Tsinghua Science and Technology

Volume 26 | Issue 5 Article 4

2021

Multi-Agent Modeling and Simulation in the AI Age

Wenhui Fan
Department of Automation, Tsinghua University, Beijing 100084, China

Peiyu Chen
Department of Automation, Tsinghua University, Beijing 100084, China

Daiming Shi
Department of Automation, Tsinghua University, Beijing 100084, China

Xudong Guo
Department of Automation, Tsinghua University, Beijing 100084, China

Li Kou
Department of Automation, Tsinghua University, Beijing 100084, China

TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 04/15 pp608–624
DOI: 10.26599/TST.2021.9010005
Volume 26, Number 5, October 2021

Multi-Agent Modeling and Simulation in the AI Age

Wenhui Fan*, Peiyu Chen*, Daiming Shi, Xudong Guo, and Li Kou

Abstract: With the rapid development of artificial intelligence (AI) technology and its successful application in various fields, modeling and simulation technology, especially multi-agent modeling and simulation (MAMS), of complex systems has rapidly advanced. In this study, we first describe the concept, technical advantages, research steps, and research status of MAMS. Then we review the development status of the hybrid modeling and simulation combining multi-agent and system dynamics, the modeling and simulation of multi-agent reinforcement learning, and the modeling and simulation of large-scale multi-agent. Lastly, we introduce existing MAMS platforms and their comparative studies. This work summarizes the current research situation of MAMS, thus helping scholars understand the systematic technology development of MAMS in the AI era. It also paves the way for further research on MAMS technology.

Key words: artificial intelligence; system dynamics; reinforcement learning; large-scale multi-agent

1 Introduction

Agents emerged in distributed artificial intelligence in the 1970s. Since the 1990s, agents have become an essential frontier in computer research, especially in the field of artificial intelligence (AI). At the same time, research on agent-based modeling and simulation (ABMS) has been conducted in various fields (such as social science, economic system, biological science, ecological science, engineering technology, and simulation science), and the related influential research and application are fruitful.

Agents have various definitions in different fields and disciplines. Agents, which were initially used only as a system component, have been extended to become intelligent software, devices, robots, computers, or even human beings. An agent is an autonomous individual that can perceive the environment it is in, calculate and reason in accordance with the information obtained, communicate with other individuals, coordinate with each other, and cooperate with each other to complete a specific task and exert influence on the external environment. In this process, each agent not only has its own behavioral characteristics, such as autonomy, social ability, responsiveness, and initiative, but also has mental state characteristics, such as goals, knowledge, beliefs, responsibilities, and commitments according to their different roles and functions[1]. Agents also have other important characteristics, such as mobility, adaptability, and reasoning ability[2].

Although agents have various characteristics and possess the ability of decision-making and execution, the ability of a single agent remains limited. Such limitation motivates the emergence of multi-agent systems (MASs). Compared with single-agent systems, MASs not only have stronger performance but also help multiple agents solve certain problems independently through the information resources of the system. In addition, agents can cope with increasingly complex problems through mutual assistance[3], which encourages the emergence and development of multi-agent modeling and simulation (MAMS) technology.

• Wenhui Fan, Peiyu Chen, Daiming Shi, Xudong Guo, and Li Kou are with Department of Automation, Tsinghua University, Beijing 100084, China. E-mail: fanwenhui@tsinghua.edu.cn; cpy19@mails.tsinghua.edu.cn; shidm18@mails.tsinghua.edu.cn; gxd20@mails.tsinghua.edu.cn; koul13@mails.tsinghua.edu.cn.

∗ To whom correspondence should be addressed.

Manuscript received: 2021-01-07; accepted: 2021-01-19

2 Multi-Agent Modeling and Simulation

2.1 Concepts and ideas

Generally, traditional system modeling and simulation methods establish system models using the deductive reasoning method before conducting simulation experiments and analyses. This approach is a typical engineering simulation modeling method, a top-down modeling method, as well as an idea of reductionism. Local and foreign researchers have indicated that the existing traditional modeling methods based on reductionism cannot appropriately describe complex systems. In the modeling and simulation of complex systems, inductive reasoning is commonly used to build the formal abstract models of the systems, that is, using abstract representations of the system to gain an insight into the objective world and the natural phenomena; such approach is based on system theory and is a bottom-up modeling method. MAMS, which builds the model by using the basic elements of a complex system and their interactive relationship, organically links the microscopic behavior and the "emergent" phenomenon in the macroscope of complex systems. MAMS, a method of ontology, is one of the most effective simulation methods to solve the problem of complex systems.

Multi-agent simulation mainly consists of two aspects. On the macro aspect, the mechanism, protocol, and strategy of communication, coordination, and collaboration between agents, as well as the decomposition and assignment of tasks, are included. On the micro level, the dynamics, reasoning, and behavior of the agents are studied. The prior task of complex system simulation is to establish the system model and conduct experiments to study the emergence of the system, that is, studying when, where, and what kind of emergence the studied object appears. The concepts and ideas of MAMS mainly include the following six aspects[4]:

(1) Agents. Agents are autonomous computing entities and basic modeling and simulation units, which can sense the environment through sensors (physical or software) and act on the environment through effectors. A computing entity is a program that physically exists and operates on a computing device. Autonomy means that it can control its own behavior to a certain extent and can take certain actions without the interventions of human beings or other systems. To meet the design goals of a system, agents pursue the corresponding subgoals and perform the corresponding tasks. In addition, these subgoals and tasks can be complementary or conflicting.

(2) Model aggregation. Agents are a nested hierarchical concept. Basic agents, called meta-agents, have all the attributes of an agent. Model aggregation is needed to solve problems in complex large-scale systems; in some cases, hierarchical modeling that aggregates multi-granular, multi-scale, and multilevel models is also needed.

(3) Perception and action. Agents perceive information of the surrounding environment and other agents through internal perceptions and then react to the environment and other agents with the help of effectors, thereby realizing the functional behavior modeling of individual agents.

(4) Decentralized control. Each agent is relatively independent in MAMS, where no central control and coordination agents are needed for usual cases. Moreover, multi-agent control has been applied in various practical applications, such as robotics.

(5) External environment. In MAMS, the external environment not only provides several conditions for the existence of agents, but also acts as the perception and action object of agents.

(6) Interactions and associations. The interaction and association between agents are the cause of complex behavior in complex systems, as well as an important way to solve the dependence, conflict, and competition among agents.

2.2 Technological advantage

Compared with other complex system modeling and simulation technologies, MAMS has the following advantages[5]:

(1) MAMS can describe complex systems naturally. In many cases, MAMS is the most natural description of a complex system composed of numerous entities, and the concept of individuals in the system is consistent with agents.

(2) MAMS can capture the emergence phenomenon in complex systems. The "whole is greater than the sum of its parts" feature of emergence makes it difficult to predict. MAMS is a normative method for modeling this emergence phenomenon. By modeling the behavior of the micro individuals of the system, the purpose of describing the macro behavior of the system is achieved.

(3) MAMS has flexible organizational framework and evolution mechanism. Agent-based paradigms have a flexible potential computing mechanism for the

formation, maintenance, evolution, and disintegration of organizations and are especially suitable for solving the problem of flexible organization and scheduling in different complex systems.

(4) Agent autonomic solving and decision-making ability. The autonomic solution ability is the capacity of accepting the "stimulus" of the environment and other agents, reacting to the environment and other agents in accordance with the internal state and stimulus information, and modifying its own rules and states.

(5) Agent's ability to interact with users. Direct manipulation interfaces are used in traditional complex modeling systems; however, with the increase in task complexity, the manipulation process becomes increasingly complicated and thus can eventually affect the stability of the system. Nevertheless, the intelligence mechanism of agents creates technical conditions for human-computer interaction and cooperation, thus easily forming a harmoniously coexisting problem-solving system where the interaction and collaboration between users and agents are realized. In addition, the ability also guarantees system stability; thus, a single module error does not easily lead to system collapse.

(6) Suitable for distributed simulation. Agents describe the individual entities of a system and can be conveniently distributed to multiple computer nodes, thus providing the possibility for a distributed simulation of large-scale complex systems.

(7) Reusability of the model. Agents are abstraction entities with greater granularity than the object. By reusing a mature agent model, the efficiency of software development can be improved.

2.3 Research procedure

The MAMS procedure is mainly divided into the following five steps[6]:

(1) Specification of the aims of simulation. The complexity characteristics of the target system and the requirements of simulation are analyzed. First, the complexity of the target system is analyzed, and the objectives, requirements, and system boundaries of the simulation are specified. Then the specific characteristics of the entities in the system are analyzed, and the formal expression of the system is determined. Finally, the evaluation mechanism and method are defined, the data expression mode is determined, and the supporting functions of the simulation environment are summarized and formulated.

(2) Reasonable selection of abstract level. Hierarchy structures are the inherent structures of complex systems. Therefore, a multilevel abstract modeling method must be used to establish the abstract model of various object entities in a complex system, thus ensuring that the abstract level is reasonable and sufficient.

