A reflective memory based framework for crowd network simulations

Sun Hongbo and Mi Zhang
School of Computer and Control Engineering, Yantai University, Yantai, China

Received 28 January 2018
Revised 12 April 2018
Accepted 13 April 2018

Abstract

Purpose – As main mode of modern service industry and future economy society, the research on crowd network can greatly facilitate governances of economy society and make it more efficient, humane, sustainable and at the same time avoid disorders. However, because most results cannot be observed in real world, the research of crowd network cannot follow a traditional way. Simulation is the main means to put forward related research studies. Compared with other large-scale interactive simulations, simulation for crowd network has challenges of dynamic, diversification and massive participants. Fortunately, known as the most famous and widely accepted standard, high level architecture (HLA) has been widely used in large-scale simulations. But when it comes to crowd network, HLA has shortcomings like fixed federation, limited scale and agreement outside the software system.

Design/methodology/approach – This paper proposes a novel reflective memory-based framework for crowd network simulations. The proposed framework adopts a two-level federation-based architecture, which separates simulation-related environments into physical and logical aspect to enhance the flexibility of simulations. Simulation definition is introduced in this architecture to resolve the problem of outside agreements and share resources pool (constructed by reflective memory) is used to address the systemic emergence and scale problem.

Findings – With reference to HLA, this paper proposes a novel reflective memory-based framework toward crowd network simulations. The proposed framework adopts a two-level federation-based architecture, system-level simulation (system federation) and application-level simulation (application federations), which separates simulation-related environments into physical and logical aspect to enhance the flexibility of simulations. Simulation definition is introduced in this architecture to resolve the problem of outside agreements and share resources pool (constructed by reflective memory) is used to address the systemic emergence and scale problem.

Originality/value – Simulation syntax and semantic are all settled under this framework by templates, especially interface templates, as simulations are separated by two-level federations, physical and logical simulation environment are considered separately; the definition of simulation execution is flexible. When developing new simulations, recompile is not necessary, which can acquire much more reusability, because reflective memory is adopted as share memory within given simulation execution in this framework; population can be perceived by all federates, which greatly enhances the scalability of this kind of simulations; communication efficiency and capability has greatly improved by this share memory-based framework.

Keywords Crowd network, HLA-high level architecture, Large scale simulation, Reflective memory

Paper type Technical paper

This work is supported by National Key Research and Development Program of China (Grant No. 2017YFB1400105).

1. Introduction

As the proverb goes "two heads are better than one" and "everybody's business is nobody's business", the phenomenon of crowd intelligence can be easily observed in our daily lives. At the same time, with the rapid development of network technologies, crowd intelligence becomes much more complicate and universal, for human, enterprises, governments, equipments and articles turns to be more and more intelligent, and these intelligent agents are connecting to form numerous crowd network systems, such as e-commerce platforms, networked supply chains, Wikipedia and network elections (Michelucci and Dickinson, 2016).

As main mode of modern service industry and future economy society, the research on crowd network can greatly facilitate governances of economy society and make it more efficient, humane, sustainable and at the same time avoid disorders (Chai *et al.*, 2017). However, because most results cannot be observed in real world, the research of crowd network cannot follow a traditional way. Simulation is the main means to put forward related research studies.

Compared with traditional large-scale simulations, crowd network simulations have several obvious challenges as follows:

- Dynamic. Member attributes and states of crowd network simulations may vary with time in an uncertain mode. Members are more loosely coupled, but member behaviors and intention variations may lead to change of group states and intentions in extending scopes.
- Diversification. Time advance strategy may base on slow variables, events, clock or hybrid mode. And as members are multiform and multi-disciplinary, transactions are uncertain and various, disturbances have several sources and subscriptions exist in different layer and aspects, disturbances injection strategy and matching strategy are all need to take diversification into considerations.
- Scale. Member of crowd network simulations may need to achieve millions or even trillions so as to discover or verify its principals and regularities.

In a word, crowd network simulation is a new development of large scale simulations, which faces both opportunities and challenges. This paper proposes a novel reflective memory-based architecture to resolve problems mentioned above, and the rest of this paper is organized as follows: Section 2 gives a brief review of related efforts towards crowd network simulations, Section 3 analyzes shortcomings and advantages of high level architecture (HLA) based simulations, Section 4 proposes a novel reflective memory-based simulation architecture; Section 5 demonstrates an implementation architecture and Section 6 draws a conclusion of the proposed approach.

