DOI: 10.18721/JCSTCS.14106
УДК 004

REINFORCEMENT LEARNING FOR INDUSTRIAL MANUFACTURING CONTROL SYSTEM

M.Ya. Hanafi, V.P. Shkodyrev

Peter the Great St. Petersburg Polytechnic University,
St. Petersburg, Russian Federation

The problem posed is a very general case of optimal control of a dynamic, potentially stochastic, and partially observable system for which a model is not necessarily available. We analyze the disadvantages of classical approaches of the control theory and present a new modified numerical reinforcement learning rule of machine learning algorithm. Control theory is a field that has been studied for a very long time and which deals with the behavior of dynamic systems and how to influence it. Among the best-known examples are LQG (Linear Quadratic Gaussian) or PID (Proportional Integral Derivative) controllers. Most of the existing approaches presuppose (analytical) knowledge of the dynamic system, and one of the constraints is the need to be able to free oneself from a priori models. We focus on modified reinforcement learning approach to adaptive control policy as perspective area of control of complex dynamical system under uncertainty.

Keywords: reinforcement learning, multi agent system, oil manufacturing, Bellman equations, dynamic programming.

Citation: Hanafi M.Ya., Shkodyrev V.P. Reinforcement learning for industrial manufacturing control system. Computing, Telecommunications and Control, 2021, Vol. 14, No. 1, Pp. 60–69. DOI: 10.18721/JCSTCS.14106

УСИЛЕНИЕ ОБУЧЕНИЯ ДЛЯ СИСТЕМЫ УПРАВЛЕНИЯ ПРОМЫШЛЕННЫМ ПРОИЗВОДСТВОМ

Я.М. Ханафи, В.П. Шкодырев

Санкт-Петербургский политехнический университет Петра Великого,
Санкт-Петербург, Российская Федерация

Поставленная задача представляет общий случай оптимального управления динамической, потенциально стохастической и частично наблюдаемой системой, для которой модель не обязательно доступна. В статье представлен анализ недостатков классических подходов теории управления и предлагается новый модифицированный алгоритм машинного обучения с подкреплением. Теория управления – область, которая изучалась очень долгое время и которая касается поведения динамических систем и того, как на нее влиять. Среди наиболее известных примеров – LQG (Линейно-квадратичное гауссовское управление) или ПИД-контроллеры. Большинство существующих подходов предполагают (аналитическое) знание динамической системы, и одним из ограничений является необходимость иметь возможность освободиться от априорных моделей. Мы концентрируем внимание на преимуществах использования моделей машинного обучения с подкреплением как перспективной стратегии управления сложными динамическими системами в условиях неопределенности.

Ключевые слова: усиление обучения, мультиагентная система, нефтепереработка, уравнения Беллмана, динамическое программирование.

Ссылка при цитировании: Hanafi M.Ya., Shkodyrev V.P. Reinforcement learning for industrial manufacturing control system // Computing, Telecommunications and Control. 2021. Vol. 14. No. 1. Pp. 60–69. DOI: 10.18721/JCSTCS.14106

Introduction

Reinforcement learning is the digital learning environment's solution to the problem of optimal control. In this paradigm, IT agents learn to control an environment by interacting with it [1, 2]. They regularly receive local information about the quality of the control carried out in the form of a digital reward (or reinforcement signal), and their objective is to maximize a cumulative function of these rewards over the long term, generally modelled by a so-called value "reward function". The choice of actions applied to the environment according to its configuration is called a policy, and the value function therefore quantifies the quality of that policy [3]. Generally speaking, the agent does not have a model (neither physical nor statistical for example) of its environment, nor of the reward function that defines the optimality of the control [4, 5]. However, a common assumption we make is that the environment is Markovian, i.e., the effect of the application of an action depends only on the current configuration of the environment, not on the path taken to reach it [6]. This standard is very general and makes it possible to focus on a large number of applications. However, its practical application can be difficult. First of all, when the description of the environment to be controlled is too large, an accurate representation of the value (or policy) function is not possible [7]. In this case, the problem of generalization arises (or function approximation): on the one hand, it is necessary to design algorithms whose algorithmic complexity is not too great, and on the other hand they should be capable of inferring the behavior to be followed for an unknown environment configuration when similar situations have already been experienced. Another problem lies in the fact that, in the most general case, the agent learns to control the environment while at the same time controlling it [8]. This often results in successive phases of quality assessment of a policy and its improvement. From a learning perspective, this induces non-stationarity (we evaluate the quality of a policy that is constantly modified), a problem rarely addressed in the literature as such. This interweaving of learning and control also causes a problem known as the dilemma between exploration and exploitation. For each action it chooses, the agent must decide between an action it considers optimal in relation to his imperfect knowledge of the world and another action, considered sub-optimal, aimed at improving this knowledge. To deal with this problem effectively, it should be possible to estimate confidence that the agents have in their estimates. If these different difficulties are known, the methods of the literature generally treat them separately. Thus, a thought-out method for dealing with the dilemma between exploration and exploitation will not necessarily adapt to the problem of generalization, and vice versa.