(3) Message flow analysis. The procedure includes classifying the message types, ensuring the flow patterns of various messages, and determining whether the simulation target needs further decomposition.

(4) Agent modeling. The procedure includes building analysis trees for complex systems through hierarchical decomposition and message flow analysis, and building a meta-agent model for each leaf node and an aggregation-agent model for each non-leaf node, and abstracting out different agents in accordance with the entities' different functions.

(5) Distribution of agents. The distribution of agents depends on the specific application requirements of the complex system simulation, the adopted simulation algorithms, and the hardware environment of the simulation.

2.4 Research on application status

MAMS has been applied in many fields, including social sciences, economics, artificial life, geographical and ecological processes, as well as the industrial and military fields. However, most studies are still in their infancy, that is, these studies are still "thought experiments" in the laboratory and have the nature of academic research. In reality, simulation analysis and control of complex systems still have a long way to go. Nevertheless, research on the ABMS of complex systems have been applied in practice.

2.4.1 Social field

The social sciences are one of the fields where MAMS is most widely applied. Its research focuses on the emerging behavior and self-organization of human systems; MAMS, which has been widely recognized by many social scientists, is the most suitable method to capture these phenomena[7]. "Humans" in social systems and agents in MAMS are essentially similar. "Humans" are abstracted as agents with autonomous decision-making, learning, memory, coordination, and organization abilities. Therefore, neural networks, evolutionary computation, or other learning skills are required for agents to describe the learning and adaptive ability of "humans". For example, neural networks were used to detect network's abnormal

behavior[8] and isolation forest was applied to fault diagnosis[9]. Research applications of MAMS in the social field include flow[10], such as traffic[11], evacuation in emergency circumstances, customer flow management, organization formation, and political interaction[12]. Casti from the Santa Fe Institute simulated the traffic and environmental conditions in Albuquerque[13], whereas Raney et al. studied the traffic problems in Switzerland[14]. In addition, Epstein and Axtell developed an agent model based simulation software called ResortScape, which can be used for parking lot management and decision-making[15]. Bilge developed an agent model based software called SIMSTORE for supermarket management and monitoring; the software was practically applied to the operation and management of several supermarkets in the UK[16].

2.4.2 Economic field

Economics is a field in which MAMS is widely applied. Researchers in Sandia National Laboratories in the United States developed an agent-based US economic simulation model called Aspen[17], which is a blend of the latest technology of evolution learning and parallel computing in the Sandia lab; this model is superior in many aspects compared with the traditional economic model. The variation of the influence of laws, rules, and policies is considered in a single, consistent economy simulation computing environment (e.g., building detailed models for monetary policy, tax law, and trade policy research; analyzing different economic sectors separately or together with other departments, thus facilitating the understanding of the entire economic process; accurately simulating the behavior of the basic decision-making sectors of the economy, such as residents, banks, companies, and policies). Aspen analyzed the effect of policies on micro and macro units by regarding micro units, such as individuals, residents, and enterprises, as simulated objects. Through the statistics, analysis, inference, and synthesis of characteristic variables, the influence of policy changes on micro individuals and the effects of policy implementation on macro and various levels can be observed. Sandia National Laboratory has already established a prototype model of the simple market economy (a simple simulation of the US economy) and a transition economy simulation model (a transition economy simulation), which is a detailed model[18].

In addition, the virtual stock market developed by a Bios team led by Arthur of the Santa Fe Institute has been successfully applied to the NASDAQ stock market simulation[19, 20]. The agent-based NASDAQ simulation model successfully combined the agent-based modeling idea with AI technologies, such as neural networks and reinforcement learning. This simulation method has further been used to study the US housing market[21]. Agents in the stock market interact by adopting different strategies from simple to complex. Through the interaction between agents, the dynamic behavior of the entire stock market can be expressed.

2.4.3 Military field

The military field is a new area of application for MAMS. Military confrontation and land warfare systems are complex adaptive systems (CASs)[22–25], which have been recognized by researchers. Therefore, MAMS can be used to study battlefield behavior, such as military confrontation[26]. The results of existing research show that MAMS, which has strong vitality, is more effective than the current combat model based on the Lanchester equation and is a good method for battlefield simulation.

The US Department of Defense (DOD) hopes to have real-time, omnidirectional access to information in future battlefields. For C4ISR to be useful, advanced real-time distributed modeling and simulation tools must be used, and complexity science can help develop C4ISR. As the MAMS is a complexity science methodology, it naturally became the DOD's advanced modeling and simulation methodology. The DOD's applications on ABMS include the irreducible semi-autonomous adaptive combat (ISAAC) developed by the US Marine Corps Combat Development Command (MCCDC), the enhanced ISAAC neural simulation toolkit (EINSTein) and SWarrior, the adaptive collection management environment (ACME) developed by the US Army Intelligence and Security Command (INSCOM), and the tactical sensor and ubiquitous network agent modeling initiative (TSUNAMI) co-developed by the Naval Warfare Development Command and Argonne National Laboratory's Center for Complex Adaptive Systems Simulation.

ISAAC[22, 23] was developed on the basis of the agent model. Through the simulation of war, questions, such as "to what extent do land warfare systems have the characteristic of self-organizing CAS", can be answered. The software was not designed to construct a system-level battlefield model but to serve as a simulation kit to explore interactions from different low-level rules

to high-level emergent behavior (for example, from individual warriors to a squad). ISAAC's long-term goal is to turn its subsequent product into a toolkit through which the emergent aggregation behavior on the battlefield can be explored. The agent in ISAAC has four characteristics of rule, task, situational awareness, and self-adaptability. Through the interaction of simple rules, the ISAAC system presents operational concepts, such as forward advance, frontline attack, local clustering, penetration, retreat, attack attitude, containment and containment, encircle maneuver, and guerrilla attack.

EINSTein[22, 23] is an enhanced version of ISAAC. The main improvements include a Windows-style Console User Interface (CUI) interface, an object-oriented C++ code, context-dependent and user-defined agent behavior, personalized script representation, online genetic algorithms, neural networks, reinforcement learning and pattern recognition toolkits, online data collection and multidimensional visualization toolkits, an online analysis toolbox, and a fitness coevolution legend display. At present, EINSTein studies two basic problems: command and control of topology and battle-related information.

MCCDC also developed SWarrior[24], which is based on Swarm, by combining certain characteristics of ISAAC. SWarrior aims to transform Swarm into a new analytical tool to gain an insight into future military confrontation based on agent-based simulation.

INSCOM and the Bios team of Santa Fe developed ACME[25], which helps commanders manage and acquire real-time battlefield information and obtain the enemy command post locations with the constantly changing battlefield maps.

An agent-based model was built by TSUNAMI[25] for the red, blue, and neutral forces, which have complex behavior and different properties, such as communication equipment, perception ability, mobility, memory ability, and fuel and battery energy. TSUNAMI simulates the battlefield space motion and interaction by describing the real terrain, and it can "clone" various sensors and use rule sets to simulate message flow and service protocol quality.

In addition, the Australian Defense Force Academy developed the reducible agent battlefield behaviour through life emulation (RABBLE)[27]. In contrast to ISAAC, RABBLE uses an MAS structure, adding up the learning mechanism and thus making the simulated group behavior conducive to decision-making. SWARM and Battle Model[26] developed by the Air Operations Division in Australia builds an agent-based model for pilots, fight managers, sensor managers, air combat defense commanders, and ground crew in air combat. In the military field, Heinze investigated the military combat concept by using the agent-based model[28], and Hill designed and implemented Tactical Simulation, an intelligent agent software model for air combat (especially over-the-horizon combat) simulation[29].

3 Hybrid Modeling and Simulation Based on Multi-Agent and System Dynamics

Hybrid modeling and simulation have received considerable attention over the past few years due to their flexibility and improved support tools. The construction of hybrid modeling and simulation aims to describe the complex behavior of the system[30]. A hybrid simulation method should comprehensively consider all types of interactions in the hybrid model to simulate various behaviors in the complex system. MAMS and system dynamics modeling and simulation (SDMS) are two important simulation methods[31].

SDMS is a top-down feedback method proposed by Forrester[32]. The essence of this method is a high-order, multi-loop, and nonlinear feedback structure. SDMS is a method for visualizing, analyzing, and understanding complex dynamic feedback[33]. The advantage of SDMS lies in its ability to consider nonlinear feedback and the time-delay characteristics of the dynamics[34]. SDMS has been widely used to solve various problems in social, industrial, environmental, and project management systems[35]. It uses feedback loops, stocks, and streams to simulate the dynamics of complex systems over time. In SDMS, the behavior of the system consists of two basic types of feedback loops, including a negative feedback loop for system regulation and a positive feedback loop with the function of strengthening input[36].

