UDC 004.7:681.5 **doi:** https://doi.org/10.20998/2522-9052.2026.2.05

Kyrylo Rukkas[1], Anastasiia Morozova[1], Iryna Zaretska[1], Yevheniia Andriichenko[2], Dmytro Chumachenko[3]

[1] V.N. Karazin Kharkiv National University, Kharkiv, Ukraine
[2] CAMPUS 02 University of Applied Sciences, Graz, Austria
[3] University of Waterloo, Waterloo, Canada

DEVELOPMENT OF A MULTI-AGENT SYSTEM FOR DYNAMIC SDN CONTROL

Abstract. **Relevance.** Currently, the volume of transmitted information and the quality requirements for its transmission are increasing. Recently, Software-Defined Networking (SDN) technology has been gaining popularity; however, challenges arise related to the uncertainty of the SDN network state and its elements, as well as the integration of various data streams that have different quality delivery requirements. Therefore, the task of utilizing an intelligent multi-agent system (MAS) for managing SDN networks becomes relevant. **The object of research** is the process of managing SDN networks. **The subject of the research** is models and methods for managing SDN networks. **The purpose of this paper** is to develop a model for the interaction of intelligent agents to ensure the effective functioning of multi-agent systems (MAS) in dynamic management of SDN. **Research results.** Using the analytical framework of probabilistic temporal graphs, mathematical models have been developed for two options for coordinating agents in the dynamic management of SDN networks: a system with a coordinating agent and a self-regulating MAS. Based on the analysis of the obtained probabilistic temporal characteristics of various coordination options, it has been established that self-regulating MAS are advisable in situations where agents have sufficient knowledge to solve the overwhelming majority of emerging tasks, where these solutions are highly likely to be correct, the number of agents in the system is small, and there is a high probability of effective control over the decisions made.

Keywords: SDN; network; computer network management; multi-agent systems; probabilistic-temporal graphs.

Introduction

Relevance. To date, both the volume of transmitted data and the Quality of Service (QoS) requirements have been steadily increasing. Recently, Software-Defined Networking (SDN) has gained significant momentum, offering enhanced flexibility for dynamic management processes. However, dynamic control faces several challenges, including the uncertainty of the network state and its individual elements, as well as the integration of diverse traffic flows with varying delivery requirements. One promising approach to addressing these issues is the implementation of an intelligent Multi-Agent System (MAS). Consequently, the development of a multi-agent system for dynamic SDN control has become a highly relevant task.

An overview of scientific works. In [1], a multi-agent system for the dynamic control of an Integrated Services Digital Network (ISDN) is proposed. Various approaches to addressing specific dynamic control problems using artificial intelligence methods are presented in [2–5]. Most implementations of intelligent systems focus on solving individual dynamic management tasks, such as routing [2], flow control [3, 4], and network access control [5], among others. Studies [6, 7] investigate the integration of multi-agent systems and AI to optimize distributed SDN management, enabling greater scalability and robustness. These works demonstrate that a multi-agent approach is essential for the efficient dynamic management of SDN.

Setting objectives. Therefore, it becomes necessary to develop the corresponding mathematical models. Based on this, **the goal of this paper** is to develop an agent interaction model that ensures the efficient operation of a multi-agent system for dynamic SDN control.

SDN Architecture and Features

SDN networks have a three-tier architecture as shown in Fig. 1 [8].

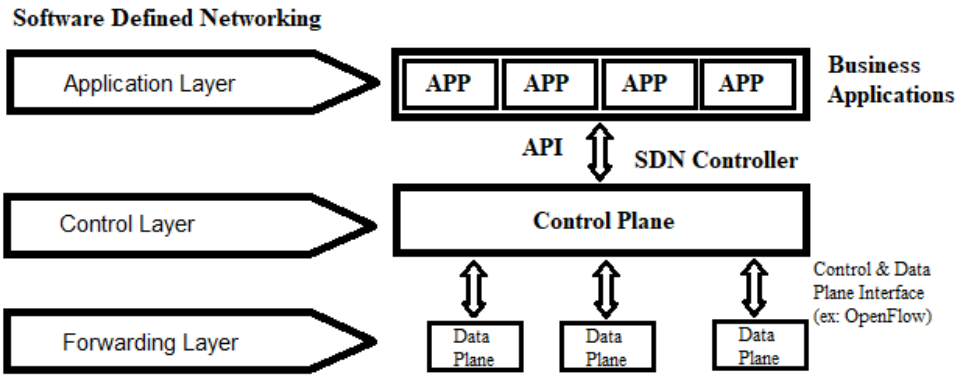


Fig. 1. Block diagram of an SDN network

At the lower level, user data is transmitted via network switches. The second level is responsible for managing the operation of these switches. The top level enables the implementation of network applications and

