ADAPTIVE CONTROL OF ROBOTIC PRODUCTION SYSTEMS

Viktor Smorodin
Department of Mathematical Problems
of Control and Informatics
Francisk Skorina Gomel State University
Gomel, Belarus
Email: smorodin@gsu.by

Vladislav Prokhorenko
Department of Mathematical Problems
of Control and Informatics
Francisk Skorina Gomel State University
Gomel, Belarus
Email: snysc@mail.ru

***Abstract*—The purpose of the work, that is presented in this paper, is to develop a method for adaptive control of a technological production cycle based on a software and hardware system that includes indicators of the hardware units states, parameters of the technological production cycle operation, simulation model of the probabilistic technological process and a built-in decision-making system. Operational interaction of the software and hardware system components and construction of the feedback control connections is implemented through the control parameters and variables of the simulation model based on the output of the neuroregulator model. To address the described problem, tasks related to implementation of the neural network technologies when constructing architecture and mathematical model of the neuroregulator were solved. The mathematical model of the neuroregulator is based on parameters of operation of the physical prototype, construction of the feedback connections for the real-time control (adaptive control) is based on the procedure of training of a recurrent neural network that includes LSTM-cells. Considering the testing results is was found out that recurrent neural networks with LSTM-cells can be successfully used as an approximator of the Q-function for the given problem in the conditions when the observable region of the system states has complex structure.**

***Index Terms*—technological production process, parameters of operation, probabilistic network chart, state indicators, methods of adaptive control, neural network, LSTM, Q-learning**

I. Introduction

The modern analysis of the research in the field of controlled production systems demonstrates that the problem of determining the parameters of real-time operation for such objects of study arises primarily in the cases when they involve production of complex technical items, which require precision of production methods and high labor productivity.

During that process a multi-criteria control optimization problem is considered. It sets strict requirements for quality and algorithmic execution of the production process, minimization of the human factor effects on the implementation quality of the technological production cycle, prevention of the occurrence of technogenic emergencies. Such situation is specific for the robotic production systems, that operate under control of software and hardware controllers, which administer the operation of the technological cycle control system in accordance with the implemented programs.

However, the occurrence of emergency situations due to hardware failures, random external control influences, including the human factor, leads to deviation of the operation parameters of the production system from the intended values. This leads to the necessity of adjustment of the control parameters in real-time, based on the neuroregulator models that operate within the means of software and hardware pairing with the technological production cycle.

The formalization of the control structure for technological production cycle is based on the research results in the field of the analysis of operation of probabilistic technological systems, that include a technological production process as a controlled object. The technological production process is characterized by a relatively slow speed of performance of technological operations, which are interconnected within the cycle, and it's structure is defined by a simulation model of the probabilistic network chart[1][2].

Special AI models such as artificial neural networks (ANNs) have unique qualities, can be used as universal approximators[3], and were able to produce in the recent years a variety of impressive results in different kinds of control tasks[4][5][6], including complex decision-making tasks[7][8], and being able to reach human level performance in some of them[9][10].

While the idea of applying ANNs to the control problems is not new[11][12][13][14], some of the recent interesting developments in the field are related to implementation of deep reinforcement learning methods[5]. The possibilities of application of such methods to the considered problem of constructing an effective controller model for the technological production process are explored in this paper.

II. Parameters, statistics and responses of the control of technological production process

Operational interaction of the system components and the technological production cycle that operates in real-time is carried out based on constant monitoring of the hardware unit states and the control parameters through the registers-indicators and special technical pairing means.

The system of software and hardware pairing for the operational control for the technological production cycle consists

of a simulation model of the technological production process, which has a structure defined by a probabilistic network chart; a specialized decision making system, which performs analysis and control of the current developments in the operating environment of the technological production cycle; a control block, that is used as a mediator between the decision making system and the neuroregulator model. The scheme of the system is shown in Fig.1.

