A Framework to Support Decision-making based on AI and Simulation of Large-scale Models

Unai Arronategui, José Ángel Bañares, and José Manuel Colom

Aragón Institute of Engineering Research (I3A)
University of Zaragoza, Zaragoza, Spain

Abstract. Big data collection and analysis is used in industry and public organizations to support decision-making. However, simulation as a core technology to support optimization, or the exploration of large state spaces in artificial intelligence have serious difficulties for industrial adoption. Our approach to solve these difficulties is the adoption of a modelling methodology supported by a cohesive framework based on the Petri net formalism for efficient simulation of complex discrete event systems over large computational infrastructures.

Keywords: Distributed Simulation · Petri Nets · Artificial Intelligence· Decision-making.

1 Introduction

The growing interest in large-scale distributed simulations (DS) is motivated by the increasing scale, resolution, and complexity of systems to be studied. Many optimization problems in science and industry involve time-consuming simulations and expensive objective function. The integration of computer simulation with artificial intelligence (AI) is based on the advances in Modelling and Simulation (M&S) and the increase of computational power. However, the scalability of these approaches is limited by the need to fit the model in a computer. Conventional simulation practice on optimization problems typically involves developing a single model and performing simulation experiments in sequence on a single computer. The most common approach is the simplification of the model to fit it to a computer due to the difficulties of transferring it to a distributed simulation using a cluster of computers. Recent work has focused their interest on optimization techniques combined with simulation [11, 10, 3].

Our approach to solve these difficulties is the adoption of a modelling methodology for efficient simulation of complex discrete event systems (DES) over large computational infrastructures [2, 1]. This methodology is supported by a cohesive framework based on the Petri net (PN) formalisms, which collects languages (high level specifications language, code for efficient DS), tools, services (efficient interpreters, compilers, models and scripts to generate code, to automate configuration, deployment and supervision of simulations on premise, public and hybrid clouds with heterogeneous resources), and mechanisms (such as synchronization and dynamic load balancing mechanisms). These mechanisms and tools are the core functionality to support the exploration of large-scale models, requiring strategies implementing different policies to automatize the exploration of large scale space states to support decision-making by the exhaustive exploration and optimization of systems.

DS of large scale DES is a complex problem whose current challenges such as scalability, dynamic configurations, and the integration of different formalisms prevent its adoption by the industry. This work presents the potential of the synergistic integration of AI and simulation.

2 Scalable Compilation, Dynamic Configuration & Federation of Hybrid Simulations

We proposed the integration of AI with the simulation of DES in the context of our developed framework for the M&S of large-scale models. The main characteristic of the previous work has been the adoption of PN as a well-known formalism for modelling complex systems and the proposal of a representation of PNs suitable for efficient distributed simulation. The characteristics of the proposed representation allow facing the most important challenges of the M&S of large systems: scalability, dynamic configuration and interoperability [2].

Instead of the direct emulation of high-level conceptual models, we transform the original structured model into a flat Place/Transition net model. Then, the compilation stage transforms a flat Place/Transition net into an efficient representation for DS. The model compilation bottleneck is the size of code generated limited by the memory of the computer. Scaling the size of the models in a DS requires the generation of large-scale models of arbitrary structure [4], and *modular compilations* based on components that can be compiled separately, deployed at different nodes, and linked with other distributed components. With DS of large-scale systems being complex, configuration management can be a delicate undertaking to implement.

Dynamic configuration of DS is also essential to modify a simulation while it is running. The high coupling of code partitions to be simulated requires that deployment of the code on the execution infrastructure must be sensitive to the model complexity, communications with neighbouring nodes, heterogeneous resource physical characteristics such as processor speed, and the evolution of simulation.

Finally, the complexity of the system must be faced by combining different formalisms to represent all the aspects involved. The adoption of standards such as the High Level Architecture (HLA) solves this problem from the point of view of data interoperability. However, the integration of discrete event simulators with other formalisms, such as continuous specifications, remains a challenge. The compilation of PN specifications produces an event dependency network. This representation supports the production and consumption of events, and facilitates the integration of components in an event-driven architecture.

3 Artificial Intelligence and Simulation

AI today provides a set of mechanisms and tools than can be integrated into the simulation of DES of huge dimensions and intricate structure. The motivation is twofold: On the one hand, searching is an integral part of AI, and a search problem requires the modelling of the problem as a DES. Therefore, for complex systems, AI is faced with the modelling and exploration of large state spaces. On the other hand, the model partition for a DS is a complex optimization problem. Heuristic such as simulating annealing, or genetic algorithms have been proposed to define an initial partition, and reinforcement learning for a dynamic partition considering the evolutionary nature of the simulation [9].