services. Interaction between the Data and Control planes is facilitated through various southbound protocols, such as OpenFlow and NETCONF [9, 10]. Each switch must contain one or more flow tables, a group table, and support an OpenFlow channel for communication with a remote controller (server). Each flow table in the switch contains a set of flow entries or rules. Each such entry consists of match fields, counters, and a set of instructions.

Instructions associated with each table entry describe actions related to packet forwarding, header modification, group table processing, pipeline processing, and forwarding the packet to a specific switch port. The instructions also define the rules for modifying counters, which can be used to collect various statistics.

If no matching rule is found in the first table, the packet is encapsulated and sent to the controller.

The controller then generates an appropriate rule for packets of this type and installs it on the switch (or on a set of switches under its control); alternatively, the packet may be dropped depending on the switch configuration. In OpenFlow, data management is performed at the flow level rather than at the level of individual packets [11].

MAS Features

Managing such a network is a highly complex process. One promising approach to effectively addressing this challenge is the use of multi-agent systems [12, 13]. This approach allows the dynamic SDN management task to be decomposed into subtasks, which are then assigned to individual agents.

During the operation of a MAS, the actions of one agent impact the performance of others. This may lead to a situation where an agent's efforts to maximize its own local efficiency result in a performance decline for other agents and, consequently, the system as a whole [14].

To improve MAS performance, agents must coordinate their actions. There are two primary approaches to agent coordination [15]:

1) use of a coordinating agent;

2) implementation of a self-regulation (self-organization) method.

In the first case, a coordinating agent is present in the system to identify and resolve conflicts arising during agent interaction.

When using a self-regulation mechanism, agents must coordinate their work to achieve their individual goals while simultaneously contributing to the overall efficiency of the system. The self-regulation method assumes that agents apply a specific set of rules in various situations, which in most cases ensures high performance of the MAS.

To analyze these coordination options in various scenarios, mathematical models were developed for both coordination methods using the probabilistic-temporal graph method [16].

Regardless of the chosen option, the coordination process involves two common stages:

1) information delivery from the coordinating agent to the managed agents and vice versa;

2) execution of subtasks by the managed agents.

Let us assume that error-free information delivery is described by the function $f_1(x)$, while delivery with errors is described by $f_2(x)$.

At the stage of executing subtasks, the managed agents utilize the information received from the coordinating agent. Let t_d denote the time required for an agent to solve its subtask. The correctness of the solution depends on whether the necessary rules are present in the managed agent's knowledge base. The probability of these rules being present is denoted by P_{kb}.

The process of a single managed agent solving its subtask is described using the PTG (probabilistic-temporal graph) shown in Fig. 2 (a). This graph is transformed into the form shown in Fig. 2 (b). Here, the transition function from the initial state (before the subtask is solved) to the state corresponding to its correct solution is defined as:

$$f_3' = (1 - P_{kb})P_{rd1}z^{t_d} + P_{kb}P_{rd2}z^{t_{d1}}, \qquad (1)$$

where P_{rd1} – the probability of the agent correctly solving its subtask if the appropriate rules are absent from the knowledge base; P_{rd2} – the probability of the agent correctly solving its subtask if the appropriate rules are present in the knowledge base; t_d – the time taken by the agent to solve its subtask when no appropriate rules are found in the knowledge base; t_{d1} – the time taken by the agent to solve its subtask when the appropriate rules are available in the knowledge base.

