UDC 004.032.26

Mykhailo Novotarskyi

ASYNCHRONOUS METHOD FOR ACTOR-CRITIC REINFORCEMENT LEARNING

Михайло Новотарський

АСИНХРОННИЙ МЕТОД НАВЧАННЯ З ПІДКРІПЛЕННЯМ ДЛЯ АКТОР-КРИТИК АРХІТЕКТУР

В роботі розглянуто метод глибокого навчання з підкріпленням, яке ґрунтується на застосуванні асинхронного підходу при використанні градієнтного спуску. На основі запропонованого методу побудовано актор-критик алгоритм, що характеризується більшим стабілізуючим ефектом при навчанні у порівнянні з існуючими паралельними методами. Крім того, запропонований підхід дозволяє розпаралелювати процес навчання шляхом застосування властивості багатоядерності сучасних комп'ютерів.

Ключові слова: навчання з підкріпленням, актор-критик алгоритм, паралельне навчання, асинхронний градієнтний спуск.

Рис.: 9. Бібл.: 6.

The method of deep reinforcement learning is considered in the work, which implements an asynchronous approach with the use of gradient descent. An actor-critic algorithm based on the proposed method is constructed. Parallel asynchronous algorithm is characterized by a greater stabilizing effect during learning process compared with existing parallel methods. In addition, the proposed approach allows parallelizing the learning process by applying the multi-core properties of modern computers.

Key words: reinforcement learning, actor-critic algorithm, parallel learning, asynchronous gradient descent.

Fig .: 9. Bibl.: 6.

Introduction. In connection with the growing popularity of algorithms concerning the field of artificial intelligence, the number of new approaches to the development of reinforcement learning algorithms has significantly increased.

At the same time, deep neural networks are a promising means of implementing artificial intelligence algorithms. However, application of reinforcement learning algorithms to the mentioned networks is problematic, which is caused by fundamentally unstable learning processes. In order to overcome this problem considerable efforts have been made in recent years [1,2]. The main result of the study

was that the main direction of increasing the sustainability of the learning process is to use the previous experience of the agent to correct the following actions.

However, this approach also has its own negative sides, because it requires additional memory and additional calculations per agent activity. Therefore, the main direction of further development of these algorithms lies in the field of parallelization. It is known that the fundamental difference between parallel algorithms from sequential is the presence of a phase of interaction in which parallel fragments must perform information exchange for the further successful advancement of local sub-processes that form a common parallel process. From the point of view of the total time of the algorithm, the interaction phase has a negative effect and therefore it is natural to combine this phase with the phase of computation, which leads to asynchronous parallel algorithms.

One of the well-known approaches that uses a parallel asynchronous algorithm for generalized reinforcing learning is Gorila (General Reinforcement Learning Architecture) [3]. In the Gorila algorithm, each agent is represented as a local sub-process that operates in a separate copy of the medium with its own memory for the accumulation of experience, and a learner who selects data from memory and calculates loss gradients. The gradients are asynchronously transmitted to the central parameter server, which updates the main copy of the model. Updated policy settings are sent to agent-learner at certain intervals. Such an approach makes sense for distributed systems because it is characterized by coarse-grained parallelism.

The algorithm "Sarsa" was proposed in [4]. This algorithm, as in the previous case, uses several parallel sub-processes such as "agent-learner" to accelerate the overall learning process. Each local sub-process shows a separate operation and periodically exchanges information about the learning outcomes using direct communications with other sub-processes. This approach allows for flexible computational organization, but leads to a significant increase in the time of interaction when there is an increase in the number of local sub-processes.

On the basis of the generalization of the above-mentioned approaches, a framework of asynchronous algorithms was proposed in [5] to parallelize the reinforcement learning process. In this framework, the parallel development algorithm "Sarsa" was further developed, algorithms of the actor-critic method and the n-stepping methods were considered.

The comparative characteristics of learning outcomes based on a wide range of research in various fields of practical application have shown that actor-critic methods have the best prospects. In this paper, the algorithm of parallel asynchronous learning based on the actor-critic method is considered. This algorithm allows for scaling and can be implemented without significant changes, so in multicore processors as in cloud environments.

Basics of the reinforcement learning theory. The reinforcements learning is used in algorithms to achieve a certain complex goal by maximizing the target

parameter. As a rule, the goal is achieved by performing a significant number of steps in a given direction. The choice of the right direction of movement is achieved by choosing the appropriate rewards and punishments.

The actor-critic structure consists of two main parts, as shown in fig.