2. Related work

Nowadays, most economy and society experiments are depended on simulations, for they can greatly enhance replicability of these experiments, such as Anylogic, Swarm, Repast, MASON, NetLogo, GEMPACK and HLA.

AnyLogic is a widely used modeling and simulation tool for discrete, system dynamics, multi-agent and hybrid systems, which adopts a model driven architecture as its bases. Wallis and Paich (2017) propose an agent-based model (ABM) with reinforcement learning for autonomous fleet coordination; they demonstrate and describe in detail a version of the AnyLogic Consumer Market Model that has been modified to include adaptive dynamics based on deep learning and describe approaches to integrating machine learning to the design and development of simulations. But when it comes to agent-based simulation, AnyLogic classifies population into several groups, and individuals cannot be distinguished from their group. As individuals in crowd network are all customized, it is not convenient for crowd network simulations.

Developed by Santa fe Institute of the USA, Swarm is a platform for ABMs that includes a conceptual framework for designing, describing and conducting experiments on ABMs; software implementing that framework and providing many handy tools; and a community of users and developers that share ideas, software and experience. Zheng *et al.* (2016) propose a particle swarm optimization-based combinational auction model and its swarm-based simulation to solve multi-dispatching problem. Swarm supports scientists by standard software tools named Alla and does not put any restraints on models and interactions. Because of this architecture and written by Objective C and Java, its running speed is a real challenge toward crowd network simulations.

Developed by Argonne National Laboratory, Repast (Recursive Porous Agent Simulation Toolkit) is a multi-agent modeling tool that integrates 11 class libraries which include hundreds of classes to generate, perform, display and collect simulation data. Sun and Zhong (2017) propose a rumor propagation model considering rumor acceptability function and simulate rumor propagation on complex networks with Repast simulation platform. As modeling of Repast is more like to design a state machine, the whole state of crowd network cannot be easily described.

As a joint effort between George Mason University's Evolutionary Computation Laboratory and the GMU Center for Social Complexity, MASON (multi-agent simulator of neighborhoods or network) is a fast discrete-event multi-agent simulation library core in Java, designed to be the foundation for large custom-purpose Java simulations and also to provide more than enough functionality for many lightweight simulation needs. MASON contains both a model library and an optional suite of visualization tools in 2D and 3D. Wendt and Julien (2016) introduce Mason to support modular contextual reasoning development by handling low-level sensor routing and abstracting data sources as composable and functionally reactive data streams. But it still needs further development to be a mature simulation tool (Chen *et al.*, 2011).

Developed by CCL (center for connected learning and computer-based modeling) of Northwestern University, NetLogo is a programable modeling environment for simulating natural and social phenomena, which is particularly well-suited for modeling complex systems developing over time. Modelers can give instructions to hundreds or thousands of "agents" all operating independently. This makes it possible to explore the connection between the micro-level behavior of individuals and the macro-level patterns that emerge from their interactions. Kponyo *et al.* (2016) propose a DITS (distributed intelligent traffic system) which uses ACO (ant colony optimization) to solve the traffic problem, and implement it in NetLogo and simulated while studying traffic factors such as average travel speed, average waiting time of cars and the number of stopped cars in queue. NetLogo improves Logo by supporting multi-individuals but single one, but it does not provide any simulation clocks and event handling mechanism (Sklar, 2007).

HLA is general purpose architecture for distributed computer simulation systems. Its early development was sponsored by US Defense Modeling and Simulation Office. In 2000, it was adopted by IEEE as an international standard IEEE 1516 [Simulation Interoperability Standards Committee(SISC) of the IEEE Computer Society, 2000]. As federates exist within a federation in the form of data abstraction, federated integration keeps the independency of its participants well. This kind of integration is more suitable for and is widely used in distributed and loosely coupled simulation integrations. Federation only defines interesting domains for given objectives and rules for inter operations, so it is a real loosely coupled integration solution. Within a federation, subsystems collaborate in an indirect way so that the context of interoperation can be considered.

