Multi-agent deep reinforcement learning concept for mobile cyber-physical systems control

Vyacheslav Petrenko*, and Mikhail Gurchinskiy

Institute of Mathematics and Information Technologies named after Professor N.I. Chervyakov, North Caucasus Federal University, 355017 Stavropol, Russian Federation

Abstract. High complexity of mobile cyber physical systems (MCPS) dynamics makes it difficult to apply classical methods to optimize the MCPS agent management policy. In this regard, the use of intelligent control methods, in particular, with the help of artificial neural networks (ANN) and multi-agent deep reinforcement learning (MDRL), is gaining relevance. In practice, the application of MDRL in MCPS faces the following problems: 1) existing MDRL methods have low scalability; 2) the inference of the used ANNs has high computational complexity; 3) MCPS trained using existing methods have low functional safety. To solve these problems, we propose the concept of a new MDRL method based on the existing MADDPG method. Within the framework of the concept, it is proposed: 1) to increase the scalability of MDRL by using information not about all other MCPS agents, but only about n nearest neighbors; 2) reduce the computational complexity of ANN inference by using a sparse ANN structure; 3) to increase the functional safety of trained MCPS by using a training set with uneven distribution of states. The proposed concept is expected to help address the challenges of applying MDRL to MCPS. To confirm this, it is planned to conduct experimental studies.

1 Introduction

With the advancement of deep reinforcement learning methods, more and more complex problems are falling into the area of interest of researchers. One of the newest and actively developing areas of deep reinforcement learning is multi-agent deep reinforcement learning (MDRL) of multi-agent systems (MAS). The interest in MAS is due to the higher efficiency of decentralized problem solving using MAS in comparison with similar centralized methods [1–3]. MAS are used in such areas [4] as manufacturing [5], transportation, smart homes, robotics [6–10], aviation, infrastructure, medicine [11], etc. An important problem in the development of MAS is the ambiguity and non-triviality of the synthesis of behavior policy of individual agents according to the specified MAS targets. At the same time, single-agent deep reinforcement learning (SDRL) has established itself as a powerful and versatile tool for solving intellectual problems at a level comparable to that of a human [12–14]. Taken together, these factors determine the relevance of the development of SDRL for use in MAS in the form of MDRL.

* Corresponding author: vip.petrenko@gmail.com

The two main challenges for MDRL are learning convergence and scalability [15]. The scalability problem also includes the problem of changing the number of agents during the MAS operation. This problem is of particular importance in the context of such a type of MAS as mobile cyber physical systems (MCPS). MCPS, such as groups of UAVs [16] or mobile robots, in comparison with other types of MAS are characterized by functioning in a non-deterministic environment with a relatively high probability of failure of individual agents, instability of the bandwidth and topology of communication channels [17], etc. MCPS also has higher requirements for the reliability of control methods [17, 18]. An error in the process of calculations [20, 21] can lead to damage and failure of MCPS agents.

MCPS also features: complex dynamics of functioning; a large number of possible states of the system; high unit cost of agents' computing power. MDRL is an artificial intelligence method based on the use of ANN, the features of which are: difficulties in interpreting learning outcomes; high computational complexity; poor scalability.

The listed features of MCPS and MDRL have led to the emergence of two architectures of MCPS control systems using MDRL:

- 1) with the calculation of ANN inference on a computing system external to the agent;
- 2) with the calculation of ANN inference on the on-board computing systems of agents.

In an inference-computing architecture on an external computing system, agents are equipped with a microcontroller and a simple multi-core GPU. Inference calculation is performed on an external high-performance server for all agents at once. With this architecture, it becomes critical that agents communicate with the base station.

In an inference-computing architecture, agents are equipped with high-performance graphics processing units (GPUs) such as Nvidia GTX 1080 or Nvidia TX2. Calculation of inference using onboard equipment makes it possible for agents to work autonomously in the absence of communication with the base station. The high cost of the GPUs used leads to a significantly higher cost of individual agents and the entire MCPS.