Background

There are several approaches to deal with the reinforcement learning paradigm. However, an important part of the literature is based on dynamic programming and it is with this view that we approach it.

Dynamic programming. Dynamic programming can be defined in a very general way as a set of algorithmic techniques whose principle is to determine the optimal solution of a problem from an optimal solution of a sub-problem. In our context, these are all the methods that allow the exact (or approximate) solution of the Bellman equation, without any learning component [9].

Bellman equations. The objective of dynamic programming is to discover one of the policies whose value function is maximal for the set of states [10]. Noting $\left(S \to \mathbb{R}\right) \subset \mathbb{R}^{|S|}$ where $\left|S\right|$ is the cardinal of the state space, or in other words that the value function can be seen as a vector with as many components as there are states, it is possible to equip the value functions with a partial order relation [11]:

$$V_1 \le V_2 \Leftrightarrow \forall s \in S, V(s) \le V_2(s). \quad (1)$$

A partial order can be defined based on the policies, via the associated value functions:

$$\pi_1 \le \pi_2 \Leftrightarrow V^{\pi_1} \le V^{\pi_2}. \quad (2)$$

The objective is therefore to determine the optimal policy π^* defined by:

$$\pi^* = \arg\max_{\pi} V^{\pi}. \quad (3)$$

However, it is possible to define the value function recursively:

$$\begin{aligned} V^{\pi}(s) &= E\left[\sum_{i=0}^{\infty} \gamma^i R(S_i, \pi(S_i), S_{i+1}) \middle| S_0 = s, \pi\right]; \\ &= E_{s'|s,\pi(s)}\left[R(s, \pi(s), s') + \gamma V^{\pi}(s')\right]; \\ &= \sum_{s' \in S} p(s'|s, \pi(s))\left(R(s, \pi(s), s') + \gamma V^{\pi}(s')\right). \end{aligned} \quad (4)$$

The value of a state is therefore the average (according to the transition probabilities) of the sum of the reward obtained following the application of the action specified by the policy we wish to evaluate and the value of the state towards which the system transits, weighted by the discount factor [12]. This equation, called the Bellman evaluation equation, defines a linear system of $|S|$ equations with $|S|$ unknowns to determine the value function of a given policy and thus quantify its quality [13].

Another equation, this one being non-linear, allows us to directly determine the function of optimal value $V^* = V^{\pi^*}$. Given the state s, assume the optimal value function known in the states s' to which the system can transit. The optimal value function in state s' maximizes (weighted by the transition probabilities) the immediate reward plus the optimal value in state s' to which the system transits, weighted by the discount factor. This is the Bellman optimality equation [11, 13]:

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} p(s'|s, a)\left(R(s, a, s') + \gamma V^*(s')\right). \quad (5)$$

This defines a non-linear system with $|S|$ equations and $|S|$ unknowns. If the optimal value function is known, it is easy to deduce the optimal policy, which is greedy with respect to the latter, i.e., it verifies:

$$\pi^*(s) = \arg\max_{a \in A} \sum_{s' \in S} p(s'|s, a)\left(R(s, a, s') + \gamma V^*(s')\right). \quad (6)$$

We summarize these important equations for the following, defining in passing the corresponding operators T^{π} and T^*.

Bellman's evaluation equation is used to determine the value function of a given policy π:

$$\forall s \in S, V^{\pi}(s) = \sum_{s' \in S} p(s'|s, a)\left(R(s, a, s') + \gamma V^*(s')\right) \Leftrightarrow V^{\pi} = T^{\pi} V^{\pi}. \quad (7)$$

The Bellman optimality equation is used to determine the optimal value function V^*:

$$\forall s \in S, V^*(s) = \max_{a \in A} \sum_{s' \in S} p(s'|s,a)\left(R(s,a,s') + \gamma V^*(s')\right) \Leftrightarrow V^* = T^* V^*. \quad (8)$$

Moreover, the Bellman operators T^π and T^* are infinite norm contractions.