MAMS is a computer simulation method describing the behavior of complex systems. It adopts the bottom-up information feedback method to describe the interaction between individuals, identifies each entity as heterogeneous rather than identical, and allows individuals to evolve and adapt dynamically[37]. In MAMS, the complexity of the system is reflected in the interaction between different agents[32, 38]. The purpose of simulation is to track the interactions between agents in artificial environments and understand the emergence of global patterns[39]. Agents in MAMS have certain properties and interact by defining appropriate rules in

a given environment[40]. Agents act or produce output on the basis of their interactions, environments, and the rules they follow.

The hybrid model combining MAMS and SDMS emerged in the late 1990s. Scholl[41] conducted the first studies on the applicable scope of SDMS and MAMS.

In this study, the author also studied the advantages and disadvantages of each method and summarized the possibility of multimethod modeling based on SDMS and MAMS[41]. Pourdehnad et al.[42] deepened the above work by conceptually comparing these two methods. The authors discussed the potential synergies between the two paradigms to solve problems in the teaching and decision-making processes[42]. Similarly, Stemate et al.[43] compared these modeling approaches and listed a range of possible cross-applications, which provided a reliable basis for researchers to conduct further research. Lorenz and Jost[25] believed that the combination of SDMS and MAMS makes the simulation model closer to reality.

Schieritz et al.[44, 45] mainly worked on comparing SDMS and MAMS in the field of operations research. They identified the unique features of each approach and presented a table of major differences. In Ref. [46], a method that combines SDMS and MAMS was proposed to solve supply chain management problems. The results suggested that the use of combined methods does not produce the same results as using SDMS alone. To explore the reasons for these differences, the authors pointed out that further research is needed[46].

Ramandad and Sterman[22] compared the results of simulating the dynamic process of infectious disease transmission using MAMS and SDMS, transformed MAMS into SDMS, and examined the effects of individual heterogeneity and different network topologies. They concluded that SDMS produces a single trajectory for each parameter set, whereas random MAMS produces a distribution of the results.

Scholl[41] provided an overview of the general modeling principles of SDMS and MAMS, described their areas of applicability, discussed their relative strengths and weaknesses, and attempted to identify areas where the two modeling and simulation methods complement each other and overlap. MAMS is inductive because the resulting emergent behavior of such agents is the basic unit of analysis. By contrast, SDMS is deductive because it is described by its feedback structure at an aggregate level. Both techniques aim to discover leverage points in complex aggregate systems; modelers of agent-based models seek them in rules and agents, whereas SDMS does so in the feedback structure of a system.

Similarly, Lorenz[24] proposed three aspects to consider when choosing between SDMS and MAMS: structure, behavior, and appearance. Structure is related to how the model is built. The structure of the SDMS model is static, whereas that of MAMS is dynamic. In SDMS, all elements of emulation are preset, whereas in MAMS, agents can be created or destroyed, and interaction rules are defined during simulation. Another aspect is the difference in the behavior center generators of the model. For SDMS, the behavior generators are feedback and accumulation, whereas for MAMS, the generators are the interactions between system elements. Both methods involve feedback. However, MAMS has feedback at multiple modeling levels. In addition, the ability of emergence in the simulation varies in the two methods. In this study, the author pointed out that MAMS can simulate emergence, whereas the single-layer structure of SDMS cannot[24].

Despite the increasing amount of research on hybrid simulation methods, research on the properties and types of hybrid models is limited[27], and the hybrid simulation framework should be able to consider all types of interactions within the hybrid model[30]. To address this problem, Swinerd and McNaught[27] proposed three hybrid SDMS-MAMS simulation frameworks of integration, interface, and sequence.

In recent years, the comparative research results of SDMS and MAMS show that SDMS and MAMS have their advantages and disadvantages in complex system simulation. SDMS focuses on the dynamic behavior of the system and the analysis of the interaction and accumulation effect between different elements while ignoring the spatial factors. MAMS focuses on the interaction in space and ignores the feedback effect of macro-socioeconomic factors on agents[26]. To capture the heterogeneity and homogeneity of complex socioeconomic system models in dynamic simulation environments, two types of hybrid models combining MAMS and SDMS were proposed. In the first type, MAMS is used to create aggregate constructions, and SDMS is used to aggregate constructions and thus generate dynamic behavior. In the second type, SDMS is used to deduce the characteristics (states) of agents at the micro level, whereas MAMS deals with the interaction process of these agents under different rules[47]. In the past two decades, an increasing number of scholars

have begun to explore the combination of SDMS and MAMS[22, 46]. The hybrid approach has already been applied to solve problems in different fields, such as transportation[48], health-related research[24, 25, 43, 48], psychology[49], work environment[50], and ecological modeling[51–54].

4 Multi-Agent Reinforcement Learning Modeling and Simulation

Reinforcement learning is a mode of machine learning that can actively perceive the environment through different behaviors or actions, evaluate behavior simultaneously, and apply the evaluation to adjust the follow-up behavior; that is, it is a learning technology that can map different environmental states to behaviors[55]. The main purpose of reinforcement learning is to select the optimal behavior of an agent to complete the target. It is widely used in robot control systems[56, 57], intelligent decision[58], nonlinear optimal control[59–61], and other fields[62]. Complex practical problems cannot be reflected due to the lack of decision-making and environmental awareness ability of a single agent; thus, the concept of multi-agent, which is a collection of multiple intelligence called an MAS, was proposed in the late 20th century. MAS, a frontier domain of distributed AI, is mainly used to study the coordination[63], communication, and conflict between groups of agents[64].

4.1 Single-agent reinforcement learning

Studies on reinforcement learning mainly began in the 1850s[55], and the idea of trial-and-error learning was established soon after. The representative of modern reinforcement learning, i.e., temporal difference learning, began to be applied to the Markov decision process, and its efficiency was much higher than that of the traditional reinforcement learning algorithm[65]. Subsequently, the researchers integrated the time-series difference method and other optimization methods and proposed the Q-learning algorithm[66], which is acknowledged as the basis of most of the current deep reinforcement learning algorithms.

After the proposal of deep learning, traditional reinforcement learning actively explored the possibility of combining with deep learning. In the beginning, the combination of deep learning and reinforcement learning methods is mostly value-based, that is, it uses a deep neural network approximation function or an action-value function to extend the problems that reinforcement learning can solve from limited state space to continuous state space. The value-based deep reinforcement learning algorithm can fit the continuous state space; however, the value-based method cannot solve the problem of continuous action space. Hence, the stochastic strategy gradient method and the deterministic strategy gradient algorithm, which are based on the actor-critic framework policy gradient algorithm, were gradually developed[67].

4.2 Multi-agent reinforcement learning

In 2000, machine learning researchers took MAS as an important application background of AI[68] and proposed the concept of multi-agent learning in 2006[69]. In 2013, the deep Q-network combined reinforcement learning with deep learning for the first time[70] and thus remarkably improved the performance and stability of reinforcement learning algorithms, attracted several researchers to take reinforcement learning as the learning method of MAS, and initiated the study of the multi-agent reinforcement learning[71, 72]. In the game theory field, multi-agent reinforcement learning is classified into complete cooperation, perfect competition, and mixed type in accordance with the task types to be solved[73]; it can also be classified into four types according to different research contents and methods: analysis of emergence behavior, learning how to communicate, cooperate, and model the adversary[74].

The first is the analysis of emergence behavior. This part of research mainly applies the existing single-agent deep reinforcement learning algorithm to MAS, analyzes the performance and behavior of multi-agent, and observes the evolution of multi-agent. For example, the competitive behavior of deep reinforcement learning agents in table tennis tasks was studied[75]. In Ref. [76], the performance and property of typical reinforcement learning algorithms in counterattack tasks were explored.

The second is to learn how to communicate. In this part of the study, the communication between agents can help agents in completing tasks. This method mainly learns when and how agents communicate with each other. For example, Facebook proposed a CommNet network structure that aggregates the communication between agents through a summation operation[77]. In Ref. [78], the communication relationship and communication information between agents were intelligently selected on the basis of the attention mechanism.

The third part is to learn how to cooperate. This part

ignores how agents communicate with each other and focuses on the performance of their cooperation. Multi-agent deep deterministic policy gradient, an extension algorithm based on deep deterministic policy gradient, proposed a learning method of centralized training-decentralized execution, which has become the paradigm for most cooperative methods[79]. By contrast, the researchers proposed a value decomposition network, which decomposes the global value function into the accumulation of the multi-agent's local value functions and further simplifies the center's evaluation network[80].

The fourth part, which is how to model the adversary, models the adversary's strategy. The modeling of other agents has applications and benefits in cooperative and competitive tasks and can also help explain the behavior of agents[81]. For example, performance can be improved in the hiding information task by modeling others[82] and in the study of the overfitting problem in the strategy modeling of other agents[83].