Thus, the following issues arise when applying MDRL to MCPS:

1. Poor scalability of existing methods leads to a large amount of computational resources, a long time and / or high cost of the MDRL process with an increase in the number of MCPS agents.
2. High computational complexity of the ANN inference used to control MCPS agents requires an increase in the computational power of the agents, which leads to a significant increase in their cost.
3. Low functional safety of trained MCPS leads to a significant limitation of their areas of use.

To solve the above problems, the article proposes the concept of a new MDRL method based on the MADDPG method. The article is structured as follows. Section 2 reviews the existing MDRL methods and problems, substantiates the use of the MADDPG method as a prototype, and discusses effective solutions applied in areas adjacent to MDRL. Section 3 analyzes the problem areas of the MADDPG method and describes the proposed concept for its improvement.

2 Literature review

According to the approximation object, MDRL methods can be divided into value-based and policy gradient. According to the conditions of training, MDRL methods are divided into collaborative and independent.

In value-based methods, the object of approximation is the value function; in gradient methods, the object of approximation is the policy of decision making with or without a value function. Value-based methods [22–29] are represented mainly by methods based on the

DQN (deep Q-network) learning algorithm [12] and its modification DRQN (deep recurrent Q-network) [30]. The main disadvantages of value methods are:

- relatively low training efficiency [31, 32];
- focus on working with a discrete space of actions.

The gradient method of independent learning is the FTW method [33]. The FTW method is an adaptation of the single-agent deep reinforcement learning method based on the actor-learner IMPALA structure [34]. It was shown in [35] that adapted one-agent gradient methods in a multi-agent environment have weak convergence. This phenomenon is due to the fact that the reward r_i received by the i -th agent depends not only on his action a_i but also on the actions of other agents $\{a_j | j \neq i\}$; therefore, the probability of obtaining the correct direction of the gradient decreases exponentially with an increase in the number of agents. To solve this problem, the work [35] proposed a gradient method of joint learning MADDPG (Multi-Agent Deep Deterministic Policy Gradient). It was shown in [35] that the MADDPG method is more efficient than the independent learning methods based on the DQN, DDPG, and TRPO algorithms. Gradient collaborative learning methods have the following disadvantages:

- the need to ensure the equality of the number of agents at the learning stage N and the number of agents at the stage of functioning K ;
- the complexity of scaling due to nonlinearly increasing computational complexity of the learning process with an increase in the number of agents.

The value-based MDRL methods are applicable only for tasks with discrete set of actions. Gradient-based MDRL co-learning methods are applicable to continuous-action tasks and have good convergence, but poor scalability. Gradient MDRL methods with independent learning of agents have good scalability, however, they have problems with overcoming the nonstationarity of the environment [15].

The listed features of the MDRL gradient methods with joint and independent training of agents determine the relevance of the development of hybrid MDRL gradient methods, which simultaneously have high convergence and scalability. In this paper, we propose such a hybrid method based on the gradient method of joint learning MADDPG and the concept of parameter separation [23]. The concept of parameter separation is to use the same ANN to transform signals from different agents.

Despite the practical results obtained with ANN, the state of the ANN in the learning process remains uninterpretable due to the large number of elements and complex structure. Understanding how ANN learns remains one of the central problems in machine learning [36]. This problem complicates the choice of the optimal configuration and structure of the ANN. In practice, the ANN configuration is usually determined empirically. Some initial value is selected based on expert assessments, then training is performed. If specific signs are found, the current configuration is deemed unsuitable and changed as necessary [37]. There are methods for the automated synthesis of not only the configuration, but also the architecture of the ANN [38], but these methods require large computational costs at the training stage.

ANN structure optimization methods have become widespread in such a form of machine learning as supervised learning. An urgent task is to study their applicability and effectiveness when used in MDRL. A review of various methods for optimizing the ANN structure is presented in [39]. Also in [39] the difficulties of direct comparison of the set of existing methods with each other are shown. In this article, for application in the concept, the SET method for optimizing the ANN structure, proposed in [40], is considered. The essence of the method is to use an evolutionary algorithm to dynamically change the ANN structure during training, instead of using an unchanged predefined ANN structure. To do this, instead of a fully connected ANN structure, a sparse one is used. Due to the sparseness, the used ANN structure has the ability to change (evolution), requires storing fewer weights in

memory and performing fewer operations. During the learning process, the ANN structure is periodically modified, during which some of the links with weights close to zero are removed, and new ones are randomly created instead. It is shown in the work that, despite the simplification of the ANN structure, its performance remains at the same level as compared to the fully connected ANN, and in some cases even has a greater convergence of training.