Method Notation

We focus on two methods whose ideas are used extensively in reinforcement learning, namely the policy iteration and value iteration algorithms.

Policy Iteration. The first approach we present is policy iteration. The algorithm (see Listing 1) is initialized with any policy. Its principle is to evaluate the value function of the current policy (which we call policy evaluation), and then improve this policy by considering the greedy policy with respect to the previously calculated value function. In other words, in a given state, the action chosen is the one that leads (on average) to the greatest accumulation of rewards. It is important to note that this is not necessarily the action chosen by the policy, unless the policy is optimal [10–12, 14]. More formally, if at iteration i the policy π_i is evaluated, the improved policy π_{i+1} is defined by:

$$\forall s \in S, \pi_{i+1}(s) = \arg\max_{a \in A} \sum_{s' \in S} p(s'|s,a)\left(R(s,a,s') + \gamma V^{\pi_i}(s')\right). \quad (9)$$

With this improvement scheme, it can be shown that there is policy improvement, i.e. $\pi_{i+1} \geq \pi_i$. If one has equality, then Bellman's optimality equation is verified, and the algorithm has converged. Moreover, since the number of policies is finite, this algorithm converges in a finite number of iterations (this number of iterations being empirically much smaller than the cardinal of the policy space). It should be noted that the evaluation of the policy is equivalent to solving a linear system, the complexity of this evaluation is therefore in $O\left(|S|^3\right)$. The existence of a solution to this system is guaranteed by the Banach fixed point theorem (V_i is the fixed point of the contraction T^{π_i}). The computation of the associated greedy policy is in $O\left(|S|^2 |A|\right)$. The complexity of this algorithm is therefore in $O\left(|S|^2 |A| + |S|^3\right)$ per iteration. The ideas of this algorithm, i.e., evaluation of a policy followed by improvement, are widely used in the ideas of this algorithm, i.e., policy evaluation followed by improvement, are widely used in reinforcement learning algorithms [14, 15].

Value Iteration. The second approach we present, based on Bellman's optimality equation, aims at directly determining the optimal value function V^*. Since the Bellman operator T^* is a contraction and V^* is its unique fixed point, the *Algorithm 02* has a finite number of iterations, which is why a stopping criterion

Listing 1. Algorithm 1: Policy Iteration

Algorithm 1: **Policy Iteration**

Initialization
Policy π_i
i=0
While π_{i+1} != π_i do
    Evaluation of the policy;
    Solve $V_i = T^{\pi i} V_i$
    Improvement of the policy
    For ALL $s \in S$
        $\pi_{i+1}(s) = \arg\max_{a \in A} \sum_{s' \in S} p(s'|s,a)\left(R(s,a,s') + \gamma V^{\pi_i}(s')\right)$
    i++

Listing 2. Algorithm 2: Value Iteration

Algorithm 2: **Value Iteration**

Initialization
Value function V_0
i=0
While $\|V_{i+1} - V_i\|_\infty \geq \varepsilon$ **do**
    Iterating the value;
    $V_{i+1} = T^* V_i$
    i++

is introduced. This allows the error to be limited. Indeed, it can be shown that if the stopping criterion is verified at iteration i, then $\|V_i - V^*\| \leq \frac{2\gamma}{1-\gamma}\varepsilon$. However, this bound does not guarantee the quality of the associated greedy policy (see Listing 2).

Case studies and experimental results

Our case studies are based on real oil manufacturing production data. Oil manufacturing production is a complex process that has a hierarchy structure and many complex sub-systems.

Every manufacturing is based on a sequence of processes, each of those processes has its inputs and outputs. Depending on its structure, each process can have one or more outputs, which in turn become the inputs of the next process or the final products. Each process can be controlled by a set of factors, that can interfere with the outputs. Each of those processes can be composed by a sub-process so at the end of the structure of the factories we will have a hierarchy structure the base of this structure is the key to optimal control of the manufacturing system. We describe objective control of a complex technical system as a network of interest in a manufacturing subsystem. Every manufacture aims for high profit from its prod-