4.3 Status and outlook

With the popularity of robots, manufacturing, logistics, disaster relief, unmanned vehicles, and other fields have become the typical application scenarios of MASs. With economic development, MASs with high efficiency and intelligence are urged. As the frontier of solving MAS problems in the field of AI, multi-agent reinforcement learning provides a feasible method for developing intelligent algorithms in different environments and tasks. However, the current multi-agent reinforcement learning algorithm still suffers from the following problems:

(1) The convergence and stability of multi-agent reinforcement learning have yet to be proven systematically. Reinforcement learning optimizes strategies by the agents' self-exploration, which requires agents to balance random exploration and autonomous decision-making. The convergence and stability of various single-agent reinforcement learning algorithms have been widely studied and demonstrated; however, the convergence of multi-agent reinforcement learning algorithms depends not only on the environment but also on the performance of other agents, thus making it difficult to prove the convergence of the optimal strategy and the stability after convergence of multi-agent reinforcement learning[73].

(2) The state space of multi-agent reinforcement learning is huge, and the training time is long. To ensure the convergence of the strategy optimization process, the single-agent reinforcement learning algorithm needs to explore massive possibilities of the "state-action" space. The joint "state-action" space of a multi-agent increases exponentially with the number of agents in the system, leading to an exponential increase in training time, limiting the extensibility of the number of agents in multi-agent reinforcement learning.

(3) The expansibility and knowledge transfer ability of multi-agent reinforcement learning are poor. On one hand, the current multi-agent reinforcement learning can only be designed for specific tasks and has trouble expanding and transferring strategies for different tasks so that each task needs to be trained and learned again. On the other hand, each retraining requires training for multiple agents. How to use experience between agents[84] and how to store and transfer knowledge[85] are still hot topics in this field.

Although the theory and practice of multi-agent reinforcement learning still face great challenges, multi-agent reinforcement learning is still regarded as an essential method to produce collective intelligence[86]. This study argues that the contributions of multi-agent reinforcement learning have two aspects. First, multi-agent reinforcement learning can bring robot teams with cooperative consciousness and human-like intelligence. Multi-agent reinforcement learning can train AI to complete complex multi-agent tasks of Dota2[87] and StarCraft II[88], and its performance can reach the level of e-sports athletes. Second, research on the training process of multi-agent reinforcement learning can help explain the emergence and evolution of biological group behavior and provide quantitative model support for the development of anthropology and sociology. The multi-agent reinforcement learning method combines the actual characteristics of distributed decision between biological population and robot population and the advantages of independent learning optimization strategy in AI. Moreover, it is becoming an interdisciplinary science of multi-agent simulation, swarm game theory, reinforcement learning, and other fields.

5 Large-Scale MAMS

In many disciplines, such as physics, social sciences, electronic communication, ecology, and military research, the number of agents involved in MAMS is always large, and the millionth magnitude of scale makes it difficult for ordinary computers to provide enough computing power[89]. Similar computing problems

occur when the agent-based simulation algorithm is complex[90]. These problems can be collectively regarded as large-scale MAMS problems. Numerous approaches have emerged in various fields trying to solve this problem. Among them are two most effective and common solutions: One is to restructure the model at the software level by super individual or other methods; another is to speed up large-scale computing by distributed parallel computing[91, 92] or using new computation tools, such as the Quantum tool[93].

Numerous software reorganizations have been successful. For example, Blythe et al. successfully simulated approximately 3 million agents and generated a total of 30 million operations by using stationary probability models, embedding link prediction, and introducing Bayesian models, thus enabling large-scale multi-agent simulation to model the evolution of GitHub (a large collaborative software development ecosystem)[94]. Campagne et al. proposed the use of morphology to represent and control the state of large organizations composed of large-scale agents that represented the state of the system as the shape in abstract geometric space[95]. In the era of AI, neural networks can also be used in the abstraction and simplification of models. Zhou et al. integrated the emerging mean field game theory with reinforcement learning technology on the basis of self-organizing neural networks, which effectively break the "curse of dimension" of large-scale MAMS and greatly reduce computational complexity[96].

Progress has been achieved in distributed parallel computing in recent years. Fachada et al. compared different parallel computing strategies, such as equal, equal with repeatability, equal with row synchronization, and on-demand, and pointed out that different parallelization strategies have specific trade-offs in terms of performance and simulation repeatability[97]. Predator-prey for high-performance computing, an effective reference model that can compare different parallelization strategies from performance and statistical accuracy, was also proposed[98]. Many scholars have begun to explore the use of GPU and other hardwares. For example, the GPU-based mobility simulator GEMSim designed by Saprykin et al. accelerates the process of large-scale multi-agent simulation, and its simulation cycle is more than 12 times faster than the previous method MATSim[99]. P-HASE designed by Marurngsith and Mongkolsin can generate the GPU code OpenCL automatically without any GPU programming language knowledge to optimize large-scale multi-agent simulation. By being evaluated through the experiment on two GPU platforms, NVIDIA GeForce 240m LE and AMD Radeon HD6650M, the large-scale multi-agent model in support of the newly generated GPU can be 14 times faster than its multicore CPU version[100]. The REPAST HPC framework raised by Collier and North uses C++ and MPI for large-scale distributed computing, which is accelerated by multi-process parallel computing[101].

The application of the above two methods enables us to extend the multi-agent simulation modeling method to large-scale problems and calculate the simulation results in a short time without losing too much computational accuracy, which has important application and promotion value. For example, in the transportation field, Klügl and Rindsfüer simulated a scene of more than 40 000 agents passing through the Bourne railway station within 1.5 virtual hours, not only ensuring that the agents are moving without collision between two predefined positions but also planning and replanning the way the agents pass through the railway station flexibly[102]. Zhang et al. simulated the traffic flow situation of Shanghai, China, with 200 000 agents on a network with 50 000 links[103]. In the field of medical imaging, Haroun et al. introduced large-scale multi-agents to merge the local image processing results after local processing on brain magnetic resonance imaging and improve the quality of the image after segmentation[104]. Zhang and Verbraeck studied the control strategy for the mass spread of infectious diseases on the basis of the large-scale ABMS method on PC; the population size, namely, the number of agents, reached 19.6 million[105]. Suzumura et al.[106] proposed agent-based complex cellular automata architecture, which realizes the traffic flow simulation of one billion agents on a supercomputer. The Los Amos National Laboratory developed an agent-based model software package called Traffic Analysis Simulation System[107]. The software is currently used to simulate the traffic conditions in Portland, including 120 000 traffic links and 1.5 million agents.

6 Multi-Agent Modeling and Simulation Platform

With the constant advancement of agent research, various agent simulation platforms have been developed internationally. Among them are typical platforms

with strong pertinence to a certain field. Examples include OpEMCSS, which can perform complex traffic system simulation; MaDKit, which simulates complex supply chains; and James providing multi-negotiation simulation between agents. However, these platforms have poor generality, that is, their strong simulation capability only precipitates in specific fields. As for the general agent simulation platform, typical platforms, including Java agent development framework (JADE)[108], NetLogo[109], Swarm[110], REPAST[111], MASON[112], AnyLogic[113], and JCass[114], have been widely used.

6.1 JADE

JADE is a software platform that provides basic, intermediate layer functions[108]. It follows the rules of the Foundation for Intelligent Physical Agents (FIPA) and can develop standard agent programs to complete the interaction and simulation between multi-agents. FIPA, established in 1996, aims to standardize agent technology and improve the availability of agents. The origin of JADE was to validate the FIPA specification set, which was launched by Telecom Italia in 1998 and has since evolved into JADE. The research of the platform focuses on the simplicity and usability of agent-based software development. The platform opened its source code in 2000, becoming a free, open-source platform. The biggest advantage of JADE is that it uses Java language for agent abstract programming, making JADE flexible, portable, and maintainable.

JADE can complete all agent basic services, such as life cycle management, mobility, white and yellow page services, information transmission, and security management. In JADE, agents communicate with each other in an asynchronous mode. Each agent has its own unique ID, and its own message queue for sending and receiving messages, and these features do not rely on location. JADE does not provide a visual window for model simulation and thus needs further development. In recent years, JADE has been widely applied in various agent-based simulation developments.

The agent of JADE exists in the container, and JADE is composed of many agent containers distributed on the network. The container is a Java process, which can run multiple agents and provide the services needed by JADE to run, manage, and execute agents. A main container must be started first to provide access to JADE, and other containers can only be added to the main container if they are registered with it.

6.2 NetLogo

NetLogo is a programmable modeling environment for simulating natural and social phenomena[109]. Uri Wilensky introduced the tool in 1999 (http://ccl.northwestern.edu/netlogo/), and the tool has been further developed at the Center of Connected Learning and Computer-Based Modeling, Northwestern University. NetLogo, equipped with various types of agents, such as turtle, patch, and link, is especially suitable for modeling the evolution of complex systems. Modelers can send instructions to thousands of independently operated agents and thus help in gaining insight into the connection between individual behavior at the micro level and the macroscopic model, which emerges through many individual interactions.

The bottom layer of NetLogo is implemented in the Java programming language and can run on all major platforms (Mac, Windows, Linux, etc.) or in browsers, such as Java applets. NetLogo has detailed documentation and teaching materials. It also comes with a model library that contains many already written simulation models, which cover many areas of the natural and social sciences, including biology and medicine, physics and chemistry, mathematics and computer science, economics, social psychology, and so on.