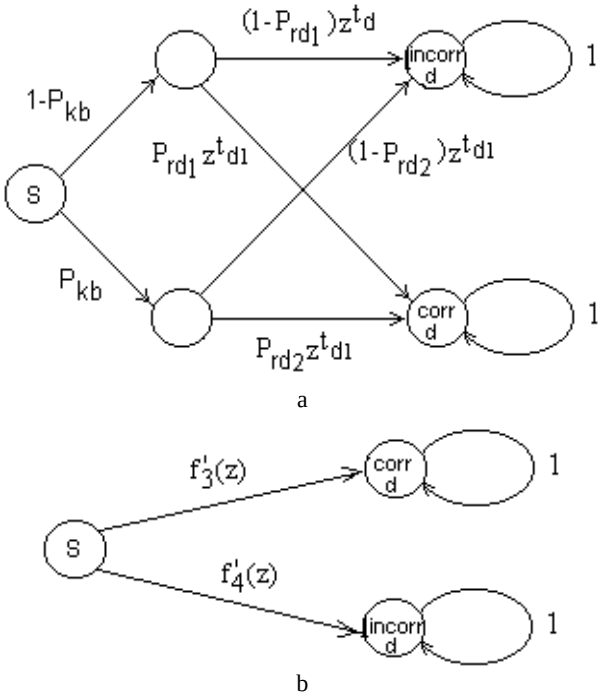


Fig. 2. PTG of the subtask execution process by a single agent

The transition function from the initial state (before the subtask is solved) to the state corresponding to its incorrect solution is defined as:

$$f_4' = (1 - P_{kb})(1 - P_{rd1})z^{t_d} + P_{kb}(1 - P_{rd2})z^{t_{d1}}. \qquad (2)$$

If the system consists of two managed agents, the process of these agents executing their subtasks is described by the graph shown in Fig. 3.

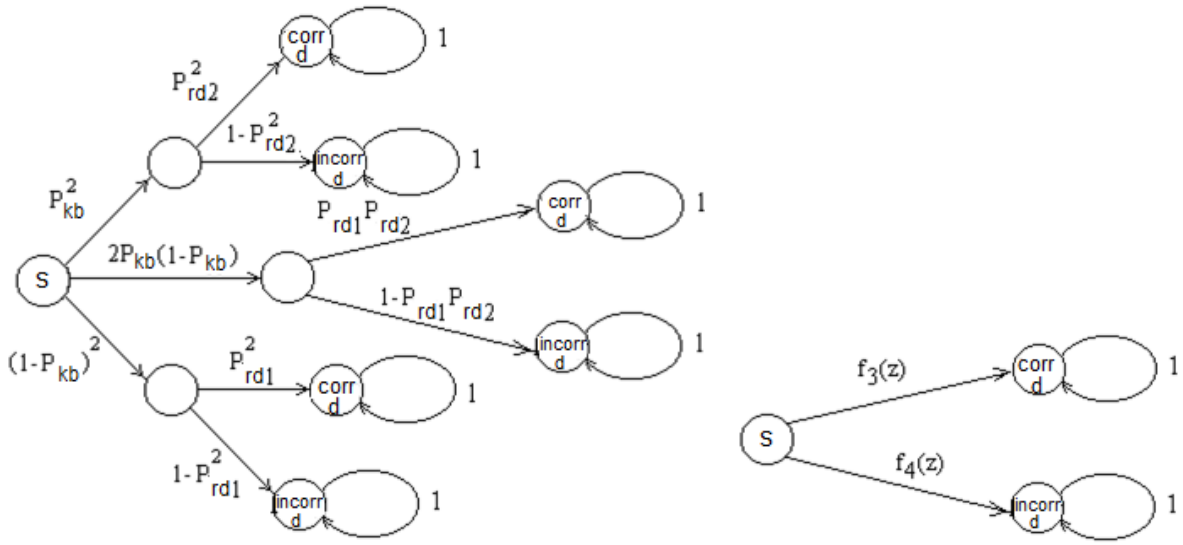


Fig. 3. PTG of the subtask execution process by two agents

For systems with $n > 2$ managed agents, these graphs will exhibit an even more complex structure [17]. An analysis of the transition functions of interest for these graphs allows for the derivation of generalized transition functions for the system into states of correct and incorrect subtask execution.

These functions are valid for systems with an arbitrary number of managed agents n:

$$f_3(z) = (P_{kb}P_{rd2} + (1 - P_{kb})P_{rd1})^n z^{t_d}; \quad (3)$$

$$f_4(z) = 1 - f_3(z). \quad (4)$$

Below, we describe the specific features of the mathematical models for the two coordination methods.

Mathematical model of MAS with a coordinating agent

The probabilistic-temporal graph describing the operational process of MAS with a coordinating agent is shown in Fig. 4.

At the beginning, the coordinating agent sends information to the managed agents regarding the subtasks they need to execute.