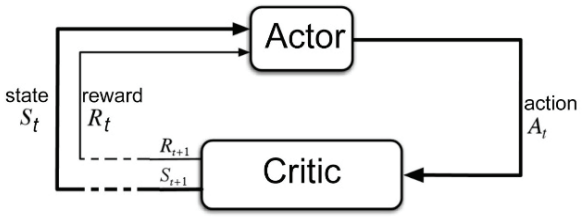

Fig. 1. The actor-critic structure

The terminology for such algorithms includes such concepts as actors, critics, states, rewards, etc. The actor performs certain activities that form a set of activities A. As a result of these actions, the actor changes his current state to a state that is an element of the set of possible states S. If these activities move the learning process into the right direction, then the actor receives an reward from the set of rewards R. The actor chooses the next activity based on an analysis of the current state by applying a policy π. The essence of π policy is to provide the actor with the greatest rewards from the next activity. In addition to the short-term benefits of current activity, the actor is also guided by the long-term profit that he can get in the long run after a certain amount of activity. Therefore, we will determine $V^{\pi}\left(s\right)$ as expected long-term profits from the current state s because of π policy implementation. There is one more long-term profit Q, which differs from V that depends not only on the current state, but also on the current activity $a \in A$. So $Q^{\pi}\left(s,a\right)$ means the long-term profits from the current state, if the activity is carried out as a result of policy π. Thus at any moment t, the policy π maps the state s_t to the activity a_t. After performing the activity a_t, the actor goes into the next state s_{t+1} and receives a reward, which is represented by a scalar value r_t. After k steps, starting with the step t, the actor can accumulate total profits $R_t = \sum_k \gamma^k r_{t+k}$, where $\gamma \in \left[0,1\right]$ is the discount factor. The main objective of the actor is to maximize profits from each state s_t. Consequently, long-term profit from the implementation of activity a_t in a state s_t is determined from the expression $Q\left(s_t,a_t\right) = \max_{\pi} Q^{\pi}\left(s_t,a_t\right)$ by choosing the maximum long-term profit achieved by any policy. With the use of artificial neural networks, as a universal approximator, we obtain the optimal function of long-term profit $Q\left(s,a\right) \approx Q\left(s,a;\theta\right)$. For this case, the parameter θ is optimized by the

iterative process of minimizing loss functions, where i -th loss function can be given by an expression

$$L_i\left(\theta_i\right)=\mathrm{E}\left(r+\gamma\max_{a'}Q\left(s_{t+1},a_{t+1};\theta_{i-1}\right)-Q\left(s_t,a_t;\theta_i\right)\right)^2,$$

where s_{t+1} is the state that arises after the state s_t and a_{t+1} is the activity that goes after the activity a_t.

This is one-step learning method, since the update of the value of the profit takes place after every step in value $r+\gamma\max_{a_{t+1}}Q\left(s_{t+1},a_{t+1};\theta\right)$.

This is the simplest approach since the reword r depends directly on the previous state and activity. Its main disadvantage is that the history of previous states s and activities a can only be taken into account indirectly through updates $Q\left(s,a\right)$. This leads to a slower learning due to reducing the sustainability of the learning process.

The natural way to increase the stability of the learning process is to apply multi-step methods. In the case of using an artificial neural network to approximate the function of long-term profit $Q\left(s,a\right)\approx Q\left(s,a;\theta\right)$, we receive an update after n steps that corresponds to the value

$$r_t+\gamma r_{t+1}+\ldots+\gamma^{n-1}r_{t+n-1}+\max_a\gamma^n Q\left(s_{t+n},a\right).$$

Therefore, the parameter θ can be optimized by minimizing loss functions $L_i\left(\theta_i\right)$ only after n steps, where the loss function can be given by expression

$$L_i\left(\theta_i\right)=\mathrm{E}\left(r_t+\sum_{k=1}^{k=n-1}\gamma^k r_{t+k}+\max_a\gamma^n Q\left(s_{t+n},a,\theta_{i-1}\right)-Q\left(s_t,a_t;\theta_i\right)\right)^2.$$

A significant disadvantage of this approach is the low degree of system response to changing environmental conditions. Because the result is formed on the basis of previous steps. Thus, the multi-step method can increase the stability of the learning method by increasing the number of calculations and increasing the response time to a situation that arises due to changes in the parameters of the environment.

Asynchronous learning. In this paper, it is proposed an approach to reinforcement training that combines the benefits of one-step and multi-step methods for the actor-critic structures. The main direction of further development reinforcement learning algorithms for deep neural networks lies in the field of parallelization. Therefore, in this approach, it is suggested to use parallel computing with shared RAM. This allows you to effectively use the multi-core architecture of modern

processors. At the same time, it is also important to have an effective computation organization in such a parallel structure. It is known that the fundamental difference between parallel algorithms from sequential is the presence of an interaction phase. The organization of parallel computing with synchronous and asynchronous interaction of processes is shown in fig. 2.