NetLogo, a programming development platform inherited from the Logo language, can control thousands of individuals in modeling and thus compensate for the deficiency wherein the Logo language can only control a single individual. Therefore, NetLogo modeling can well simulate the behavior of micro individuals and the emergence of macro patterns, as well as the relationship between them. NetLogo is a programming language and modeling platform for natural and social phenomenon simulation, especially for complex systems that develop over time.

6.3 Swarm

Swarm is a standard multi-agent software toolset for computer simulation developed by the Santa Fe Institute of the United States on the basis of the theory of CASs[110]. It provides an efficient, reliable, and reusable software experiment platform. By establishing computer models based on Swarm and invoking the rich class libraries provided in the platform, simulation can be performed in many research fields.

Given that the model and the way of interaction between the model elements in Swarm are without restrictions, the user can focus on their interested specific

systems rather than being bothered by data process, user interface, and other pure software and programming problems. It is also user friendly for noncomputer professional scholars. Therefore, Swarm has received wide attention from economics, management science, ecology, systematics, military simulation, computer science, and other fields.

Generally, the Swarm model is composed of ModelSwarm, ObserverSwarm, and Individual Agent and Environment, and it supports the analysis, display, and control of simulation experiments through the class library. ModelSwarm includes a schedule of actions in the model and a set of inputs and outputs. The input includes model parameters, such as the number of objects and initial value. The output includes the value of the variable to be observed and the result of the model. ObserverSwarm is a window for target model observation and measurement, including a set of individuals and a schedule of behavior. Among them, the individual is the detector used for observation and the output interface, such as charts and two-dimensional grid points. The behavior schedule is used to describe the interval and sequence of sampling for each detector.

The Swarm agent can be further summarized by four characteristics based on the CAS-based Swarm modeling idea and its structure. The first is aggregation that a single agent can adhere to each other to form the aggregation of multi-agents that have the same movement trend as a single agent. The second is nonlinearity, that is, agents and their properties in the event of change are not completely linear but nonlinear. The third is flow, which refers to the exchange of information flow, energy flow, and material flow between agents. In addition, the channel and speed of flow directly affect the process of the system. The fourth is diversity, that is, a differentiation trend exists between agents.

In addition, the activation mechanisms of Swarm agents are as follows: The first is tagging, which helps realize information exchange and propose specific implementation methods of searching and receiving information in the environment. The second is internal models, which indicate the concept of hierarchy, that is, every agent has a complex internal mechanism. The third is building blocks, that is, complex systems are often formed on the basis of relatively simple components by changing their combination. Therefore, Swarm's agents are living individuals with multiple levels, continuous interaction with the outside world, and continuous development and evolution.

6.4 REPAST

REPAST was jointly developed by the University of Chicago and Argonne National Laboratory and was subsequently maintained and updated by the REPAST Organization for Architecture and Development[111]. The platform supports Java, C#, and Python. Its software architecture is similar to Swarm, which is mainly used in the field of social science and has professional tools for social science model development. The platform provides some simple model libraries, class libraries, and genetic, regression, and other algorithms, which can be used for model development by using interfaces and display agent simulation data. REPAST was only available as an implementation in Java and has been available in C# and Python since version 3.0 was proposed to expand the REPAST user base.

Since the release of REPAST, several applications can be divided into the following four categories:

(1) Theoretical research: The generation process of a specific phenomenon in the system is observed through simulation, and the general rules of CAS can be found and verified. Examples include implementing the famous model "ECHO" in CAS theory with REPAST and studying game theory through multi-agent simulation, such as the prisoner's dilemma problem.

(2) Social system simulation: Studies include the interaction between agents and their environment and how multi-agents with different goals and interests achieve cooperative behavior. In addition, the agent can be an individual or an organization.

(3) Economic system simulation: In agent-based computational economics (an emerging branch of economics), REPAST has been used to implement and simulate economic models (e.g., commercial network simulation and supply chain simulation).

(4) Comprehensive application: The Argonne National Laboratory of the United States has extended REPAST to support GIS, distributed simulation, and other functions, and developed some large CAS simulation (CASS) on this basis (e.g., the United States electricity market simulation).

Advantages of REPAST: REPAST borrows substantial design experience from Swarm, and the graphical user interfaces of the two are similar. Hence, REPAST is considered as a Swarm-like emulation kit. Four multi-agent simulation tools were evaluated and compared, including REPAST and Swarm, and the results showed that REPAST ranked the first in

almost all the scoring programs, such as documentation, modeling, simulation capability, and usability; its synthesis score is also the highest.

6.5 MASON

MASON, which was developed by George MASON University[112], is programmed with Java language and is mainly used for agent-based discrete event simulation. MASON's main characteristics are fast execution, flexible use, and graphical interface for 2D and 3D visual display. Given the limited software size of the MASON platform, it can only perform simulations of a lower-magnitude model.

6.6 AnyLogic

AnyLogic, which was developed by XJ Technologies[113], is a widely used tool for discrete, system dynamics, and multi-agent and hybrid system modeling and simulation. In addition to the basic simulation, the platform also contains enterprise libraries. AnyLogic supports development with Java and Unified Modeling Language (UML)-Real Time, as well as models by differential equations. Its professional library covers a wide range of fields, including logistics, transportation, urban planning, and other aspects. AnyLogic is the first to use UML for simulation and is the only commercial software that supports mixed state machine language for simulation development. AnyLogic has a complete visual window that can observe the simulation process clearly and intuitively. It can integrate multi-intelligence simulation with machine learning, build a training environment for multi-agent reinforcement learning schemes, and model all cooperative, competitive, or hierarchical behavior.

6.7 JCass

JCass is a distributed simulation platform for complex systems and was developed by the State Key Laboratory of Parallel and Distributed Processing, National University of Defense Technology[114]. It is a general CASS platform that can be applied in many disciplines.

JCass supports hierarchical simulation and hierarchical scheduling, allowing users to build and test multitier models for the description of emergence. The basic unit of JCass simulation is agents who communicate with each other through a message mechanism. JCass provides not only a conservative time promotion mechanism and an optimistic time promotion mechanism but also a hierarchical hybrid time management protocol. It is developed in Java, supports cross-platform emulation, and provides libraries for building agents and debugging programs.

6.8 Comparison among platforms

Considering the space environment of the model, JADE and Swarm only support 2D space environments, whereas other platforms can support 2D and 3D space environments. In terms of algorithms, NetLogo does not support complex algorithms temporarily. Swarm and REPAST support genetic algorithms, neural network algorithms, and other Java computing packages. MASON supports evolutionary algorithms and other Java computing packages. AnyLogic and JADE support any Java language based algorithm. In terms of graphics and data visualization, all the platforms except JADE are equipped with a visual display module. In terms of performance, MASON, REPAST, and Swarm are framework class library platforms, among which the development of MASON is relatively immature, whereas the development of Swarm is relatively mature. Swarm is a stand-alone platform, which has low portability and poor simulation performance for complex situations. Similar to Swarm, REPAST is suitable for simple and small-scale simulations, and its composition and design cannot be self-improved. NetLogo and AnyLogic are commercial softwares without independent property rights for platform development and thus need a certain fee to obtain all functions. JADE, an open-source platform, is flexible and thus can be distributed across different hosts. JCass is a general simulation platform that uses the Java programming language. It can strongly support the heterogeneous distributed platform, design the rule base based on XML, and support hierarchical multi-agent modeling. However, it also has shortcomings, such as the single underlying communication method, high cost of time management, nonsupport of environment visual modeling, and insufficient support for evolution mechanism modeling.

7 Conclusion

The concepts and ideas of MAMS, including computational entities, model aggregation, perception and behavior, decentralized control, environment, and interaction, were introduced. This study expounded the advantages of multi-agent simulation technology from seven aspects, namely, descriptive ability, emergence analysis, organizational framework and evolutionary mechanism, autonomic solution and decision-making ability, interaction ability, distributed simulation, and

model reuse, and presented the application status of the technology in the social, economic, and military fields.

This work studied the developing status of hybrid modeling and simulation based on system dynamics and multi-agents and pointed out the advantages and disadvantages of SDMS and MAMS in the complex system simulation. SDMS focuses on system dynamic behavior, analyzing the interaction between different elements and accumulation effects while ignoring the spatial factors. MAMS focuses on the interaction in space and ignores the feedback effect of macro social and economic factors on agents. The hybrid model based on MAMS and SDMS is an effective method for complex system modeling and simulation.

Our study reviewed the development of multi-agent reinforcement learning modeling and simulation and presented the algorithm problems to be solved in multi-agent reinforcement learning. These problems are summarized as follows: First, the convergence and stability of multi-agent reinforcement learning have not been proven systematically. Second, the state space of multi-agent reinforcement learning is too large, and the training time is long. Third, the expansibility and knowledge transfer of multi-agent reinforcement learning are poor.

The current situation of large-scale MAMS was also summarized. The two most effective and common solutions for large-scale MAMS were analyzed. First, the model was reorganized at the software level via "super individual". Second, distributed parallel computing was used to accelerate large-scale computing.