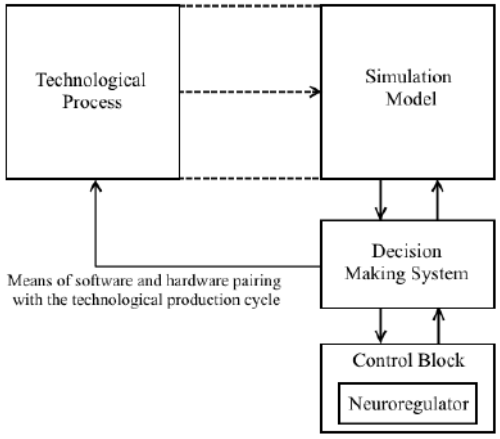

Figure 1. Scheme of the software and hardware system for the operational control in the described formalization.

Operational interaction of the system components is implemented through the parameters and variables of the simulation model of the control system: reliability characteristics for the hardware units operation G^*_{rh}; indicators of the current hardware units states ind^*_{rh}, that accumulate each unit's time before failure; indicators of the emergency occurrences due to hardware failure π^*_{abrh}, that have effect on the configuration of the technological cycle and the control system; current values U^*_{fh} of the variables for the control of the technological process; values of the control variables adjustments ΔU^*_{fh} that are based on the output of the neuroregulator model; parameters of the operational conditional of the technological process Z^*_{fh}, that depend on the values of the control variables U^*_{fh}. The decision making system sends three types of signals to the hardware and the control registers of the technological process: corrective adjustments ΔU^*_{fh} for the control variables of the technological production process; signals α^*_{rh} that allow to switch to the backup hardware or signals to initiate maintenance or repairs for hardware units.

III. Features of the technological process control structure based on a neuroregulator model

In this section the technological and operational control of the simulation model is described, as well as the features of the control structure. For the implementation of the simulation model in the described formalization, the following parameters are used:

- number of hardware units N;
- list of the K operation stages of the technological process, that consists of pairs $(t_k, \{R(M_{ik}\})$, where t_k is duration of the stage, $R(M_{ik})$ - acceptable mode of operation for the i-th unit at this stage;
- maximum number of the possible simultaneous maintenance jobs for the hardware units S;
- reliability characteristics for the hardware units operation G^*_{rh}, which include:
  1) distributions for the duration of operation till failure for the i-th hardware unit $F_{ir}(t_{wf})$ in the operation mode $R(M_i)$;
  2) distributions for the duration of the required maintenance (repairs) of the i-th hardware unit in the event of failure $F_{if}(t_r)$;
  3) distributions for the duration of the liquidation of the emergency due to failure of the i-th hardware unit $F_{ife}(t_{re})$;
  4) probabilities of the emergency occurrence due to failure of the i-th hardware unit P_{ie}.

The simulation model operates during the given time interval, restarting the production cycle and carrying on before each start the maintenance actions if necessary. In order to make a decision if such actions are necessary, a neuroregulator model is used, which operates based on the current environment state data.

IV. Procedure of neuroregulator generation

When solving this task a group of reinforcement learning algorithms, and particularly Q-learning, is interesting to consider as a basis for the method to determine the optimal control strategy that has feedback connections and takes into account the complex structure of the space of available actions.

Reinforcement learning algorithms assume implementation of an agent capable of observing the environment and manipulating it with actions, as well as the mechanism of numerical evaluation of agent's performance - the reward function[15]. When training a controller for the agent during the reinforcement learning process, an exploration process is involved in order to determine the agent's actions that lead to the highest rewards. Such approaches have a potential to train a controller capable of implementing the optimal strategy.

In this paper reinforcement learning (Q-learning) is used to train a neuroregulator capable of forming a set of maintenance recommendations for the technological production cycle at each point of time.