We can see that in the case of asynchronous computing we have the advantage of combining the computation and interaction phases.

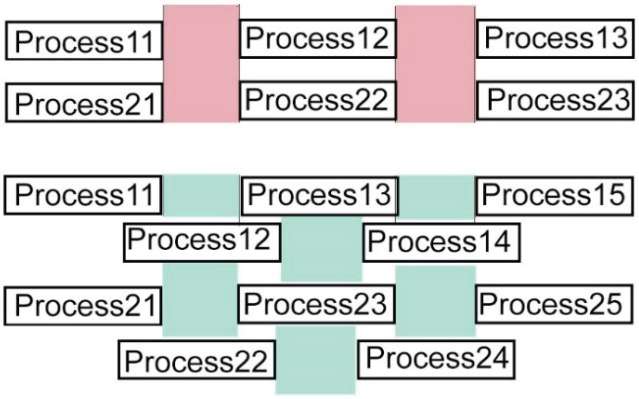

Fig. 2. Synchronous and asynchronous calculations

Our parallel asynchronous actor-critic of the structure is shown in fig. 3.

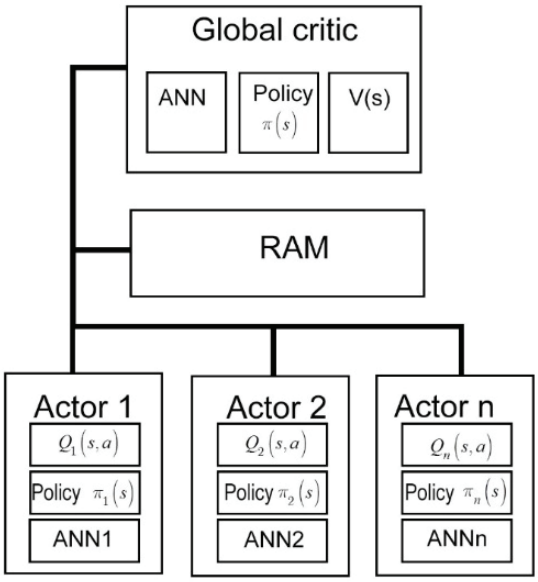

Fig.3. Parallel asynchronous actor-critic structure

The proposed parallel structure contains n actors that can simultaneously accumulate experience of the environment, guided by local politics $\pi_i(s_t)$. As shown in [6], such an approach can be considered as an actor-critic architecture for which a separate policy π is typical for actors and the critic is determined by some value

function $V\left(s_t\right)$. Then the value $R_t - V\left(s_t\right)$ can be considered as an estimate of some advantage for activity a_t in the state s_t.

If we denote by $A\left(s_t, a_t, \theta\right)$ the function of the advantage generated by artificial neural networks with parameters θ, then we obtain the general expression for parallel asynchronous method

$$A\left(s_t, a_t; \theta\right) = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^k V\left(s_{t+n}; \theta\right) - V\left(s_t; \theta\right).$$

The parallel asynchronous algorithm is proposed for reinforcement learning which is based on the described approach. The pseudocode of this algorithm is shown in fig. 7.

```
Set shared and local parameters θ_t and θ_{t+1}
Set the shared an local counter T = 0 and t = 1
While T < T_max:
    dθ = 0; θ_{t+1} = θ_t; t_start = t
    Get(s_t)
    While s_t != s_final or t - t_start < t_max:
      Calculate a_t according to π(a_t | s_t; θ_{t+1})
      T+=1; t+=1
    If s_t == s_final: R=0 else R = V(s_t, θ_{t+1})
    For i in {t-1,…,t_start}:
        R = γr_i + R; Calculate θ_{t+1} = dθ + ∇_{θ_t} A(s, a; θ)
    Asynchronous update of the θ parameters.
```

Fig. 7. Pseudo-code for parallel asynchronous algorithm

Practical implementation with the cart-pole problem. The cart-pole also known as an inverted pendulum with a center of gravity above its pivot point. It is unstable and falls over but can be controlled by moving the cart.

The goal of the problem is to keep the pole balanced by moving the cart left or right by applying appropriate forces to the pivot point. Fig. 8 demonstrates the simplest implementation of the cart-pole task.

This implementation made on the base of gym library. It is a collection of test problems. We can use them to work out our reinforcement learning algorithms. These environments have a shared interface that allows writing the general algorithms.