Typical MAMS platforms, including JADE, NetLogo, Swarm, REPAST, MASON, and AnyLogic, were introduced. Furthermore, representative and widely used general multi-agent modeling and simulation platforms were compared and analyzed.

References

[1] C. M. Macal and M. J. North, Agent-based modeling and simulation: Desktop ABMS, presented at 2007 Winter Simulation Conf., Washington, DC, USA, 2007, pp. 95–106.

[2] Z. B. Tao He, Research on the key technology of multi-agent system design, (in Chinese), *Modern Electron. Techn.*, vol. 29, no. 14, pp. 31–34, 2006.

[3] K. Liu, Modeling and simulation of complex adaptive system based on multi-agent, PhD dissertation, Central South University, Changsha, China, 2011.

[4] M. Abbas, Agent-based modeling and simulation, presented at Artificial Intelligence Applications to Critical Transportation Issues, Washington, DC, USA, 2012, p. 58.

[5] H. Z. Deng, Modeling and simulation method and its application based on multi-agent, (in Chinese), PhD dissertation, National University of Defense Technology, Xi'an, China, 2002.

[6] H. Z. Deng, Y. J. Tan, and Y. Chi, A complex system research method-multiagent-based ensemble modeling and simulation method, (in Chinese), *Syst. Eng.*, vol. 18, no. 4, pp. 73–78, 2000.

[7] B. J. L. Berry, L. D. Kiel, and E. Elliott, Adaptive agents, intelligence, and emergent human organization: Capturing complexity through agent-based modeling, *Proc. Nat. Acad. Sci. USA*, vol. 99, no. S3, pp. 7187–7188, 2002.

[8] M. H. Haghighat and J. Li, Intrusion detection system using voting-based neural network, *Tsinghua Science and Technology*, vol. 26, no. 4, pp. 484–495, 2021.

[9] J. Zhang, Z. Tang, Y. Xie, M. Ai, and W. Gui, Visual perception-based fault diagnosis in froth flotation using statistical approaches, *Tsinghua Science and Technology*, vol. 26, no. 2, pp. 172–184, 2021.

[10] D. Bloembergen, K. Tuyls, D. Hennes, and M. Kaisers, Evolutionary dynamics of multi-agent learning: A survey, *J. Artif. Intell. Res.*, vol. 53, no. 1, pp. 659–697, 2015.

[11] P. Tranouez, E. Daudé, and P. Langlois, A multiagent urban traffic simulation, arXiv preprint arXiv: 1201.5472, 2012.

[12] D. Schreiber, The emergence of parties: An agent-based simulation, *Polit. Res. Quart.*, vol. 67, no. 1, pp. 136–151, 2014.

[13] J. L. Casti, *Would-be Worlds*: *How Simulation is Changing the Frontiers of Science*. New York, NY, USA: John Wiley & Sons, Inc., 1998.

[14] B. Raney, N. Cetin, A. Völlmy, M. Vrtic, K. Axhausen, and K. Nagel, An agent-based microsimulation model of swiss travel: First results, *Netw. Spat. Econ.*, vol. 3, no. 1, pp. 23–41, 2003.

[15] J. M. Epstein and R. Axtell, *Growing Artificial Societies*: *Social Science from the Bottom Up*. Washington, DC, USA: Brookings Institution Press, 1996.

[16] C. Bilge, Venables, an agent-based model of a supermarket, http://www.simworld.co.uk/, 2021.

[17] R. Pryor, N. Basu, and T. Quint, Development of ASPEN: A microanalytic simulation model of the US economy, Working Paper SAND-96–0434, Sandia National Laboratories, Albuquerque, NM, USA, 1996.

[18] N. Basu and R. J. Pryor, Growing a market economy. SAND-97-2093 Distribution, presented at Unlimited Release Category UCD905, Albuquerque, NM, USA, 1997.

[19] W. B. Arthur, S. N. Durlauf, and D. Lane, *The Economy As an Evolving Complex System II*. Boca Raton, FL, USA: CRC Press, 1997.

[20] W. B. Arthur, J. H. Holland, B. LeBaron, R. Palmer, and P. Tayler, Asset pricing under endogenous expectations in an artificial stock market, presented at the Economy As an Evolving Complex System II, Reading, MA, USA, 1996.

[21] A. Khalafallah, Neural network based model for predicting housing market performance, *Tsinghua Science and Technology*, vol. 13, no. S1, pp. 325–328, 2008.

[22] H. Rahmandad and J. Sterman, Heterogeneity and network structure in the dynamics of diffusion: Comparing agent-based and differential equation models, *Manag. Sci.*, vol.

54, no. 5, pp. 998–1014, 2008.
[23] M. Komosinski and A. Adamatzky, *Artificial Life Models in Software*. 2nd ed. London, UK: Springer Science & Business Media, 2009.
[24] T. Lorenz, Abductive fallacies with agent-based modeling and system dynamics, in *Int. Workshop on Epistemological Aspects of Computer Simulation in the Social Sciences*, F. Squazzoni, ed. Berlin, Germany: Springer-Verlag, 2009, pp. 141–152.
[25] T. Lorenz and A. Jost, Towards an orientation framework in multi-paradigm modeling, in *Proc.*24^{th} *Int. Conf. System Dynamics Society*, Nijmegen, the Netherlands, 2006, pp. 2134–2151.
[26] Z. K. Ding, W. Y. Gong, S. H. Li, and Z. Z. Wu, System dynamics versus agent-based modeling: A review of complexity simulation in construction waste management, *Sustainability*, vol. 10, no. 7, p. 2484, 2018.
[27] C. Swinerd and K. R. McNaught, Design classes for hybrid simulations involving agent-based and system dynamics models, *Simulat. Model. Pract. Theory*, vol. 25, pp. 118–133, 2012.
[28] C. Heinze, B. Smith, and M. Cross, Thinking quickly: Agents for modeling air warfare, in *Australian Joint Conf. Artificial Intelligence AI 1998*, G. Antoniou and J. Slaney, eds. Berlin, Germany: Springer, 1998, pp. 47–58.
[29] R. W. Hill, J. Chen, J. Gratch, P. Rosenbloom, and M. Tambe, Soar-RWA: Planning, teamwork, and intelligent behavior for synthetic rotary wing aircraft, in *Proc.*7^{th} *Conf. Computer Generated Forces & Behavioral Representation*, Orlando, FL, USA, 1998, pp. 12–14.
[30] A. Alvanchi, S. Lee, and S. AbouRizk, Modeling framework and architecture of hybrid system dynamics and discrete event simulation for construction, *Comp. Aided Civil Infrastruct. Eng.*, vol. 26, no. 2, pp. 77–91, 2011.
[31] L. Lättilä, P. Hilletofth, and B. S. Lin, Hybrid simulation models–when, why, how? *Expert Syst. Appl.*, vol. 37, no. 12, pp. 7969–7975, 2010.
[32] J. W. Forrester, System dynamics, systems thinking, and soft OR, *Syst. Dynam. Rev.*, vol. 10, nos. 2 & 3, pp. 245–256, 1994.
[33] F. Nasirzadeh, M. Khanzadi, and M. Mir, A hybrid simulation framework for modelling construction projects using agent-based modelling and system dynamics: An application to model construction workers' safety behavior, *Int. J. Construct. Manag.*, vol. 18, no. 2, pp. 132–143, 2018.
[34] D. D. Wu, K. F. Xie, H. Liu, S. Zhao, and D. L. Olson, Modeling technological innovation risks of an entrepreneurial team using system dynamics: An agent-based perspective, *Technol. Forecast. Soc. Change*, vol. 77, no. 6, pp. 857–869, 2010.
[35] F. Nasirzadeh, A. Afshar, and M. Khanzadi, System dynamics approach for construction risk analysis, *Int. J. Civil Eng.*, vol. 6, no. 2, pp. 120–131, 2008.
[36] J. Swanson, Business dynamics-systems thinking and modeling for a complex world, *J. Oper. Res. Soc.*, vol. 53, no. 4, pp. 472–473, 2002.
[37] M. Watkins, A. Mukherjee, N. Onder, and K. Mattila, Using agent-based modeling to study construction labor productivity as an emergent property of individual and crew interactions, *J. Construct. Eng. Manage.*, vol. 135, no. 7, pp. 657–667, 2009.
[38] S. E. Phelan, A note on the correspondence between complexity and systems theory, *Syst. Pract. Act. Res.*, vol. 12, no. 3, pp. 237–246, 1999.
[39] J. M. Epstein and R. Axtell, *Growing Artificial Societies: Social Science from the Bottom Up*. Cambridge, MA, USA: MIT Press, 1996.
[40] M. Barbati, G. Bruno, and A. Genovese, Applications of agent-based models for optimization problems: A literature review, *Expert Syst. Appl.*, vol. 39, no. 5, pp. 6020–6028, 2012.
[41] H. J. Scholl, Agent-based and system dynamics modeling: A call for cross study and joint research, in *Proc.*34^{th} *Ann. Hawaii Int. Conf. System Sciences*, Maui, HI, USA, 2001.
[42] J. Pourdehnad, K. Maani, and H. Sedehi, System dynamics and intelligent agent-based simulation: Where is the synergy, in *Proc.* 20^{th} *Int. Conf. System Dynamics Society*, Palermo, Italy, 2002, pp. 1–16.
[43] L. Stemate, C. Pasca, and I. Taylor, A comparison between system dynamics and agent based modeling and opportunities for cross-fertilization, in *Proc. Winter Simulation Conf.*, Washington, DC, USA, 2007, pp. 2376.
[44] N. Schieritz, Integrating system dynamics and agent-based modeling, in *Proc.* 20^{th} *Int. Conf. System Dynamics Society*, Palermo, Italy, 2002, pp. 1–3.
[45] N. Schieritz and P. M. Milling, Modeling the forest or modeling the trees: A comparison of system dynamics and agent-based simulation, in *Proc. System Dynamics Conf.*, New York, NY, USA, 2003, pp. 1–15.
[46] N. Schieritz N and A. Grobler, Emergent structures in supply chains-a study integrating agent-based and system dynamics modeling, in *Proc.* 36^{th} *Ann. Hawaii Int. Conf. System Sciences*, Big Island, HI, USA, 2003, p. 9.
[47] M. Teose, K. Ahmadizadeh, E. O'Mahony, R. L. Smith, L. Zhao, S. P. Ellner, C. Gomes, and Y. Grohn, Embedding system dynamics in agent based models for complex adaptive systems, in *Proc.* 22^{nd} *Int. Joint Conf. Artificial Intelligence*, Barcelona, Spain, 2011, pp. 2531–2538.
[48] E. Shafiei, H. Stefansson, E. I. Asgeirsson, B. Davidsdottir, and M. Raberto, Integrated agent-based and system dynamics modelling for simulation of sustainable mobility, *Transp. Rev.*, vol. 33, no. 1, pp. 44–70, 2013.
[49] C. M. Macal, To agent-based simulation from system dynamics, in *Proc. 2010 Winter Simulation Conf.*, Baltimore, MD, USA, 2010, pp. 371–382.
[50] A. Djanatliev, R. German, P. Kolominsky-Rabas, and B. M. Hofmann, Hybrid simulation with loosely coupled system dynamics and agent-based models for prospective health technology assessments, in *Proc. 2012 Winter Simulation Conf. (WSC)*, Berlin, Germany, 2012, pp. 1–12.
[51] G. P. Figueredo, P. O. Siebers, U. Aickelin, A. Whitbrook, and J. M. Garibaldi, Juxtaposition of system dynamics and agent-based simulation for a case study in immunosenescence, *PLoS One*, vol. 10, no. 3, p. e0118359, 2015.
[52] J. Schryver, J. Nutaro, and M. Shankar, Emulating a system dynamics model with agent-based models: A methodological case study in simulation of diabetes progression, *Open J. Model. Simul.*, vol. 3, no. 4, p. 60811,