3. High-level architecture-based simulation framework

Although HLA satisfied the diversification requirement and dynamic requirement well to some extent, while unfortunately HLA only fit requirements of simulation realm mostly in military field, and every professional field has its own characteristics, that is to say, when adopting HLA in other research fields, it still needs more efforts to meet new challenges. First, development of HLA federations is *ad hoc*. simulation object model and federation object model are used to define interactions among federates and simulation domains. In advancement of simulation executions, simulation object model and federation object model are fixed, and interactions among objects and federates are all set before simulation execution starts. A given HLA federation only satisfied given long-term stable cooperation. When environment changes, a new one must be developed. This is not convenient to dynamic collaborations. Second, many contradictions are solved by agreements outside information systems, so these agreements are not guaranteed by software or workflow. Third, when it comes to crowd network, systemic emergence is a key characteristic of complex systems, and HLA do not support this kind of simulations. Last but not the least important, scale of HLA simulations is thousand level at the most, but the scale of crowd network simulations often exceed million or trillion level, which leads to extremely huge burden for message communications. To reduce communication load of HLA simulations, HLA adopts publish–subscription mechanism to perform communication among federates but broadcast. For example, supposing federate publishes/subscribes information only 8 bytes per time, in TCP protocol this message will be packed to a packet with 40-byte long. Under this circumstance, network communication burden can be shown by Table I.

Most HLA/RTI owes a central control node, such as RTIG (RunTime Infrastructure Gateway), which means all messages exchanged in simulation executions are all forwarded by this central node. When it comes to large scale simulations, this central node has already been a bottleneck of network information exchange. This paper proposes a reflective memory-based simulation framework with reference to HLA/RTI.

4. A reflective memory-based simulation framework for crowd network

To settle down above problems, this paper proposes a reflective memory-based simulation framework (Figure 1). This framework is particularly suitable for crowd network simulations with features of dynamic, diversification and mass member features.

Reflective memory is a means to share common data between different and independent systems deterministically. Reflective memory networks are real-time local area networks where each device or computer always has a local up-to-date copy of the share data set. These networks are designed for highly deterministic data communications delivering tightly timed performance required on distributed control systems or simulations (Baek, 2002).

Publish–subscription chain	Packet length	Network burden
10^4 Messages	4×10^5Byte	400KB
10^6 Messages	4×10^7Byte	40MB
10^8 Messages	4×10^9Byte	4GB