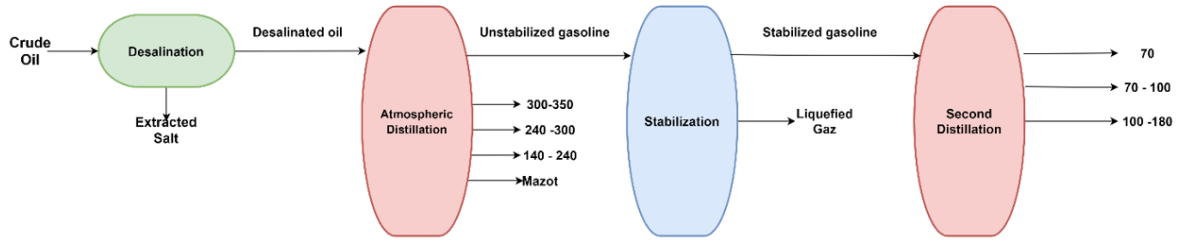

Fig. 1. Technological process of oil manufacturing

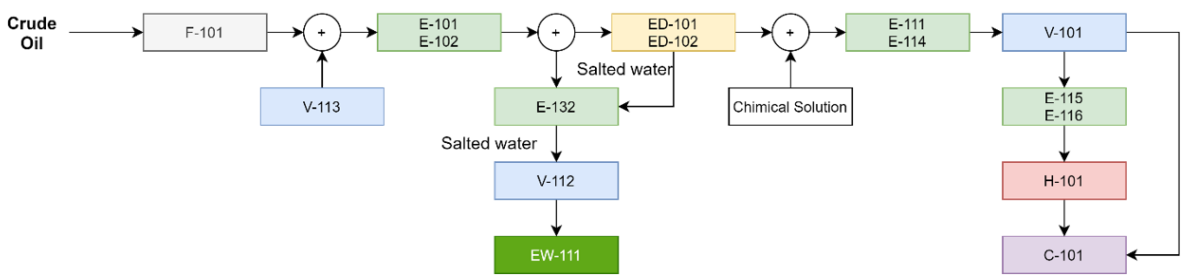

Fig. 2. Technological process of a desalination unit in oil manufacturing

ucts, and the oil manufacturing is not different. It is critical for oil manufacturing to have good quality of their production and that will be the top objective of the manufacturing alongside with the quantity of the production which is the second objective of the manufacturing to highly benefits from the production. The quality and the productivity are mostly the top objective on every manufacturing alongside other objective depends on the type of the manufacturing. So generally, there are always two or more objectives which makes us categorize this problem as a multi objective problem [16, 17].

Multi Objective Optimization: is an area of multiple criteria decisions making, that is concerned with mathematical optimization problems involving more than one objective function to be optimized simultaneously.

$$\begin{cases} \dfrac{\max}{\min} f_m(x) \quad m = 1, 2, .., M; \\ g_i(x) \geq 0 \quad j = 1, 2, .., J; \\ h_k(x) \geq 0 \quad k = 1, 2, ..., K; \\ x_i^{(L)} \leq x_i \leq x_i^{(U)} \quad i = 1, 2, ..., n. \end{cases} \tag{10}$$

The vector x is a vector of n decision variables $x = (x_1, x_2, ..., x_n)^T$.$x_i^{(L)}$ and $x_i^{(U)}$ is the lower and upper bounds of variable x_i, respectively. These variables define decision space or research space D. Generally, an element of the research space is called a possible or potential solution. The terms $g_j(x)$ and $h_k(x)$ are the constrained functions. Inequality constraints are treated as "superior or equal" type constraints since "inferior or equal" type constraints can be treated as duality. A solution x that does not satisfy all $(J + K)$ constraints is said to be an unfeasible solution. The set of feasible solutions constitutes a feasible region. The vector $f(x) = (f_1(x), f_2(x), ..., f_M(x))^T$ is the objective vector. Each of the M objective functions is either maximizing or minimizing which depends on the problem addressed. Using the principle of duality, a problem of maximization can be reduced to a problem of minimization by multiplying the objective function by -1 [18, 19].

Multi agent parallelization system. The oil manufacturing environment has a distributed and complex hierarchy. For that, we need to use a multi agent parallelization system as it deals with a large control process that controls multiple components.

As the Approximate Policy iterates with Value and Policy Networks, the standards scheme for approximation space value E involves using a cost function approximation $\tilde{J}$ of J (the optimal cost function). At a given state x that will minimize (or maximize) the approximation E, which forms as an expected value involved in the cost of the first stage $g(x, u, w)$ and the future costs which sufficiently reduced, in that we can note this as the approximate $\tilde{Q}$ factor corresponding to a pair (x, u), that minimizing the Q factors overall and that gives you a control that it used in state X [20].

$$\text{At } x: \min_{u \in U(x)} E\left\{ g(x,u,w) + \alpha \tilde{J}\left(f(x,u,w) \right) \right\}, \tag{11}$$

where E is the approximation, $g(x, u, w)$ is the first stage, $\alpha \tilde{J}\left(f(x,u,w) \right)$ is the optimal cost approximation (future stage).