Search is an integral and ubiquitous part of AI, and different meta-heuristics has been proposed to guide the search process that involves the exploration of large state spaces [6]. In this sense, AI can be seen as a basic tool for decision-making by searching for the best sequence of actions or configuration to optimize a function. The integration of DS with meta-heuristic can help afford the complexity of scheduling activities, planning a production, or managing resources for complex systems such as those emerging in Cyber-physical systems, Industry 4.0, Digital Twins and Smart environments.

However, the applicability of search-based approaches are often limited by the state-space explosion problem. Of special interest is the integration of DS with meta-heuristic methods that are easily parallelizable [8]. Three modes of DS have been identified to speed-up the exploration of the state-space [11]: 1) To exploit the parallelism of a single simulation; 2) to build and simulate large models by federating simulators, reducing the cost of developing new models; and 3) running simultaneously the same model with different parameters.

The use of DS for single-state methods allow the exploration of large-scale models, and population methods are most readily parallelizable, since they already can simultaneously explore candidate solutions. Even cutting-edge techniques such Deep Reinforcement Learning (DRL), are often limited by its excessive training time dues to the state-space explosion problem [7, 5].

On the other direction, regarding the application of AI to DS, it is necessary to design intelligent algorithms that automate the process of looking for the best system configuration. It includes the use of AI to manage the complexity of the simulation task itself in its different phases: model partition and deployment; resource allocation; tuning the distributed simulation in runtime; extraction, storage and compression of the simulation results; and exploitation of the results and traces of the simulation.

The main idea is the use of meta-heuristics such as simulating annealing or genetic algorithms for the definition of the initial partitioning, and DRL for a dynamic model partitioning. It implies a clear definition of the state in each simulation engine and their neighbours, and the actions available to improve the configuration. The work will focus on evaluating metrics for use with the dynamic load balancing mechanism in distributed simulations.

4 Future work

In our previous work we have developed a PN based framework for M&S of large scale and complex systems. Previous works has described a PN based modular language, the translation of the PN to a representation for efficient distributed simulation, the algorithms for distributed simulation and load-balancing mechanism. Future work will focus in the incorporation of AI in two directions: to adapt meta-heuristics algorithms to take advantage of distributed simulation of large systems; and the incorporation of AI to take decisions on runtime using developed mechanisms to obtain the best configuration.

Acknowledgments This work was co-financed by the Aragonese Government and the European Regional Development Fund "Construyendo Europa desde Aragón" (COSMOS research group); and by the Spanish program "Programa estatal del Generación de Conocimiento y Fortalecimiento Científico y Tecnológico del Sistema de I+D+i ", project PGC2018-099815-B-100.

References

1. Arronategui, U., Bañares, J.Á., Colom, J.M.: A MDE approach for modelling and distributed simulation of health systems. In: GECON 2020 - International Conference on the Economics of Grids, Clouds, Systems, and Services. pp. 89–103. Springer (2020)
2. Bañares, J.Á., Colom, J.M.: Model and simulation engines for distributed simulation of discrete event systems. In: GECON 2018 - International Conference on the Economics of Grids, Clouds, Systems, and Services. pp. 77–91. Springer (2018)
3. Bartz-Beielstein, T., Filipic, B., Korosec, P., Talbi, E. (eds.): High-Performance Simulation-Based Optimization, Studies in Computational Intelligence, vol. 833. Springer (2020)
4. Bergero, F., Kofman, E.: A vectorial devs extension for large scale system modeling and parallel simulation. SIMULATION 90(5), 522–546 (2014)
5. Capocchi, L., Santucci, J.F.: Discrete event modeling and simulation for reinforcement learning system design. Information 13(3) (2022)
6. Hussain, K., Salleh, M.N.M., Cheng, S., Shi, Y.: Metaheuristic research: a comprehensive survey. Artif. Intell. Rev. 52(4), 2191–2233 (2019)
7. Körber, M., Lange, J., Rediske, S., Steinmann, S., Glück, R.: Comparing popular simulation environments in the scope of robotics and reinforcement learning (2021), https://arxiv.org/abs/2103.04616
8. Luke, S.: Essentials of Metaheuristics. second edn. (2013), available for free at http://cs.gmu.edu/∼sean/book/metaheuristics/
9. Meraji, S., Tropper, C.: Optimizing techniques for parallel digital logic simulation. IEEE Transactions on Parallel and Distributed Systems 23(6), 1135–1146 (2012)
10. Rabe, M., Deininger, M., Juan, A.A.: Speeding up computational times in simheuristics combining genetic algorithms with discrete-event simulation. Simul. Model. Pract. Theory 103, 102089 (2020)
11. Taylor, S.J.: Distributed simulation: state-of-the-art and potential for operational research. European Journal of Operational Research 273(1), 1–19 (2019)