Table I. Relationship between message scale and network burden

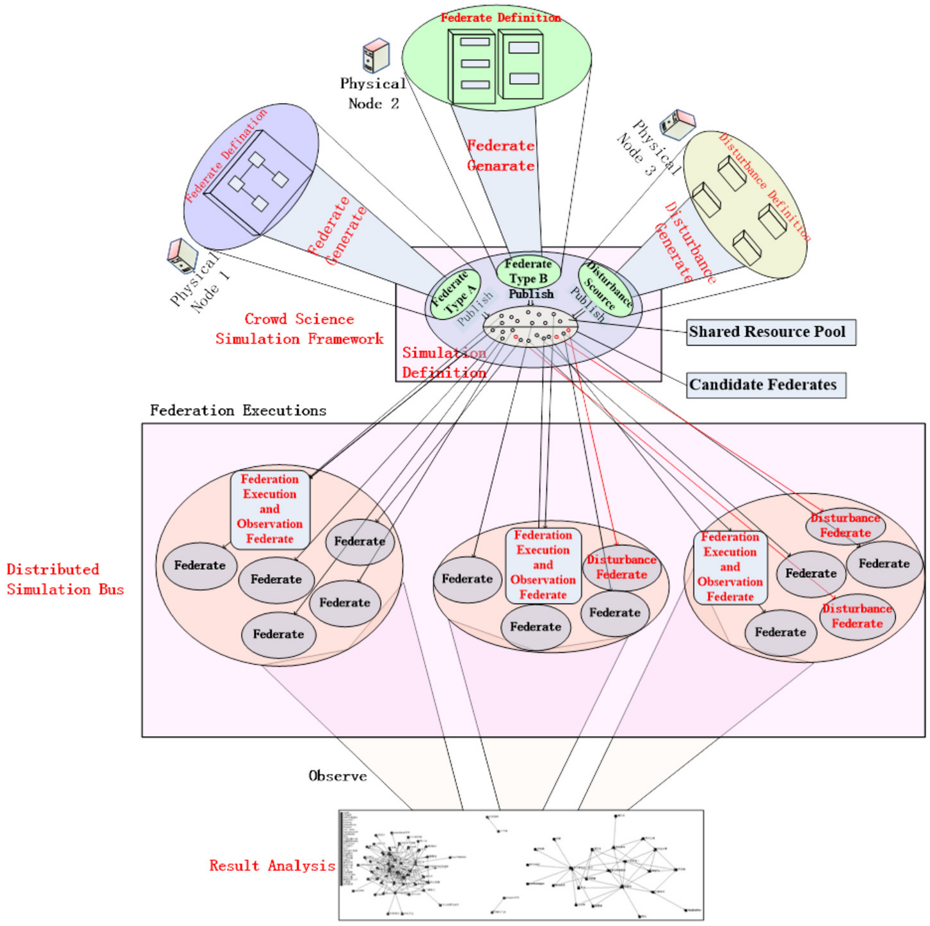

Figure 1.
Crowd network simulation framework

To solve the fixed federation development problem, this paper proposes a two-level federation-based simulation framework, system-level simulation (system federation) and application-level simulation (application federations). System federation defines semantic environment of physical aspects, and application federations define semantic environment of logical ones. The cooperation individuals of system federation are projected from real systems which are intend to collaborate together, and the meta-model of cooperative individuals which will participate in application federation is also defined in system applications. Collaborations in system federation are relatively simple and monotonous, that is publishing sharable resources and candidate application federates. After simulation definition, these candidate ones may join in an application federation to be a real application federate. When semantic environment defines, semantic rules and sharable resources are all settled.

To solve the outside agreement problem, simulation definition is introduced in this framework. Simulation definition specifies condition-, simulation- and interaction-related parameters. Condition-related parameters give initial environment of simulations, such as population scale, sampling scale and population distribution. Simulation-related parameters define how simulation can be performed, such as simulation generation, nature selection strength, multiplier of return, cooperation cost, contribution rate of cooperation, mutation rate, punishment cost, punishment strength, disturbance injection time and disturbance strength.

To address the systemic emergence and scale problem, share resources pool is used in this framework. Share resources pool stores different types of candidate initial population for future application federation use. When application federation executes, all resources in share resource pool can be accessed by federates that participate in given application federation executions.

Development of this framework can be divided into three stages: preparation stage, execution stage and analysis stage (Figure 2).

Crowd network simulation execution starts from federates definition tools to generate digital self-models and interaction models. And simulation definition tools define execution parameters and structure of simulation domain.

Simulation execution stage performs execution of simulation domains. Driven by Distributed Simulation Bus, federates deduce behaviors and states evolution of digital selves. Simulation Execution and Observation Federate observes and controls execution states of other federates, simulation bus and simulation domain. Disturbance Federates produce disturbance parameters and then these federates impose them to digital self-federates by simulation bus.

Analysis stage mainly evaluates simulation results, which analyzes simulation result data by evaluation models to verify digital self-models, evolution mechanism and robustness conclusions about crowd networks.

5. Implementation architecture

To apply this framework, an implementation architecture is introduced (Figure 3) in this paper. In this architecture, crowd network simulation platform is composed by federate definition tools, simulation definition tools, simulation execution and observation tools, simulation result evaluation tools and distributed simulation bus. And disturbance federates are special federates to research evolution of digital selves that are constructed by disturbance models and effect mechanism. Digital self-federates are applications which contain simulation objects of digital selves. They define and perform interactions among digital selves by federate definition tools and simulate evolution of digital selves driven by simulation bus.

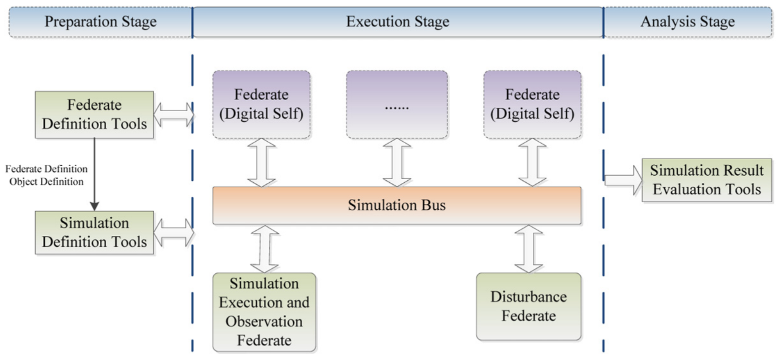

Figure 2. Crowd network simulation development process

5.1 Distributed interaction simulation bus

Most HLA/RTI adopt a central control node, which means all messages exchanged in simulation executions are all forwarded by this central node. Obviously, this architecture can greatly improve the efficiency of interactive message management. While when it comes to large scale simulations, this central node has already been a bottleneck of network information exchange and greatly influent efficiency and credibility of given simulation.