One of the issues emerging is how to approximate the policies. The solution of that is to introduce a family of policies $\mu(x, r)$, a parametric family that depends on a parameter r [21].

Figures 3 and 4 show that each sub-system is a system environment having its own agent that controls it depending on the state of the environment itself. By applying the reinforcement learning to our system, we can describe its characteristics as:

• *Environment*: the process itself is the environment of the agent, which has every value that the agents need to take actions.

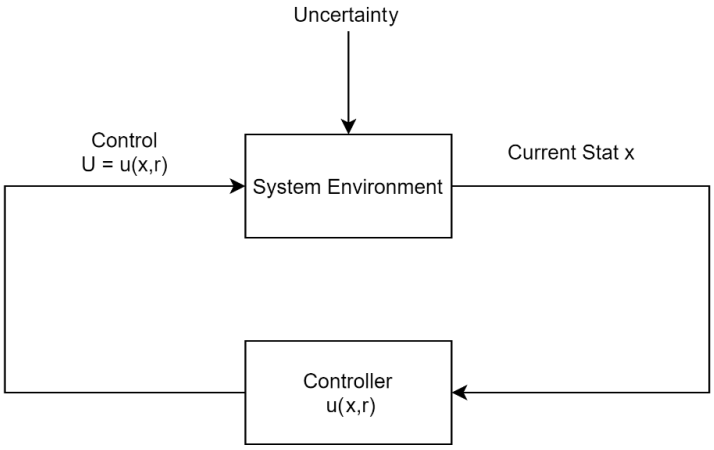

Fig. 3 Optimization and training over parametric family of policies

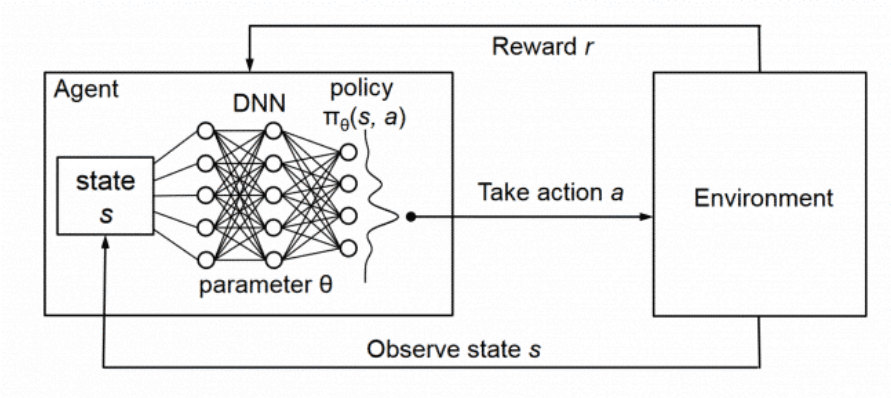

Fig. 4. Reinforcement Learning Loop

• *State*: is it the actual situation for the agent, which it retains if the result is satisfactory and there are no problems, or there are no changes in the environment or task from other agents.

• *Reward*: the agent's reward depends on the output goals, in the start of the learning the reward can be a failure, but over time, it learns the best configuration that can be used.

• *Discount Factor*: as the process advances, the agents put new goals to follow (increasing the productivity and the quality).

• *Policy*: our agents use dynamic policy to achieve the optimal goals, after a certain time of learning the agents stick to a certain policy that gives the best results.

• *Value*: as those processes have limitation, an agent calculates a future value (best value to obtain) and tries to achieve it; if it achieves it over time, the agent determines a new value for future achievement.

• *Q-Value*: the agents take extra action in case of unknown configuration, in case a configuration exceeds a limit, or a hazard is detected.

• *Action*: depending on all the above parameters, the agents act by changing the control values of the process.

Figure 5 presents the sequence of an agent process control by updating its own value and policy leading to optimal control.

As the agent has an objective to maximize quality and productivity, Fig. 5 (1–6) shows that each time it developed a new policy and determined a new value to reach the objective depending on the environment.