During the model training the values of the reward function are calculated according to the following rules during each "maintenance-cycle" procedure:

- $r = 0$
- $r = r - REW\_REPAIR\_COST$ – penalty for each maintenance (repairs), initiated due to the agent's decision during the maintenance stage of the given iteration;
- $r = r - REW\_FAILURE\_COST$ – penalty for each hardware unit failure that happened during the cycle stage of the given iteration;

In the case of emergency that happened as a result of a hardware unit failure:

- $r = r - REW\_EMERGENCY\_COST$ – penalty for the emergency that as caused by one of the hardware units failure during the cycle stage of the given iteration.

It is presumed that the penalty value REW_REPAIR_COST is significantly less than the values for $REW\_FAILURE\_COST$ and $REW\_EMERGENCY\_COST$, so that the action of the hardware unit maintenance is more preferable than the situation when the unit fails during the cycle, causing the cycle to stop and possibly causing an emergency. On the other hand, this value is non-zero because the maintenance procedure consumes resources and time and should be used only when necessary.

The choice of the specific numerical values for the parameters of the reward function with regard to the practical requirements for the reliability and cost reduction will have effect on the action selection policy of the agent learned during training. For example shaping the reward function with usage of the calculated cost values may result in agent learning to prefer lower maintenance costs at the expense of higher failure rates.

The idea of Q-learning is to train the agent's controller to approximate Q^* - the function that estimates reward values for the next environment state as a result of the chosen control actions[16][17]. A neural network can be used as an approximator for that function. In this case the training task is to find such values of the trainable parameters of the network, so that the approximate function Q is close enough to the optimal function Q^* and therefore the determined policy of action selection π is close enough to being optimal[9]:

$$Q(s,a) \approx Q^*(s,a) = \max_{\pi} E\left[R_t | s_t = s, a_t = a\right] \quad (1)$$

And Bellman equation is true for Q^*:

$$Q^*(s,a) = E(r + \gamma \max_{a'} Q^*(s',a') | s,a) \quad (2)$$

When solving complex real-world problems, the agent often has the whole information about the environment unavailable. In this case the agent that operates based upon a Q approximator calculating values depending only on the current observable environment state may be inefficient when the environment structure has high complexity or is of temporal nature and involves processes, both dependent and independent of the agent's actions. In the described situation it is necessary to use some mechanism of memory for the agent. A recurrent neural network with a hidden state preserved across multiple iterations can be used for such purpose[7][18]. LSTM-cells as a structural component of the neural network architecture allow the network to approximate temporal dependencies stretched over long periods of time[19].

Therefore the neural network model choice for the problem in the described formalization:

DQRN1 – agent that uses a recurrent neural network based on the multi-layer perceptron architecture with LSTM-cells. The current environment state is feed to the network input.

Neural network structure:

1) Dense x16 ReLU;
2) Dense x16 ReLU;
3) Dense x32 ReLU;
4) LSTM x32 ReLU;
5) Dense x6 no activation.

The input vector of the network is formed at each point of time based on values of the indicators ind^*_{rh} of the current state of the hardware units.

The output vector of the network has dimension $N + 1$. The values of the elements of the output vector define a set of maintenance recommendations for the N hardware units. In case value of an element exceeds some given threshold level, it is considered that the model recommends to perform maintenance of the corresponding hardware unit. In case the last element's value exceeds the threshold, it is considered that the model recommends not to perform any maintenance at this time.

Based on the values of the elements of the output vector of the model, the signals for performing the maintenance (repairs) for the corresponding hardware units α^*_{rh} are determined.

To implement model training based on Q-learning algorithm, a custom environment was developed that includes a simulation model of a technological process in the given formalization. The environment was written in Python language, using simpy library for simulations, tensorflow and keras libraries for the neurocontroller model.

In this paper experience replay was used during training[9] - the biologically inspired mechanism to select randomly and demonstrate during training the previous experience of agent's interactions with the environment. The updates of the Q values were performed after each run of a simulation based on all the saved experiences. The neural network is trained on sequences of 16 timesteps.