To resolve the network burden problem, simulation bus is mostly working on reflective memory (Figure 4) to make full use of its real-time data update feature.

The whole reflective memory can be divided into three parts: global area control block (GACB), simulation global area (SGA) and federate global area. GACB stores basic information of given simulation, such as simulation tag, simulation generation, nature selection strength, multiplier of return, cooperation cost, contribution rate of cooperation, mutation rate, punishment cost, punishment strength, disturbance injection time and disturbance strength and federate numbers, which are all set by simulation definition tools and can be accessed by disturbance federate and be read by all federates. SGA mainly sustains semaphores for mutual exclusion and synchronizations, which are initialed by simulation definition tools and can be accessed by all federates. LGA keeps some attribute of federate objects and interactions at given time which can be accessed by corresponding federate. All these memory areas are managed by simulation bus, whose structure is described by Figure 5.

Simulation domain management arranges simulation domains by providing functions like create, join, quit and destroy given simulation execution.

Declaration management sustains relations of publish–subscription chain among object classes and interaction classes, which includes registration and matching of these relations.

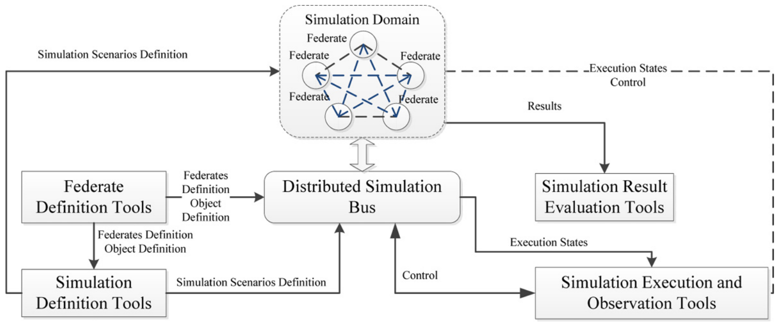
Simulation Domain
Federate
Federate
Federate
Federate
Federate
Simulation Scenarios Definition
Execution States
Control
Results
Federate Definition Tools
Federates Definition Object Definition
Distributed Simulation Bus
Simulation Result Evaluation Tools
Federates Definition Object Definition
Execution States
Control
Simulation Definition Tools
Simulation Scenarios Definition
Simulation Execution and Observation Tools

Figure 3.
An implementation architecture

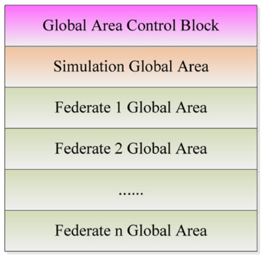

Figure 4.
Reflective memory

Object management registers and destroys instances of object classes and interaction classes. At the same time, object management adjusts information interactions dynamically among federates according to requirements of simulation advancement.

Synchronization management controls advancement of federates harmoniously. The advancement strategy includes time advancement, synchronization point advancement and generation advancement.

Ownership management transfers ownership of simulation objects or object attributes. And only given federate who owes ownership of given attribute or object can update its value to avoid conflict caused by multi-source updating.

Interaction/Perception management uses additional conditions such as value of interests, range of interests and area of interests to further exactly match information publish and subscription.

Memory reflection is a function provided by reflective memory card, which can guarantee that all memory copies on reflective memory card within given local area network are updated to be the same in real-time. Thus, a group of reflective memory cards within given local area network can be deemed as a piece of share memory for corresponding computers.

5.2 Federate definition tools

According to network, mental models federate definitions tools solve problems like how to define and generate federates and how to generate its holographic simulation models. To achieve this target, main functions of federate definitions tools are digital self-definition, federate behavior definition and interface definition, and based on them, these tools also form code framework and interface descriptions of corresponding federates.

