agent decision-making* in a fully observable environment. The agent selects an action a from the *action space* A based on the current *state* s from the *state space* S. The environment transitions to a new state s' following the transition probability function $P(s' \mid s, a)$, and the agent receives a reward r. The objective is to optimize a policy π^* that maximizes the cumulative reward. Since the entire state is observable, the decision-making process is relatively straightforward.

On the right side, the *POMDP* framework extends MDPs to *multi-agent settings* where agents operate under *partial observability*. Each agent i receives only a *partial observation* o^i rather than the full state S. The agents take individual actions a^i, forming a joint action a_t, which influences state transitions and results in individual rewards r^i. The observations are generated according to the observation function $Z(o \mid s', a)$, requiring each agent to infer the missing state information and maintain a *belief state* for effective decision-making.

In summary, *MDPs* are well-suited for *single-agent systems*, where the environment is static and fully observable, allowing the agent to make optimal decisions based on complete knowledge of the state. On the other hand, *POMDPs* are crucial for *multi-agent reinforcement learning* scenarios, where multiple agents interact dynamically in an uncertain environment with limited information. This setting introduces challenges such as coordination, competition, and reward interdependencies, making decision-making significantly more complex.

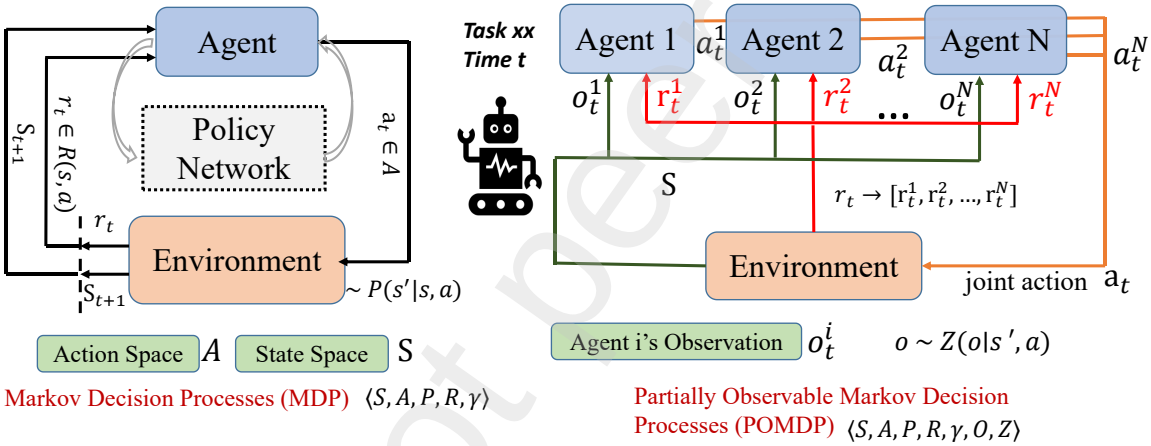


Figure A.25: The *Markov Decision Process* modeling for the single-agent reinforcement learning paradigm (left) and the *Partially Observable Markov Decision Process* modeling for the multi-agent reinforcement learning paradigm (right).