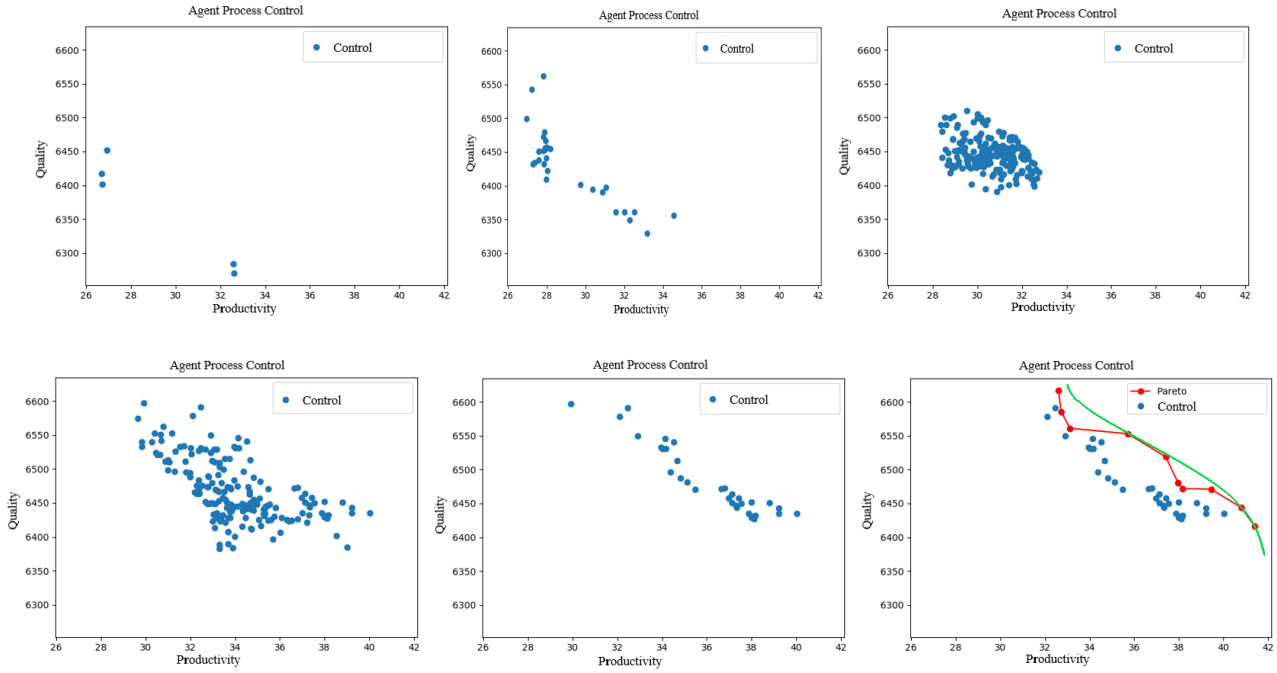

Fig. 5. Agent's policy and value updating results

The red line represents the agent's optimal value that it reaches (Pareto values), and the green line is the new value that the agent is trying to reach (The Pareto Front).

Conclusion

The field of reinforcement learning has exploded in recent years. Ever since the impressive breakthrough on the ImageNet classification challenge in 2012, the successes of supervised deep learning have continued to pile up and people from many different backgrounds have started using deep neural networks to solve a wide range of new tasks including the ways of learning intelligent behavior in complex dynamic environments. In this article, we took the advantage of the reinforcement learning, which consists in the fact that the agent can adapt to its environment by updating the policy it is using and the value it determines to reach an optimal control. The results obtained in this article not only show an optimal configuration, but can also prevent an error leading to enormous risks and losses as the agent can understand the limitation of the environment, take an extra action depending on an unknown configuration, and prevent the human errors in the manufacturing.

REFERENCES

1. **Singh S., Lewis R., Barto A., Sorg J.** Intrinsically motivated reinforcement learning: An evolutionary perspective. *Autonomous Mental Development, IEEE Transactions on*, 2010, Vol. 2, Pp. 70–82. DOI: 10.1109/TAMD.2010.2051031

2. **Sundas A., Bhatia A., Saggi M., Ashta J.** *Reinforcement Learning*, 2020, P. 281.