Training procedure:

1) Agent acts during a single simulation run (multiple runs of the production cycle, determined by a set of K stages during the given period of model time). The agent receives the current observable state of the environment and acts according to the chosen exploration strategy that determines how the random actions are selected. In this paper the softmax exploration strategy was used[15]:
$$p_t(\alpha^*) = \frac{exp(Q_t(\alpha^*)/\tau)}{\sum_i exp(Q_t(\alpha_i)/\tau)} \quad (3)$$
2) Experiences of agent's interactions with the environment are saved as sequences $\{(s_t; a_t; r_t; s_{t+1})\}$ to memory.
3) The experiences to be used for the next update of the trainable parameters of the neural network are sampled from the memory. According to the selected training parameters, these experiences are sampled party from the whole memory randomly, and partly from the latest sequences.
4) The updated Q values are calculated based on the following observable states and rewards from the experiences:
$$Q(s,a^*) = r + \gamma \max_{a} Q(s,a) \quad (4)$$

5) The neural network is trained on a set of size 32 for 1 epoch using the RMSprop algorithm[20]. The trainable parameters are updated after presenting each set of 4 sequences.
6) Go back to 1) and restart the simulation.

The training continues until the simulation number limit is reached.

V. Parameters of the simulation model used in the experiments

One cycle run under normal condition lasts for 48 ticks of simulation time. One simulation run lasts for 64*48 = 3072 ticks of simulation time.

Number of hardware units:

$N = 5$

Modes of operation $R(M_i)$:

operating/not operating – for all hardware units.

Number of maintenance jobs that can be performed simultaneously:

$S = 3$

$K = 2$ cycle stages:

$\{(4, (1, 0, 0, 0, 0)), (44, (1, 1, 1, 1, 1))\}$

First stage - operation of only the first hardware unit during 4 ticks of simulation time. The second stage - all of the units operate for 44 ticks of simulation time.

Costs:

- $CYCLE\_NON\_OPERATING\_COST = 20$ – non-operation of a production process for one simulation tick;
- $FAILURE\_DURING\_CYCLE\_COST = 150$ – hardware unit failure during the operation of the cycle;
- $EMERGENCY\_DURING\_CYCLE\_COST = 1000$ – emergency during the operation of the cycle caused by a hardware unit failure.

Distributions and probabilities:

- distributions for the operation time before failure of the i-th hardware unit$F_{ir}(t_{wf})$:
  1) uniform $U(150; 200)$;
  2) normal $N(\mu = 1000, \sigma^2 = 250)$;
  3) normal $N(\mu = 350, \sigma^2 = 50)$;
  4) uniform $U(300; 500)$;
  5) normal $N(\mu = 800, \sigma^2 = 300)$.
- distributions for the duration of the required maintenance (repairs) of the i-th hardware unit in the event of failure $F_{if}(t_r)$:
  1) uniform $U(3; 5)$;
  2) uniform $U(2; 4)$;
  3) uniform $U(3; 6)$;
  4) uniform $U(5; 10)$;
  5) uniform $U(2; 3)$.
- probabilities of the emergency occurrence due to failure of the i-th hardware unit P_{ie}: $\{0.05; 0.25; 0.1; 0.01; 0.2\}$
- distributions for the duration of the liquidation of the emergency due to failure of the i-th hardware unit $F_{ife}(t_{re})$:
  1) uniform $U(10; 15)$;
  2) uniform $U(8; 10)$;
  3) uniform $U(5; 8)$;
  4) uniform $U(25; 35)$;
  5) uniform $U(4; 6)$.

Values for the reward parameters:

1) $REW\_FAILURE\_PENALTY = 100.0$
2) $REW\_EMERGENCY\_PENALTY = 200.0$
3) $REW\_REPAIR\_PENALTY = 10.0$

VI. Training results

The model was trained as described above during the 2000 simulation runs.