Digital self-definition integrates instances of digital selves, defines static attributes of federates and controls instances of digital selves during simulation executions according to requirements of given simulation.

Federate behavior definition generates and maintains agent-based federate behavior definition templates and establishes various digital self-behaviors according to these templates.

Based on federate behavior definition, interface definition specifies federate interfaces according to interface requirements of distributed interactive simulation bus, which include publish and subscription information about object and interaction classes. To facilitate interface definitions, a set of interface templates are also provided.

5.3 Simulation definition tools

Simulation definition tools describe simulation domains and information distributions. Its functions include federate execution parameter definition, simulation domain behavior definition and federate information distribution definition.

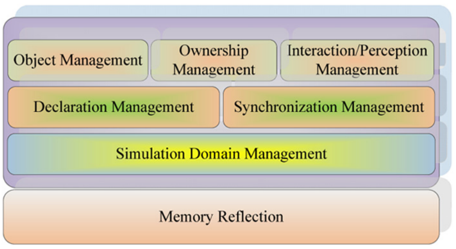
Object Management
Ownership Management
Interaction/Perception Management
Declaration Management
Synchronization Management
Simulation Domain Management
Memory Reflection

Figure 5.
Simulation bus

Module federate execution parameter definition sets parameters like federates initial parameters, simulation synchronization parameters and simulation terminate parameters.

Simulation domain behavior definition specifies sub domains, simulation bus deployment and disturbance behavior.

Federate information distribution definition describes federate resource requirements for future simulation executions, which include applications, interface templates, configuration files and federate distributions. Then, based on these definitions, federates are distributed to assigned physical nodes.

5.4 Simulation execution and observation tools

Simulation execution and observation tools control simulation executions and monitor execution states.

The main functions of simulation execution control include execution process control of crowd network simulations such as start, pause and terminate, interactive control of information from object and interactive classes and intervene simulation executions according to simulation targets, for example, disturbance injection.

Simulation execution state monitor captures execution state information from federates, sub-domains and simulation domains, and then visualizes them by figures, tables and texts to directly reflect execution states of monitored objectives.

5.5 Simulation result evaluation tools

Simulation result evaluation tools help users analyze simulation data to verify models, to validate achievements and to prompt research studies, which include data processing module, evaluation algorithm set module, evaluation index system module, evaluation instance management module and evaluation module.

Data processing module extracts, cleans and converts simulation data to evaluation related data according to evaluation targets and requirements.

Evaluation algorithm set module integrates evaluation algorithms such as average, variance, quadratic mean deviation, summation, maximum and minimum.

Evaluation index system module constructs tree-like evaluation index system according to evaluation targets and assigns evaluation algorithm and data of every evaluation nodes to form evaluation instances.

Evaluation instance management module manages and maintains evaluation instances. An evaluation instance can be established by data processing module, evaluation algorithm set module and evaluation index system module or be derived from existing evaluation instances.

Evaluation module performs evaluations and generates evaluation results. Driven by evaluation algorithm, data of leaf node are computed to form evaluation result of leaf node. After all leaf nodes are calculated, corresponding evaluation algorithm drives evaluation computation of their parent nodes till root node to achieve whole evaluation of given evaluation tree.

6. Summary

As main mode of modern service industry and future economy society, the research on crowd network can greatly facilitate governances of economy society and make it more efficient, humane, sustainable and at the same time avoid disorders. However, because most results cannot be observed in real world, the research of crowd network cannot follow a traditional way. Simulation is the main means to put forward related researches. Compared with other large scale interactive simulations, simulation for crowd network has challenges of dynamic, diversification and massive participants. Fortunately, known as the most famous and widely accepted standard, HLA has been widely used in large-scale simulations. But when it comes to crowd network, HLA has shortcomings like fixed federation, limited scale and agreement outside the software system. With reference to HLA, this paper proposes a novel reflective memory-based framework toward crowd network simulations. The proposed framework adopts a two-level federation-based architecture, system level simulation (system federation) and application level simulation (application federations), which separates simulation related environments into physical and logical aspect to enhance the flexibility of simulations. Simulation definition is introduced in this architecture to resolve the problem of outside agreements and share resources pool (constructed by reflective memory) is used to address the systemic emergence and scale problem.