3. **Matignon L., Laurent G.J., Le Fort-Piat N.** Reward function and initial values: Better choices for accelerated goal-directed reinforcement learning. *Artificial Neural Networks – ICANN 2006*, Berlin, Heidelberg, 2006, Pp. 840–849. DOI: 10.1007/11840817_87

4. **Macal C.M., North M.J.** Tutorial on agent-based modelling and simulation. *Journal Simulation*, 2010, Vol. 4, No. 3, Pp. 151–162. DOI: 10.1057/jos.2010.3

5. **van Gog T., Rummel N.** Example-based learning: Integrating cognitive and social-cognitive research perspectives. *Educ Psychol Rev.*, 2010, Vol. 22, No. 2, Pp. 155–174. DOI: 10.1007/s10648-010-9134-7

6. **Hadoux E.** *Markovian sequential decision-making in non-stationary environments: application to argumentative debates*, Nov. 2015, P. 117.

7. **Arulkumaran K., Deisenroth M.P., BrundageM., Bharath A.A.** A brief survey of deep reinforcement learning. *IEEE Signal Process. Mag.*, 2017, Vol. 34, No. 6, Pp. 26–38. DOI: 10.1109/MSP.2017.2743240

8. **Lillicrap T.P., et al.** Continuous control with deep reinforcement learning. arXiv:1509.02971 [cs. LG], Sept. 2015. Available: *http://arxiv.org/abs/1509.02971* (Accessed: 27.03.2021).

9. **Helman P.** The principle of optimality in the design of efficient algorithms. *Journal of Mathematical Analysis and Applications*, 1986, Vol. 119, No. 1–2, Pp. 97–127. DOI: 10.1016/0022-247X(86)90147-2

10. **Puterman M.L., Patrick J.** Dynamic programming. *Encyclopedia of Machine Learning*. Boston, MA: Springer US, 2010, Pp. 298–308.

11. **Feng Y., Li L., Liu Q.** A Kernel Loss for Solving the Bellman Equation. arXiv:1905.10506v3 [cs. LG], 8 Jan. 2020. Available: *http://arxiv.org/abs/1905.10506* (Accessed: 27.03.2021).

12. **Geist M., Pietquin O.** Kalman temporal differences. *Journal of Artificial Intelligence Research (JAIR)*, 2010, Vol. 39, Pp. 483–532. DOI: 10.1613/jair.3077

13. **Aguilar C.O., Krener A.J.** Numerical solutions to the Bellman equation of optimal control. *J. Optim Theory Appl*, 2014, Vol. 160, No. 2, Pp. 527–552. DOI: 10.1007/s10957-013-0403-8

14. **Otterlo M., Wiering M.** Reinforcement learning and Markov decision processes. *Reinforcement Learning: State of the Art*, 2012, Pp. 3–42. DOI: 10.1007/978-3-642-27645-3_1

15. **Beitelspacher J., Fager J., Henriques G., Mcgovern A.** *Policy Gradient vs. Value Function Approximation: A Reinforcement Learning Shootout*, March 2006.

16. **Yassine H.M., Shkodyrev V.P.** Optimal production manufacturing based on intelligent control system. *Technological Transformation: A New Role for Human, Machines and Management*, Cham, 2021, Pp. 210–220. DOI: 10.1007/978-3-030-64430-7_18

17. **Yassine H.M., Shkodyrev V.P.** The intelligent control system of optimal oil manufacturing production. *The 3rd International Conference on Computational Intelligence and Intelligent Systems*, New York, NY, USA, Nov. 2020, Pp. 131–135. DOI: 10.1145/3440840.3440848

18. **Gunantara N.** A review of multi-objective optimization: Methods and its applications. *Cogent Engineering*, 2018, Vol. 5. DOI: 10.1080/23311916.2018.1502242

19. **Deb K.** Multi-objective optimization. *Search Methodologies*. Berlin: Springer, 2014, Pp. 403–449.

20. **Nissim R., Brafman R.** *Multi-agent A* for parallel and distributed systems*. 2012, P. 1266.

21. **Rousset A., Herrmann B., Lang C., Philippe L.** A survey on parallel and distributed multi-agent systems for high performance computing simulations. *Computer Science Review*, 2016, Vol. 22. DOI: 10.1016/j.cosrev.2016.08.001

Received 07.03.2021.

СПИСОК ЛИТЕРАТУРЫ

1. **Singh S., Lewis R., Barto A., Sorg J.** Intrinsically motivated reinforcement learning: An evolutionary perspective // Autonomous Mental Development, IEEE Transactions on. 2010. Vol. 2. Pp. 70–82. DOI: 10.1109/TAMD.2010.2051031

2. **Sundas A., Bhatia A., Saggi M., Ashta J.** Reinforcement Learning. 2020. P. 281.