If this information is delivered without errors, the process transitions to State 1. Otherwise, a transition to State 2 occurs.

Upon receiving the specified information from the coordinating agent, the managed agents proceed to the stage of executing their subtasks.

If the information proves to be outdated, the agents will incorrectly solve their subtasks after a time t_d (transition from State 2 to State 1, and then to State 3). The probability of information being outdated is P_{ood}.

If the information is not outdated and the agents solve their subtasks correctly, a transition to State 7 occurs (the transition function is $f_3(z)$). In the event of an incorrect solution by the agents, a transition from State 4 to State 3 occurs, with the transition function defined as $f_4(z)$.

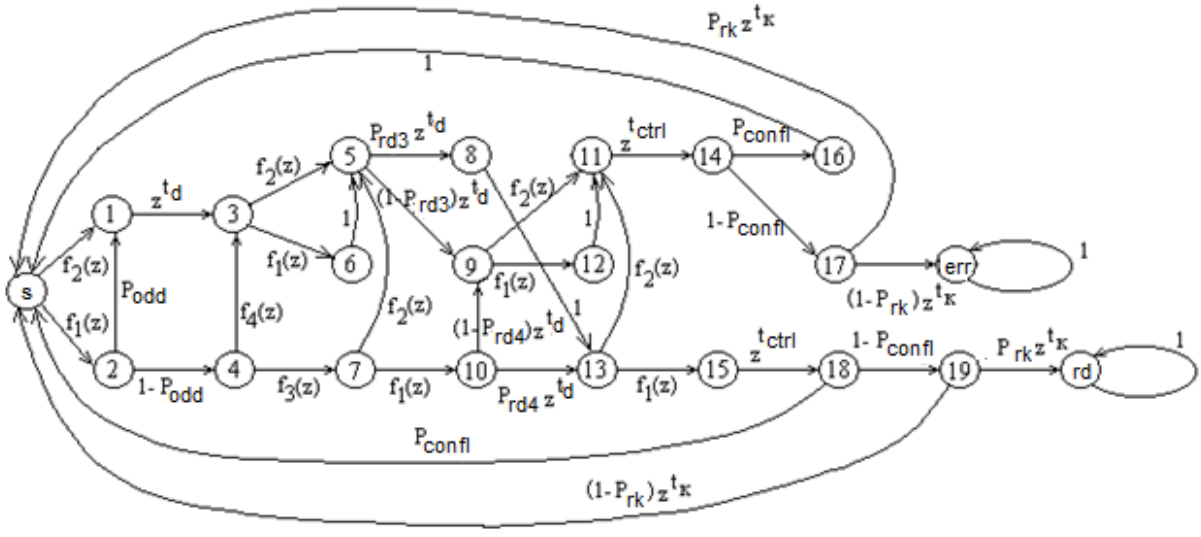


Fig. 4. PTG of the operational process of a MAS with a coordinating agent

After the agents complete the subtask execution process, the resulting solutions are sent to the coordinating agent. Information regarding the obtained solutions may be delivered correctly, which corresponds to States 6 and 10, or with an undetected error, corresponding to State 5. The transition functions for these states are $f_1(z)$ and $f_2(z)$, respectively. If the subtasks are solved correctly and the solutions are transmitted without distortion, the controlling agent will correctly solve the overall control task with probability P_{rd4} in time t_d. In this case, a transition occurs from State 10 to State 13. However, if the agents have solved their subtasks incorrectly, or if the information is transmitted to the controlling agent with distortion, the controlling agent will correctly solve the overall control task with probability P_{rd3} in time t_d.

In this case, a transition to State 13 occurs from State 5. If the controlling agent solves the overall task incorrectly, a transition to State 9 occurs.

Subsequently, the information regarding the decision made by the coordinating agent is delivered to the managed agents. If this information is delivered without errors, a transition to either State 12 or State 14 occurs, depending on the previous states.

If the information is delivered with undetected errors, a transition to State 11 occurs. The functions for these transitions are equal to the previously determined functions $f_1(\mathrm{z})$ and $f_2(\mathrm{z})$, respectively. Based on the information received from the coordinating agent, the managed agents execute a control action. If the coordinating agent's decision is incorrect or contains undetected errors, the managed agents will perform an incorrect control action. In this case, a transition occurs from State 11 to State 15. Conversely, if the managed agents receive a correct and error-free decision from the coordinating agent, they will execute a correct control action (transition from State 14 to State 18).