2015.

[53] F. Ferrada and L. M. Camarinha-Matos, A system dynamics and agent-based approach to model emotions in collaborative networks, in *Doctoral Conference on Computing*, *Electrical and Industrial Systems*, L. Camarinha-Matos, M. Parreira-Rocha, and J. Ramezani, eds. Springer, 2017, pp. 29–43.

[54] N. Marilleau, C. Lang, and P. Giraudoux, Coupling agent-based with equation-based models to study spatially explicit megapopulation dynamics, *Ecol. Model.*, vol. 384, pp. 34–42, 2018.

[55] R. S. Sutton and A. G. Barto, *Reinforcement Learning*: *An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[56] M. D. Awheda and H. M. Schwartz, The residual gradient FACL algorithm for differential games, presented at 2015 IEEE 28th Canadian Conf. Electrical and Computer Engineering (CCECE), Halifax, Canada, 2015, pp. 1006–1011.

[57] W. D. Smart and L. Pack Kaelbling, Effective reinforcement learning for mobile robots, in *Proc. 2002 IEEE Int. Conf. Robotics and Automation* (*Cat.No.02CH37292*), Washington, DC, USA, 2002, pp. 3404–3410.

[58] M. L. Littman, Markov games as a framework for multi-agent reinforcement learning, in *Proc. 12th Int. Conf.*, Elsevier, 1994, pp. 157–163.

[59] B. Luo, H. N. Wu, and T. W. Huang, Off-policy reinforcement learning for H_∞ control design, *IEEE Trans. Cybernet.*, vol. 45, no. 1, pp. 65–76, 2015.

[60] B. Luo, H. N. Wu, and H. X. Li, Data-based suboptimal neuro-control design with reinforcement learning for dissipative spatially distributed processes, *Ind. Eng. Chem. Res.*, vol. 53, no. 19, pp. 8106–8119, 2014.

[61] W. Dixon, Optimal adaptive control and differential games by reinforcement leanring principles [book review], *IEEE Contr. Syst. Mag.*, vol. 34, no. 3, pp. 80–82, 2014.

[62] P. X. Cai and Y. Zhang, Intelligent cognitive spectrum collaboration: Convergence of spectrum sensing, spectrum access, and coding technology, *Intelligent and Converged Networks*, vol. 1, no. 1, pp. 79–98, 2020.

[63] B. Hou, F. C. Sun, H. B. Li, and G. B. Liu, Consensus of second-order multi-agent systems with time-varying delays and antagonistic interactions, *Tsinghua Science and Technology*, vol. 20, no. 2, pp. 205–211, 2015.

[64] Z. H. Zhao, Y. Gao, B. Luo, and S. F. Chen, Reinforcement learning technology in multi-agent system, (in Chinese), *Comp. Sci.*, vol. 31, no. 3, pp. 23–27, 2004.

[65] R. S. Sutton and A. G. Barto, *Reinforcement Learning*: *An Introduction*. Cambridge, MA, USA: MIT Press, 1998, pp. 133–160.

[66] C. J. C. H. Watkins and P. Dayan, Q-learning, *Mach. Learn.*, vol. 8, nos. 3&4, pp. 279–292, 1992.

[67] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, Policy gradient methods for reinforcement learning with function approximation, in *Proc. 12th Int. Conf. Neural Information Proc. Systems*, Denver, CO, USA, 2000, pp. 1057–1063.

[68] P. Stone and M. Veloso, Multiagent systems: A survey from a machine learning perspective, *Auton. Robots*, vol. 8, no. 3, pp. 345–383, 2000.

[69] Y. Shoham, R. Powers, and T. Grenager, If multi-agent learning is the answer, what is the question? *Artif. Intell.*, vol. 171, no. 7, pp. 365–377, 2006.

[70] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, Playing atari with deep reinforcement learning, arXiv preprint arXiv: 1312.5602, 2013.

[71] G. Papoudakis, F. Christianos, A. Rahman, and S. V. Albrecht, Dealing with non-stationarity in multi-agent deep reinforcement learning, arXiv preprint arXiv: 1906.04737, 2019.

[72] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications, *IEEE Trans. Cybernet.*, vol. 50, no. 9, pp. 3826–3839, 2020.

[73] L. Busoniu, R. Babuska, and B. De Schutter, A comprehensive survey of multiagent reinforcement learning, *IEEE Trans. Syst. Man Cybernet. Part C Appl. Rev.*, vol. 38, no. 2, pp. 156–172, 2008.

[74] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, A survey and critique of multiagent deep reinforcement learning, *Auton. Agents Multi Agent Syst.*, vol. 33, no. 6, pp. 750–797, 2019.

[75] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, Multiagent cooperation and competition with deep reinforcement learning, *PLoS One*, vol. 12, no. 4, p. e0172395, 2017.

[76] M. Raghu, A. Irpan, J. Andreas, B. Kleinberg, Q. V. Le, and J. Kleinberg, Can deep reinforcement learning solve erdos-selfridge-spencer games? arXiv preprint arXiv: 1711.02301, 2018.

[77] S. Sukhbaatar, A. Szlam, and R. Fergus, Learning multiagent communication with backpropagation, presented at 30th Ann. Conf. Neural Information Proc. Systems, Barcelona, Spain, 2016, pp. 2244–2252.

[78] J. C. Jiang and Z. Q. Lu, Learning attentional communication for multi-agent cooperation, in *Proc. 32nd Int. Conf. Neural Information Processing Systems*, Montréal, Canada, 2018, pp. 7254–7264.

[79] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, Multi-agent actor-critic for mixed cooperative-competitive environments, in *Proc. 31st Int. Conf. Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 6379–6390.

[80] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, Counterfactual multi-agent policy gradients, arXiv preprint arXiv: 1705.08926, 2018.

[81] S. V. Albrecht and P. Stone, Autonomous agents modelling other agents: A comprehensive survey and open problems, *Artif. Intell.*, vol. 258, pp. 66–95, 2018.