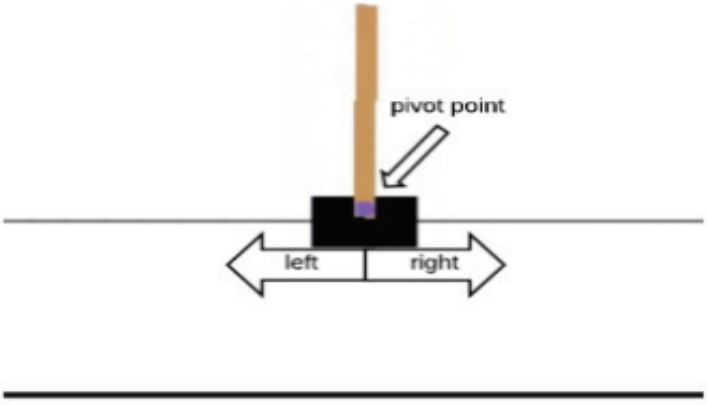

Fig. 8. A typical graphical representation of the cart-pole task

Investigation of the received algorithm showed its high stability while maintaining the high speed of learning. In fig. 9 we can show an example of comparison of learning speed using asynchronous parallel method and parallel one-step Q - method. The x-axis shows training time in hours and the y-axis shows the average score.

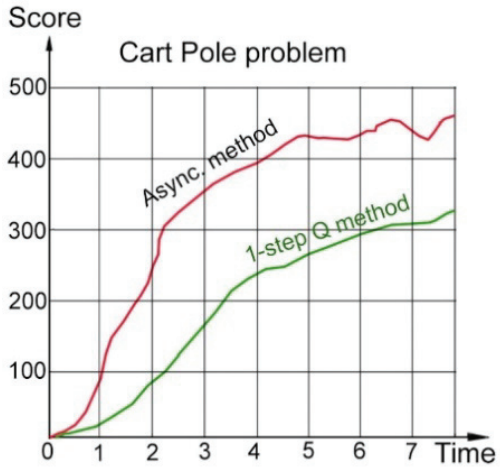

Fig. 9. Comparison of asynchronous and single-step parallel methods

Our asynchronous method shows significant speedups from using greater numbers of parallel actors.

Conclusions. In connection with the development of robotics, reinforcement learning is highly relevant as it is an important part of artificial intelligence technology. The main problem for reinforcement learning of convergent multilayer networks remains the stability of the method, which significantly influences the learning speed. The paper proposes an asynchronous parallel method for reinforcement learning, which allows increasing the speed of learning in comparison with one-step parallel methods.

References

1. Mnih V., Kavukcuoglu K., Silver D., et al.–Nature, 2015.–vol.518, №7540.– P.529-533

2. Van Hasselt H., Guez A., Silver D. Deep reinforcement learning with double q-learning, 2015.– *preprint arXiv:1509.06461*.

3. Nair A., Srinivasan P., Blackwell S., et al. – Massively Parallel Methods for Deep Reinforcement Learning, 2015. – *arXiv:1507.04296v2.*

4. Grounds M., Kudenko D. Parallel reinforcement learning with linear function approximation // *Proceedings of the 5th, 6th and 7th European Conference on Adaptive and Learning Agents and Multi-agent Systems: Adaptation and Multi-agent Learning*. – Springer-Verlag: 2008, P. 60–74.

5. Mnih V., Badia A., Mirza M., et, al. Asynchronous Methods for Deep Reinforcement Learning, 2016. – arXiv: 1602.01783 [cs.LG]

6. Sutton, R. and Barto, A. *Reinforcement Learning: an Introduction*. MIT Press, 1998. – 548 p.

Довідка про автора

Новотарський Михайло Анатолійович – професор, кафедра обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

Novotarskyi Mykhailo – professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”. E-mail: novot@ukr.net

Розширена анотація

Сучасні підходи до створення алгоритмів у сфері штучного інтелекту включають як створення нових, так і модернізацію відомих методів навчання без учителя, зокрема, навчання з підкріпленням. Водночас перспективним засобом реалізації штучного інтелекту є глибокі нейронні мережі. Проте застосування алгоритмів навчання з підкріпленням до згаданих мереж викликає певні проблеми, обумовлені принципово нестійкими процесами навчання. Основним напрямком підвищення стійкості процесу навчання є використання агентом попередньо набутого досвіду з метою коригування наступних дій. Слід відмітити, що такий підхід має свої негативні сторони, оскільки вимагає додаткової пам’яті та додаткових обчислень з розрахунку на одну активність агента. Тому основний напрямок подальшого розвитку даних алгоритмів лежить у сфері розпаралелювання.

У даній роботі розглянуто метод глибокого навчання з підкріпленням, який ґрунтується на застосуванні асинхронних паралельних алгоритмів при реалізації градієнтного спуску. На основі запропонованого методу побудовано актор-критик алгоритм, який характеризується більшим стабілізуючим ефектом при навчанні у порівнянні з існуючими паралельними методами. Крім того, запропонований підхід дозволяє розпаралелювати процес навчання шляхом застосування властивості багатоядерності сучасних комп’ютерів.