3. **Matignon L., Laurent G.J., Le Fort-Piat N.** Reward function and initial values: Better choices for accelerated goal-directed reinforcement learning // Artificial Neural Networks – ICANN 2006. Berlin: Heidelberg, 2006. Pp. 840–849. DOI: 10.1007/11840817_87

4. **Macal C.M., North M.J.** Tutorial on agent-based modelling and simulation // J. Simulation. 2010. Vol. 4. No. 3. Pp. 151–162. DOI: 10.1057/jos.2010.3

5. **van Gog T., Rummel N.** Example-based learning: Integrating cognitive and social-cognitive research perspectives // Educ Psychol Rev. 2010. Vol. 22. No. 2. Pp. 155–174. DOI: 10.1007/s10648-010-9134-7

6. **Hadoux E.** Markovian sequential decision-making in non-stationary environments: Application to argumentative debates. Nov. 2015. P. 117.

7. **Arulkumaran K., Deisenroth M.P., BrundageM., Bharath A.A.** A brief survey of deep reinforcement learning // IEEE Signal Process. Mag. 2017. Vol. 34. No. 6. Pp. 26–38. DOI: 10.1109/MSP.2017.2743240

8. **Lillicrap T.P., et al.** Continuous control with deep reinforcement learning // arXiv:1509.02971 [cs. LG], Sept. 2015 // URL: http://arxiv.org/abs/1509.02971 (Дата обращения: 27.03.2021).

9. **Helman P.** The principle of optimality in the design of efficient algorithms // J. of Mathematical Analysis and Applications. 1986. Vol. 119. No. 1–2. Pp. 97–127. DOI: 10.1016/0022-247X(86)90147-2

10. **Puterman M.L., Patrick J.** Dynamic programming // Encyclopedia of Machine Learning. Boston, MA: Springer US, 2010, Pp. 298–308.

11. **Feng Y., Li L., Liu Q.** A Kernel Loss for solving the Bellman equation // arXiv:1905.10506v3 [cs. LG], 8 Jan. 2020 // URL: http://arxiv.org/abs/1905.10506 (Дата обращения: 27.03.2021).

12. **Geist M., Pietquin O.** Kalman temporal differences // J. of Artificial Intelligence Research (JAIR). 2010. Vol. 39. Pp. 483–532. DOI: 10.1613/jair.3077

13. **Aguilar C.O., Krener A.J.** Numerical solutions to the Bellman equation of optimal control // J. Optim Theory Appl. 2014. Vol. 160. No. 2. Pp. 527–552. DOI: 10.1007/s10957-013-0403-8

14. **Otterlo M., Wiering M.** Reinforcement learning and Markov decision processes // Reinforcement Learning: State of the Art. 2012. Pp. 3–42. DOI: 10.1007/978-3-642-27645-3_1

15. **Beitelspacher J., Fager J., Henriques G., Mcgovern A.** Policy Gradient vs. Value Function Approximation: A Reinforcement Learning Shootout. March 2006.

16. **Yassine H.M., Shkodyrev V.P.** Optimal production manufacturing based on intelligent control system // Technological Transformation: A New Role for Human, Machines and Management. Cham, 2021. Pp. 210–220. DOI: 10.1007/978-3-030-64430-7_18

17. **Yassine H.M., Shkodyrev V.P.** The intelligent control system of optimal oil manufacturing production // The 3rd Internat. Conf. on Computational Intelligence and Intelligent Systems. NY, USA, Nov. 2020. Pp. 131–135. DOI: 10.1145/3440840.3440848

18. **Gunantara N.** A review of multi-objective optimization: Methods and its applications // Cogent Engineering. 2018. Vol. 5. DOI: 10.1080/23311916.2018.1502242

19. **Deb K.** Multi-objective optimization // Search Methodologies. Berlin: Springer, 2014. Pp. 403–449.

20. **Nissim R., Brafman R.** Multi-agent A* for parallel and distributed systems. 2012. P. 1266.

21. **Rousset A., Herrmann B., Lang C., Philippe L.** A survey on parallel and distributed multi-agent systems for high performance computing simulations // Computer Science Review. 2016. Vol. 22. DOI: 10.1016/j.cosrev.2016.08.001

Статья поступила в редакцию 07.03.2021.

THE AUTHORS / СВЕДЕНИЯ ОБ АВТОРАХ

Ханафи Яссин Мохамед
Hanafi Mohamed Yassine
E-mail: hanafi.med.yassine@gmail.com

Шкодырев Вячеслав Петрович
Shkodyrev Viacheslav P.
E-mail: shkodyrev@spbstu.ru