The time required to execute the control action is denoted by t_{ctrl}.

Executing a control action may lead to a conflict with a probability P_{confl}. In this case, a transition to the initial state occurs from State 16 or State 18. This transition is triggered by the need to solve the task again to resolve the conflict. If the execution of an incorrect control action by the agents does not result in a conflict, a transition from State 15 to State 17 occurs. Conversely, if no conflict occurs as a result of a correctly executed control action, a transition occurs from State 18 to State 19.

After the respective agents execute the control actions, the coordinating agent monitors the correctness of the control result. Let t_k denote the monitoring time, and P_{rk} denote the probability of correct monitoring.

If an incorrect control action did not cause a conflict and the control result was monitored correctly, the task will be re-solved (transition from State 17 to the initial state). However, if incorrect monitoring occurs in this case, a transition from State 17 to the "err" state occurs; this means the process terminates, and the overall task is solved incorrectly by the agent system. If a correct control result is achieved and correct monitoring is performed, a transition from State 19 to the "rd" state occurs, signifying the completion of the process and the successful solution of the overall task.

If the control result is monitored incorrectly, a transition from State 19 to the initial state occurs, and the task is solved anew.

By performing equivalent transformations, this graph can be reduced to the form shown in Fig. 5.

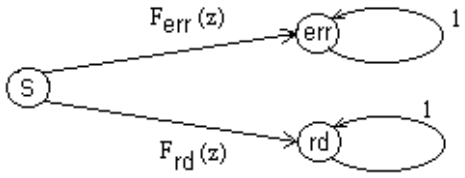


Fig. 5. Transformed PTG of the MAS operational process with a coordinating agent

Here, the functions $F_{rd}(z)$ and $F_{err}(z)$ describe transitions from the initial state to the states of correct and incorrect task solution by the MAS with a coordinating agent, respectively. Due to their complexity, the derived expressions for these functions are not provided here.

These expressions allow for an analysis of how the probability and time of a correct MAS solution depend on various parameters, which helps in developing recommendations for applying this coordination option in different scenarios.

Mathematical Model of Self-Regulating MAS

The probabilistic-temporal graph describing the task execution process of self-regulating MAS is shown in Fig. 6.

In the initial state, the controlling agent sends information to the managed agents regarding the subtasks they need to execute. If the information is received with an undetected error, a transition from the initial state to State 1 occurs.

The function of this transition is $f_2(\mathrm{z})$. If the information is received without distortion, a transition from the initial state to State 2 occurs. The function of this transition is $f_1(\mathrm{z})$. The received information used by the managed agent to solve its subtask may prove to be outdated with a probability P_{ood}. If this information is outdated, the managed agent will incorrectly solve its subtask after time t_d, and after time t_{ctrl}, it will perform an incorrect control action (transition from State 2 to State 3). If the information is not outdated, a transition from State 2 to State 4 occurs. If the managed agent solves its subtask correctly, it will execute a correct control action. In this case, a transition from State 4 to State 7 occurs. Otherwise, the transition leads to State 8. If the execution of the control action results in a conflict (the probability of this event is P_{confl}), the process returns to the initial state.

If the execution of an incorrect control action does not result in a conflict, a transition to State 6 occurs. If no conflict arises following a correctly executed control action, the process transitions to State 9.