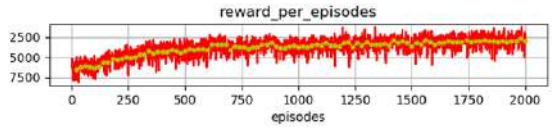

Figure 2. Total agent's reward for one simulation during training.

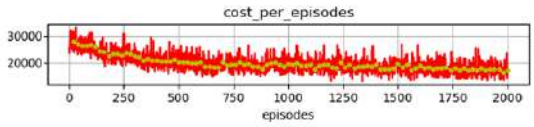

Figure 3. Total costs for running the technological cycle for one simulation during training

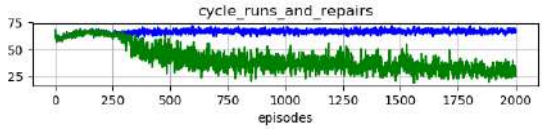

Figure 4. Total number of repairs (in green) and cycle restarts (in blue) for one simulation during training.

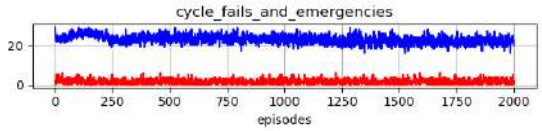

Figure 5. Total number of hardware failures (in blue) and emergencies (red) for one simulation during training.

According to the statistics gathered during training the following can be noticed:

- increase in the agent's reward (Fig.2);
- tendency towards the decrease in costs for running the cycle (Fig.3);
- decrease in the number of repairs for the hardware units (Fig.4);
- tendency towards the decrease in the number of hardware failures (Fig.5);
- increase in the average operation time for the cycle after start (Fig.6);

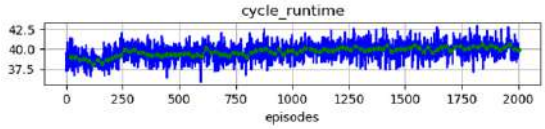

Figure 6. Average cycle operation time for one simulation during training.

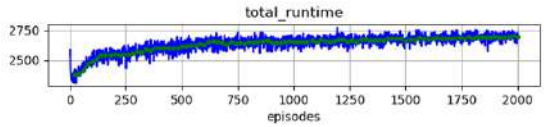

Figure 7. Total cycle operation time for one simulation during training.

- increase in the total cycle operation time (Fig.7).

It is interesting to compare the performance of the model by comparing it to some baseline models that do not implement complex action selection policies like the trained model does. The examples of such models can be a controller that never recommends maintaining the hardware units (Baseline-Zero) and a controller that implements random action selection policy (Baseline-Random). On Fig.8 and Fig.9 histograms for the distributions of costs and durations of normal cycle operation are given for 5000 test runs of the simulation.

For Baseline-Zero median costs value is 19455.82, median operation time - 2686.46. For Baseline-Random median costs value is 18002.09 and median operation time is 2737.25.

The same histograms for DQRN1 agent are shown on Fig.10. For this model the median costs value is 17115.98 and median operation time is 2697.59.

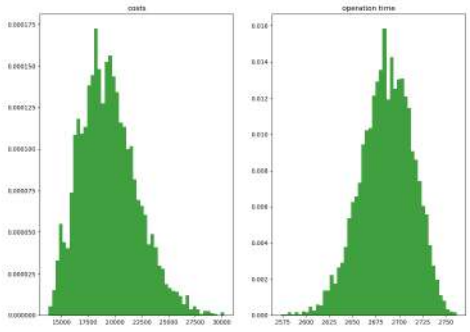

Figure 8. Histograms for the distributions of costs and cycle normal operation time for the controller Baseline-Zero during testing.

It is particularly effective to use the software and hardware system in the cases when time intervals τ_{SOB} between the emergencies in the slowly operating technological process are large enough to allow the operative control ($\tau_{SOB} > T_{kph}$, where T_{kph} - critical time of realization of the process, that was estimated using the simulation model).