In contrast to such a field of deep learning as pattern recognition, which uses a large number of different architectures [41], deep reinforcement learning methods mainly use feedforward neural networks [42] and recurrent ANNs [43]. In this paper we consider only feedforward ANNs.

The need to comply with the requirements for functional safety led to the emergence of a class of methods of “safe” reinforcement learning. This area of research was reviewed in [44]. Safe reinforcement learning can be defined as a decision policy optimization process that maximizes the average value of reward in tasks in which it is important to ensure reasonable system performance and/or adherence to security constraints during learning and/or functioning processes. The functional security of the MDRL results considered in the work can be considered as a kind of safe training, aimed at observing security constraints during the functioning of the trained system. In [44], a fairly large number of secure single-agent reinforcement learning (SRL) methods are given, while secure MDRL remains a poorly researched area [25]. In this paper, we propose solutions that, according to the classification in [44], belong to the class of methods for studying the space of states directed by risk.

3 Materials and methods

A schematic diagram of the MDRL process model for a cooperative MCPS with a shared reward is shown in Figure 1. The MDRL process includes two conditional, continuously alternating stages, indicated by dashed lines: the learning stage and the optimization stage of agent management policies. The rectangles indicate the entities that transform the information, the lines indicate the transmitted information. Each i -th MCPS agent has its own control policy μ_i that transforms the observed state s of the MCPS operation environment into action a_i taken by the i -th agent. In the case of MDRL, each μ_i policy is approximated by an actor ANN. Let us denote the set of all agent control policies μ_i as MCPS control policy μ . Within the functioning stage (Fig. 1), MCPS agents observe the state of the environment s and take an aggregate action $a = \{a_i | i = \overline{1, n}\}$, where n is the number of agents in MCPS. The environment moves to the next state, and the agents receive a reward r determined by the reward function. The reward function simulates the purpose of the MCPS in MDRL. The experience gained at the stage of functioning in the form of tuples (s, a, r, s') , where s' is the next state of the environment, is written into the training sample, which is then used at the stage of optimizing agent management policies.

The implementation of agent management policies optimization stage is determined by the considered MDRL methods. In the MADDPG method, an additional critic ANN is used to assess the effectiveness of agents' actions (Fig. 1). Within each step of optimizing agent management policies, the learning algorithm extracts minibatch b from the training sample, on the basis of which it optimizes the weights θ_Q of the critic ANN, and then the weights θ_μ of the actor ANN group.

The computational complexity of the MDRL process increases with scaling due to the increasing complexity of the ANNs used (Fig. 2). The information about the status of all MCPS agents is provided to the inputs of actor and critic ANNs. Accordingly, with an increase in the number of MCPS agents, the number of inputs and connections of the actor and critic ANNs increases.

Fig. 1. MDRL process model.

Fig. 2. Actor and critic ANNs inputs and outputs structure.

Within the framework of the proposed concept, it is assumed that agents do not need information about all other MCPS agents for efficient functioning, but only information about m nearest neighbors. According to this hypothesis, we propose to modify the MDRL method as follows:

1. Perform centralized training of not all MCPS agents, but only a certain group of m agents, $m < n$.
2. At the stage of functioning, supply information only about the m nearest neighbors to the input of actor ANN.

The success of using sparse ANNs to reduce the computational complexity of inference when solving visual information processing problems on mobile devices requires studying the applicability of these solutions for MDRL. The main essence of these solutions is the use of sparse ANNs and periodic optimization of their structure. Based on this, it is proposed to modify the MDRL process as follows:

1. Use sparse ANNs for the actor ANN.

2. At the stage of optimizing agent management policies, optimize not only the weights, but also the actor ANN structure.

The low functional safety of MDRL methods is due to the following factors.