Comparing with existing simulation methods, the proposed framework enjoys several sound improvements:

- Simulation syntax and semantic are all settled under this framework by templates, especially interface templates.
- As simulations are separated by two-level federations, physical and logical simulation environment are considered separately.
- The definition of simulation execution is flexible. When developing new simulations, recompile is not necessary, which can acquire much more reusability.
- As reflective memory is adopted as share memory within given simulation execution in this framework, population can be perceived by all federates, which greatly enhances the scalability of this kind of simulations.
- Finally, important, communication efficiency and capability has greatly improved by this share memory-based framework.

References

Baek, I.J. (2002), "A survey on reflective memory systems", *Proceedings on the 15th CISL Winter Workshop*, *Kushu*.

Chai, Y., Miao, C., Sun, B., Zheng, Y. and Li, Q. (2017), "Crowd science and engineering: concept and research framework", *International Journal of Crowd Science*, Vol. 1 No. 1, pp. 2-8.

Chen, Y., Dong, Y. and Deng, L. (2011), "Comparison of agent-based simulation platforms", *Journal of System Simulation (in Chinese)*, Vol. 23 No. 1, pp. 110-116.

Kponyo, J.J., Nwizege, K.S., Opare, K.A., Ahmed, A.-R., Hamdoun, H., Akazua, L.O., Alshehri, S. and Frank, H. (2016), "A distributed intelligent traffic system using ant colony optimization: a netlogo modeling approach", *International Conference on Systems Informatics, Modelling and Simulation (SIMS)*, *Riga, Latvia*, 1-3 June, pp. 11-17.

Michelucci, P. and Dickinson, J.L. (2016), "The power of crowds", *Science (New York, NY)*, Vol. 351 No. 6268, pp. 32-33.

Simulation Interoperability Standards Committee (SISC) of the IEEE Computer Society (2000), *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Framework and Rules*, The Institute of Electrical and Electronics Engineers, New York, NY.

Sklar, E. (2007), "NetLogo, a multi-agent simulation environment", *Artificial Life*, Vol. 13 No. 3, pp. 303-311.

Sun, R. and Zhong, Y. (2017), "Modelling and simulation for rumor propagation on complex networks with repast simulation platform", *4th International Conference on Information Science and Control Engineering (ICISCE)*, *Changsha*, 21-23 July, pp. 104-1018.

Wallis, L. and Paich, M. (2017), "Integrating Artificial Intelligence with AnyLogic Simulation", *Procedings of the 2017 Winter Simulation Conference*, *Las Vegas, NV*, 3-6 December, p. 4449.

Wendt, N. and Julien, C. (2016), "Mason: an open development contextual sensing framework enabling reactive applications", *IEEE/ACM International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, *Austin, TX*, 16-17 May, pp. 100-110

Zheng, J., Dong, J., Guan, Z. and Zhang, P. (2016), "PSO based combinational auction model and its swarm based simulation", *Systems Engineering – Theory & Practice (in Chinese)*, Vol. 36 No. 12, pp. 3142-3151.

About the authors

Sun HongBo became a Member (M) of IEEE in 2010. Hongbo was born in Fuping, Shannxi Province, China on February 13, 1977. In 2011, he got his doctor's degree in control science from Tsinghua University, Beijing, China. In 2005, he got his master's degree in software engineering from Tsinghua University, and in 1998, he got his bachelor's degree in information science from Beijing Institute of Technology, China. He is a Lecturer of Yantai University, Shandong, China now. He has been a Postdoctoral Researcher in Department of Automation, Tsinghua University, Beijing, China from July 2011 to June 2014. From November 2009 to November 2010, he has worked in National Research Council Canada as an International Visiting Worker. Between 2005 and 2006, he served as a Software Engineer of National CIMS ERC, Tsinghua University, Beijing, China. And during 1998 to 2002, he was an Assistant Professor of Shenyang Institute of Technology, Liaoning, China. His research interests include system integration, artificial intelligence, algorithm, large-scale simulation and e-commerce. Sun Hongbo is the corresponding author and can be contacted at: hsun@ytu.edu.cn

Mi Zhang was born in Dezhou, Shandong Province, China on September 26, 1992. In 2016, she got her bachelor's degree in computer science and technology from Yaitai University, Shangdong, China. She is a master's student in Yantai University.