Another important purpose of the system is realization of multiple series of simulation experiments implementing Monte-Carlo methods in order to determine how the global

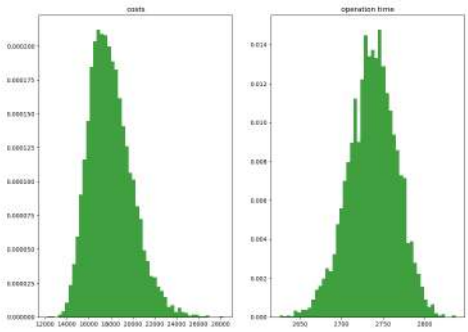

Figure 9. Histograms for the distributions of costs and cycle normal operation time for the controller Baseline-Random during testing.

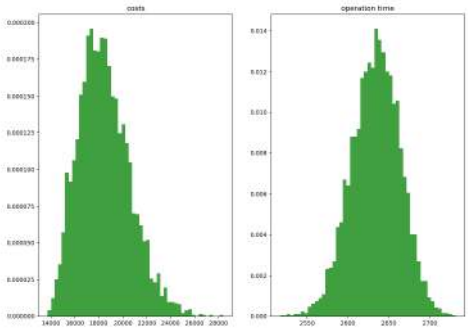

Figure 10. Histograms for the distributions of costs and cycle normal operation time for the controller DQRN1 during testing.

variables $Z_{fh}(t_0)$ and $U_{fh}(t_0)$ change with model time. Those dependencies can be then used to compare the model values to the real values: $Z_{fh}(t_0)$ to $Z^*_{fh}(t)$ and $U_{fh}(t_0)$ to $U^*_{fh}(t)$, where t_0 and t - respectively moments of model time and moments of real time in the technological production cycle.

In case when for all the elements of these vectors the absolute values of difference are within an acceptable error margin (5)(6), the simulation model is considered to be adequate in the dynamics of implementation of the control for the technological production cycle using the neuroregulator model.

$$\left|Z_{fh}(t_0) - Z^*_{fh}(t)\right| < \delta \tag{5}$$

$$\left|U_{fh}(t_0) - U^*_{fh}(t)\right| < \delta \tag{6}$$

VII. Conclusion

The proposed technology of adaptive control for the technological cycle using the neuroregulator models allows:

- to chose a rational set of resources for the technological cycle as well as a set of hardware units that will allow a required level of hardware reliability for the normal operation of the production process without emergencies;

- operatively adjust the characteristics of realization of the control process to reach the performance of the technological cycle within the acceptable margins of change for the technological parameters;
- to address the problem of estimating the costs of implementing the technological cycle with the given set of resources and hardware available at the facility.

References

[1] Maximey, I.V., Smorodin, V.S., Demidenko, O.M. *Development of simulation models of complex technical systems*, Gomel, F. Skorina State University, 2014. 298 p.

[2] Maximey, I.V., Demidenko, O.M., Smorodin, V.S. *Problems of theory and practice of modeling complex systems*, Gomel, F. Skorina State University, 2015. 263 p.

[3] Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 1989, vol. 2, no 4, pp. 303–314/

[4] Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., Zieba, K. End to End Learning for Self-Driving Cars. Available at: http://arxiv.org/abs/1604.07316 (accessed 2019, Nov)/

[5] Francois-Lavet, V. et al.: An Introduction to Deep Reinforcement Learning. Available at: http://arxiv.org/abs/1811.12560 (accessed 2019, Nov)/

[6] Levine, S., Finn, C., Darrell, T., Abbeel, P. End-to-End Training of Deep Visuomotor Policies. *J. Mach. Learn. Res.*, 2015, vol. 17, pp.39:1-39:40/

[7] Hausknecht M., Stone P. Deep recurrent q-learning for partially observable mdps. *2015 AAAI Fall Symposium Series*, 2015./