1. Optimization of ANN is carried out on the basis of data sets obtained from the training sample. Generation of a training sample is performed by placing agents in some initial state and its subsequent functioning. In existing methods, the initial state is chosen with a uniform probability distribution among all possible states. An illustration of a similar approach for MCPS involving one agent is shown in Figure 3, *a*. The state of the agent marked with a cross is illustrated by its position on a two-dimensional plane. The state corresponding to the purpose of the MCPS is indicated by a white circle. A condition that is dangerous, and the achievement of which is undesirable, is indicated by a black circle. The density of the crosses in the figure reflects the probability density of the agent hitting a given point at the beginning of the episode. In Figure 3, *a*, the probability density is uniform; in Figure 3, *b*, it is higher next to the dangerous state.

Fig. 3. Approaches to initial states generation: *a*) with a uniform probability density *b*) with a non-uniform probability density.

When generating an initial state with a uniform distribution, the probability of an agent getting into a dangerous state during training is relatively small. Therefore, the experience of functioning in states close to dangerous ones will be relatively low. To solve this problem, it is proposed to generate the initial state with an uneven probability density, which is higher in states close to dangerous ones (Fig. 3, *b*). It is expected that this will enhance the “experience” of MCPS operation in near-dangerous conditions and enhance the functional safety of trained MCPS during the operational phase.

2. In the MADDPG method, training is considered complete after the value of the average reward is set at some constant value. It is proposed to use a decrease in the probability of dangerous situations to a certain threshold value as a criterion for the completion of training.

4 Conclusion

The use of MDRL methods in MCPS is fraught with a number of MCPS-specific problems:

- MDRL methods have poor scalability;
- ANN inference has high computational complexity;
- trained MCPS has low functional safety.

Based on the analysis of publications on MDRL, as well as in areas adjacent to MDRL, in this work we proposed a new concept of MDRL, which includes the following ideas:

1. To increase the scalability of MDRL, it is proposed to supply to the actor ANN input information not about all other MCPS agents, but only about m nearest neighbors.

2. To reduce the computational complexity of ANN inference, it is proposed to use sparse ANN structures, as well as to optimize them at the stage of optimization of MCPS agent management policies.

3. To improve functional safety, it is proposed to use a training sample with uneven probability density, which is higher in conditions close to dangerous.

It is expected that the proposed solutions will solve the listed problems of applying MDRL in MCPS.

References

1. G. Kovács, N. Yussupova, and D. Rizvanov, *Pollack Period.*, **12** (2017)
2. O. Darintsev and A. Migranov, *CSCMP 2019* (2019)
3. O. V. Darintsev, B. S. Yudin, A. Y. Alekseev, D. R. Bogdanov, and A. B. Migranov, *Procedia Comput. Sci.* (2019)
4. L. Wang, M. Törngren, and M. Onori, *J. Manuf. Syst.* **37** (2015)
5. R. A. Munasypov and K. A. Masalimov, *Proc. - 2017 2nd Int. Ural Conf. Meas. Ural.* 2017 (2017)
6. F. V. Bonilla Venegas, M. J. Moya Cajas, E. Anatoly Vitalyevich, L. Anatolevich, and L. E. Marín Pillajo, *Enfoque UTE*, **9** (2018)
7. A. V. Vokhmintsev, A. V. Melnikov, K. V. Mironov, and V. V. Burlutskiy, *Dokl. Math.* **99** (2019)
8. A. Bogdanov, E. Dudorov, A. Permyakov, A. Pronin, and I. Kutlubaev, *Proc. - Int. Conf. Dev. ESystems Eng. DeSE* (2019)
9. V. I. Petrenko, F. B. Tebueva, V. B. Sychkov, V. O. Antonov, and M. M. Gurchinsky, *Int. J. Mech. Eng. Technol.* **9** (2018)
10. V. I. Petrenko, F. B. Tebueva, M. M. Gurchinsky, V. O. Antonov, and J. A. Shutova, *IOP Conf. Ser. Mater. Sci. Eng.* (2018)
11. M. Bogdanov, D. Nasyrov, I. Dumchikova, and A. Samigullin (2020)
12. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, *Nature* **518**, 529 (2015)
13. V. Petrenko, F. Tebueva, V. Antonov, and M. Gurchinsky (2020)
14. V. Petrenko, F. Tebueva, N. Svistunov, and A. Pavlov (2020)
15. P. Hernandez-Leal, B. Kartal, and M. E. Taylor, *Auton. Agent. Multi. Agent. Syst.* **33**, (2019)
16. V. Pshikhopov, M. Medvedev, and T. Medvedeva, *ICCRE 2019* (2019)
17. H. Wang, H. Zhao, D. Ma, and J. Wei, *ArXiv* (2020)
18. N. Yusupova, D. Rizvanov, and D. Andrushko, *ITIDS 2020* (2020)
19. T. Fabarisov, N. Yusupova, K. Ding, A. Morozov, and K. Janschek, *Acta Polytech. Hungarica* **17** (2020)
20. E. Valiev, N. Yusupova, A. Morozov, K. Janschek, and M. Beyer (2020)
21. M. Beyer, A. Morozov, K. Ding, S. Ding, and K. Janschek, *ISSREW 2019* (2019)