[82] R. Raileanu, E. Denton, A. Szlam, and R. Fergus, Modeling others using oneself in multi-agent reinforcement learning, arXiv preprint arXiv: 1802.09640, 2018.

[83] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Pérolat, D. Silver, and T. Graepel, A unified game-theoretic approach to multiagent reinforcement learning, in *Proc. 31st Int. Conf. Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 4190–4203.

[84] L. O. Souza, G. de Oliveira Ramos, and C. G. Ralha, Experience sharing between cooperative reinforcement learning agents, presented at 2019 IEEE 31st Int. Conf. Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 2019, pp. 963–970.

[85] S. Barrett, A. Rosenfeld, S. Kraus, and P. Stone, Making friends on the fly: Cooperating with new teammates, *Artif. Intell.*, vol. 242, pp. 132–171, 2017.

[86] P. Y. Jiang, M. L. Yang, W. D. Li, J. J. Liu, W. Guo, and P. L. Li, Ci literature review and its application exploration in social manufacturing, (in Chinese), *Chin Mech. Eng.*, vol. 31, no. 15, pp. 1852–1865, 2020.

[87] OpenAI, Openai five, https://blog.openai.com/openai-five/, 2018.

[88] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al., Grandmaster level in starcraft II using multi-agent reinforcement learning, *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[89] H. R. Parry and M. Bithell, Large scale agent-based modelling: A review and guidelines for model scaling, in *Agent-Based Models of Geographical Systems*, A. Heppenstall, A. Crooks, L. See, and M. Batty, eds. Springer, 2012, pp. 271–308.

[90] H. R. Parry and A. J. Evans, A comparative analysis of parallel processing and super-individual methods for improving the computational performance of a large individual-based model, *Ecol. Model.*, vol. 214, nos. 2&4, pp. 141–152, 2008.

[91] F. Ablayev, M. Ablayev, J. Z. Huang, K. Khadiev, N. Salikhova, and D. M. Wu, On quantum methods for machine learning problems part I: Quantum tools, *Big Data Mining and Analytics*, vol. 3, no. 1, pp. 41–55, 2020.

[92] J. Q. Huang, W. T. Han, X. Y. Wang, and W. G. Chen, Heterogeneous parallel algorithm design and performance optimization for WENO on the sunway taihulight supercomputer, *Tsinghua Scicence and Technology*, vol. 25, no. 1, pp. 56–67, 2020.

[93] Z. Bo, H. C. Zhou, G. Q. Li, and Y. H. Huang, ZenLDA: Large-scale topic model training on distributed data-parallel platform, *Big Data Mining and Analytics*, vol. 1, no. 1, pp. 57–74, 2018.

[94] J. Blythe, J. Bollenbacher, D. Huang, P. M. Hui, R. Krohn, D. Pacheco, G. Muric, A. Sapienza, A. Tregubov, Y. Y. Ahn, et al., Massive multi-agent data-driven simulations of the GitHub ecosystem, in *Int. Conf. Practical Applications of Agents and Multi-Agent Systems*, Y. Demazeau, E. Matson, J. Corchado, and F. De la Prieta, eds. Springer, 2019, pp. 3–15.

[95] J. C. Campagne, A. Cardon, E. Collomb, and T. Nishida, Massive multi-agent systems control, in *Int. Workshop on Formal Approaches to Agent-Based Systems*, M. G. Hinchey, J. L. Rash, W. F. Truszkowski, and C. A. Rouff, eds. Springer, 2004, pp. 275–280.

[96] Z. J. Zhou and H. Xu, Decentralized adaptive optimal control for massive multi-agent systems using mean field game with self-organizing neural networks, presented at 2019 IEEE 58th Conf. Decision and Control (CDC), Nice, France, 2019, pp. 1225–1230.

[97] N. Fachada, V. V. Lopes, R. C. Martins, and A. C. Rosa, Parallelization strategies for spatial agent-based models, *Int. J. Parallel Program.*, vol. 45, no. 3, pp. 449–481, 2017.

[98] N. Fachada, V. V. Lopes, R. C. Martins, and A. C. Rosa, Towards a standard model for research in agent-based modeling and simulation, *PeerJ Comp. Sci.*, vol. 1, no. 2, p. e36, 2015.

[99] A. Saprykin, N. Chokani, and R. S. Abhari, Large-scale multi-agent mobility simulations on a GPU: Towards high performance and scalability, *Proced. Comp. Sci.*, vol. 151, pp. 733–738, 2019.

[100] W. Marurngsith and Y. Mongkolsin, Creating GPU-enabled agent-based simulations using a PDES tool, in *Distributed Computing and Artificial Intelligence Advances in Intelligent Systems & Computing*, S. Omatu, J. Neves, J. Rodriguez, J. Paz Santana, and S. Gonzalez, eds. Springer, vol. 217, pp. 227–234, 2013.

[101] N. Collier and M. North, Parallel agent-based simulation with repast for high performance computing, *Simulation*, vol. 89, no. 10, pp. 1215–1235, 2013.

[102] F. Klügl and G. Rindsfüser, Large-scale agent-based pedestrian simulation, in *German Conf. Multiagent System Technologies*, P. Petta, J. P. Müller, M. Klusch, and M. Georgeff, eds. Springer, 2007, pp. 145–156.

[103] L. Zhang, W. C. Yang, J. M. Wang, and Q. Rao, Large-scale agent-based transport simulation in Shanghai, China, *Transport. Res. Rec. J. Transport. Res. Board*, vol. 2399, no. 1, pp. 34–43, 2013.

[104] R. Haroun, F. Boumghar, S. Hassas, and L. Hamami, A massive multi-agent system for brain MRI segmentation, presented at Int. Workshop on Massively Multiagent Systems, Kyoto, Japan, 2004, pp. 174–186.

[105] M. X. Zhang, R. Q. Meng, and A. Verbraeck, Including public transportation into a large-scale agent-based model for epidemic prediction and control, in *Proc. Summer Computer Simulation* (*SummerSim '15*). *Society for Computer Simulation International*, San Diego, CA, USA, 2015, pp. 1–8.

[106] T. Suzumura, C. Houngkaew, and H. Kanezashi, Towards billion-scale social simulations, in *Proc. Winter Simulation Conf.*, Savannah, GA, USA, 2015, pp. 781–792.

[107] L. A. N. Laboratory, Traffic analysis simulation system, http://transims.tsasa.lanl.gov, 2021.

[108] W. H. Yu, *JADE-based Multi-agent System Development Technology*, (in Chinese). Beijing, China: National Defense Industry Press, 2011.

[109] C. Zhu, Macroscopic and microscopic expressway traffic flow modeling and simulation based on multi-agent system, Master dissertation, Beijing University of Technology, Beijing, China, 2016.

[110] S. Y. Liao, J. Chen, H. W. Lu, and J. H. Dai, Summarization of agent-based modeling and simulation, (in Chinese), *Comput. Simul.*, vol. 25, no. 12, pp. 1–7, 2008.

[111] D. C Zhou and X. P. Wu, Realization of ACS in repast, (in Chinese), *J. Wuhan Univ. Technol.*, vol. 29, no. 8, pp. 121–124, 2007.

[112] Department of Computer Science, Nguyen Engineering Building, 4400 University Drive, Mason, https://

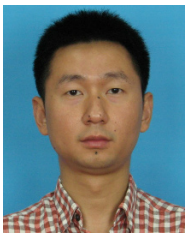
cs.gmu.edu/, 2021.
[113] T. A. Company, Anylogic, https://www.anylogic.cn/, 2021.
[114] M. Zhou, The design and implement of agent modeling in complex system distributed simulation platform, (in Chinese), Master dissertation, National University of Defense Technology, Xi'an, China, 2013.

Wenhui Fan received the PhD degree in mechanical engineering from Zhejiang University, Hangzhou, China in 1998. He obtained the postdoctoral certificate from Tsinghua University, Beijing, China in 2000. He is a vice president of China Simulation Federation. He is currently a professor at Tsinghua University, Beijing, China. His current research interests include multi-agent modeling and simulation, large scale agent modeling and simulation, and multi-agent reinforcement learning.

Peiyu Chen received the BS degree in automation from Beihang University, Beijing, China in 2019. She is now pursuing the PhD degree at Department of Automation, Tsinghua University, Beijing, China. Her research interests include complex network, multi-agent system, and hybrid simulation.

Daming Shi received the BS degree in automation from Beihang University, Beijing, China in 2018. He is now pursuing the PhD degree at Department of Automation, Tsinghua University, Beijing, China. His main research interests include multi-agent system, reinforcement learning, scheduling, and game theory.

Xudong Guo received the BS degree in automation from Tsinghua University, Beijing, China in 2020. He is now pursuing the PhD degree at Department of Automation, Tsinghua University, Beijing, China. His research interests include complex network modeling and multi-agent system.



Li Kou received the master degree from the University of Defense and Technology, Changsha, Hunan in 2006. Currently, he is pursuing the PhD degree at Tsinghua University. His research interests include modeling and simulation of complex system, intelligent decision, and information system engineering.