[8] Lample, G., Chaplot, D. S. Playing FPS Games with Deep Reinforcement Learning. *AAAI'17 Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp.2140-2146/

[9] Mnih V., Kavukcuoglu K., Silver D., Rusu A., Veness J., Bellemare M., Graves A., Riedmiller M., Fidjeland A., Ostrovski G., Petersen S., Beattie C., Sadik A., Antonoglou I., King H., Kumaran D., Wiestra D., Legg S., Hassabis D. Human-level control through deep reinforcement learning. *Nature*, 2015, vol. 518, no 7540, pp.529–533/

[10] Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Driessche, G.V., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T.P., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D. Mastering the game of Go with deep neural networks and tree search. *Nature*, 2016, vol. 529, pp.484-489/

[11] Hagan, M.T., Demuth, H.B. Neural networks for control. *Proceedings of the American Control Conference*, 1999, vol. 3, pp. 1642—1656/

[12] Narendra, K.S., Mukhopadhyay, S. Adaptive Control Using Neural Networks and Approximate Models. *IEEE Transactions on Neural Networks*, 1997, vol. 8, pp. 475-485/

[13] Narendra, K.S., Parthasarathy, K. Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Transactions on Neural Networks*, 1990, vol. 1, pp. 4-27/

[14] Lin, L.-J. *Reinforcement learning for robots using neural networks*, Pittsburgh, Carnegie Mellon University, 1992./

[15] Sutton, R. S., Barto, A. G. *Reinforcement Learning: An Introduction*, Cambridge, The MIT Press, 1998./

[16] Watkins, C. J. C. H., and Dayan, P. Q-learning. *Machine Learning*, 1992, vol. 8, no 3-4, pp. 279–292/

[17] Watkins, C. J. C. H. *Learning from Delayed Rewards*, Cambridge, University of Cambridge, 1989/

[18] Bakker, B. Reinforcement Learning with Long Short-Term Memory, *NIPS*, 2001/

[19] Hochreiter, S., Schmidhuber, J. Long short-term memory. *Neural Computation*, 1997, vol. 9, no 8, pp.1735–1780/

[20] Graves A. Generating Sequences With Recurrent Neural Networks. Available at: https://arxiv.org/abs/1308.0850 (accessed 2019, Nov)/

УПРАВЛЕНИЕ ТЕХНОЛОГИЧЕСКИМ ЦИКЛОМ ПРОИЗВОДСТВА НА ОСНОВЕ МОДЕЛИ НЕЙРОКОНТРОЛЛЕРА

Смородин В.С. - ГГУ им. Ф.Скорины, кафедра математических проблем управления и информатики

Прохоренко В.А. - ГГУ им. Ф.Скорины, кафедра математических проблем управления и информатики

Цель работы, результаты которой представлены в рамках данной статьи, состояла в разработке методики адаптивного управления технологическим циклом производства на базе программно-аппаратной системы, содержащей индикаторы состояния оборудования, параметры функционирования технологического цикла, имитационную модель вероятностного технологического процесса и встроенную систему принятия решений. Оперативное взаимодействие компонентов программно-аппаратной системы и построение обратных связей по управлению реализуется с помощью параметров управления и переменных имитационной модели на основе результатов работы модели нейрорегулятора. Для достижения поставленной цели были решены задачи, связанные с применением нейросетевых технологий при построении архитектуры и математической модели нейрорегулятора. При этом математическая модель нейрорегулятора разработана на основе параметров функционирования физического прототипа, а построение обратных связей по управлению в режиме реального времени (адаптивного управления) основано на процедуре обучения рекуррентной нейронной сети, построенной с использованием LSTM-блоков. С учетом полученных результатов установлено, что рекуррентные сети с LSTM-модулями могут успешно применяться в качестве аппроксиматора Q-функции агента для решения поставленной задачи в условиях, когда наблюдаемая область состояний системы имеет сложную структуру.