22. J. Foerster, N. Nardelli, G. Farquhar, T. T. Afouras, P. H. S. H. S. Torr, P. Kohli, S. Whiteson, N. Nardell, G. Farquhar, T. T. Afouras, P. H. S. H. S. Torr, P. Kohli, and S. Whiteson, ArXiv (2017)
23. J. K. Gupta, M. Egorov, and M. Kochenderfer, Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics (2017)
24. D. Bloembergen, M. Kaisers, and K. Tuyls, Netherlands Artif. Intell. Conf. (2010)
25. S. Omidshafiei, J. Papis, C. Amato, J. P. P. How, and J. Vian, ArXiv **6** (2017)
26. Y. Zheng, Z. Meng, J. Hao, and Z. Zhang, Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics (2018)
27. Z. W. Hong, S. Y. Su, T. Y. Shann, Y. H. Chang, and C. Y. Lee, ArXiv (2017)
28. G. Palmer, K. Tuyls, D. Bloembergen, and R. Savani, AAMAS (2018)
29. L. Matignon, G. J. Laurent, and N. Le Fort-Piat, IEEE Int. Conf. Intell. Robot. Syst. (2007)
30. M. Hausknecht and P. Stone, AAAI Fall Symp. - Tech. Rep. (2015)
31. L. Matignon, G. J. Laurent, and N. Le Fort-Piat, Knowl. Eng. Rev. **27** (2012)
32. M. Tan, Mach. Learn. Proc. 1993 (1993)
33. M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castañeda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J. Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu, and T. Graepel, Science **364** (2019)
34. L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, B. Yotam, F. Vlad, H. Tim, I. Dunning, S. Legg, K. Kavukcuoglu, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, and K. Kavukcuoglu, ArXiv (2018)
35. R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, Adv. Neural Inf. Process. Syst. **2017**, 6380 (2017)
36. M. Gabella, S. Ebli, N. Afambo, and G. Spreemann, ArXiv (2019)
37. H. Pérez-Espinosa, H. Avila-George, J. Rodriguez-Jacobo, H. A. Cruz-Mendoza, J. Martínez-Miranda, and I. Espinosa-Curiel, Res. Comput. Sci. **121** (2016)
38. H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, 35th Int. Conf. Mach. Learn. ICML 2018 (2018)
39. D. Blalock, J. J. G. Ortiz, J. Frankle, and J. Gutttag, ArXiv (2020)
40. D. C. Mocanu, E. Mocanu, P. Stone, P. H. Nguyen, M. Gibescu, and A. Liotta, Nat. Commun. **9** (2018)
41. A. Canziani, E. Culurciello, and A. Paszke, Proc. - IEEE Int. Symp. Circuits Syst. (2017)
42. Y. Ma, B. Ping, G. Liu, Y. Liao, and D. Zeng, Proc. 32nd Chinese Control Decis. Conf. CCDC 2020 (2020)
43. D. Han, K. Doya, and J. Tani, Neural Networks **129** (2020)
44. J. García and F. Fernández, J. Mach. Learn. Res. **16** (2015)