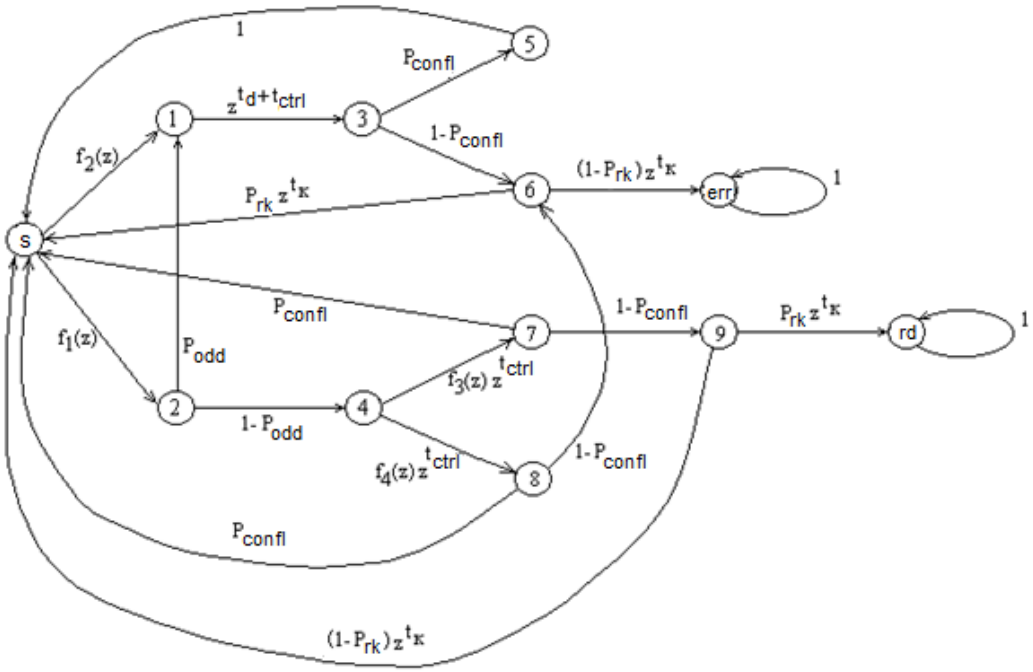


Fig. 6. PTG of the operational process of a self-regulating MAS

Through correct monitoring, it may be determined that the system has performed an incorrect control action (transition from State 6 to the initial state). In this case, the probability of correct monitoring is P_{rk}, and the monitoring time is t_k.

If incorrect monitoring fails to detect an erroneous control action, the solution process terminates, and the task is considered solved incorrectly (transition to the "***err***" state).

If a correct control result is achieved and correctly monitored, a transition from State 9 to the "***rd***" state occurs, signifying the completion of the process and the successful solution of the overall task by the agent system. If a correct control action did not cause a conflict but was monitored incorrectly, a transition from State 9 to the initial state occurs, and the task is solved anew.

By performing equivalent transformations, this graph is reduced to the form shown in Fig. 7.

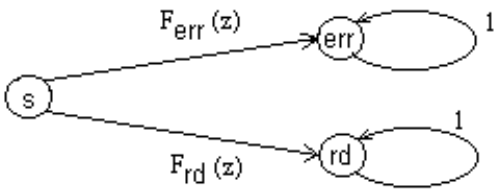


Fig. 7. Transformed PTG of the self-regulating MAS operational process.

Here, the functions $F_{rd}(z)$ and $F_{err}(z)$ describe the transitions from the initial state to the states of correct and incorrect task solution by the self-regulating MAS, respectively. Due to their complexity, the derived expressions for these functions are not provided here. These expressions allow for an analysis of how the probability of a correct solution and the execution time depend on various parameters, which helps in determining the feasibility of implementing the self-regulation mechanism in different scenarios.

Recommendations for Selecting MAS Coordination Options

Based on the proposed mathematical models, the dependencies of the average task solution time T_{avg} and the probability of a correct solution were plotted against the following variables: the probability of an individual agent correctly solving its subtask, the probability of information obsolescence, the probability of knowledge base completeness, the probability of network conflicts, the probability of correct monitoring, and the total number of agents in the system.

A comparative analysis of the probabilistic-temporal characteristics (PTC) revealed that under certain conditions, the option with a coordinating agent is preferable, while in other scenarios, self-regulating systems exhibit superior performance. In the provided graphs, the curves characterizing the operation of a MAS with a coordinating agent are represented by solid lines, whereas the curves characterizing the operation of self-organizing MAS are shown as dashed lines.

From the graphs in Fig. 8(a) and 8(b), it is evident that under conditions where network information obsolescence is slow ($P_{odd} > 0.1$), both options yield approximately equal probabilities of a correct solution; however, the solution time for self-regulating systems is significantly lower.

This can be attributed to the fact that self-organizing systems spend less time on information transfer between agents. Furthermore, an analysis of these graphs shows that when $P_{kb} > 0.95$, the probability of a correct solution remains nearly identical for both coordination options, but the solution time for self-regulating systems is substantially shorter.

Analysis of the graphs in Fig. 9 (a) and 9 (b) indicates that if the probability of an individual agent correctly solving its subtask is high, self-regulating MASs solve the control task faster, given identical probabilities of conflict occurrence. This can be explained by the fact that self-regulating systems do not spend time transmitting information to a coordinating agent.

a b

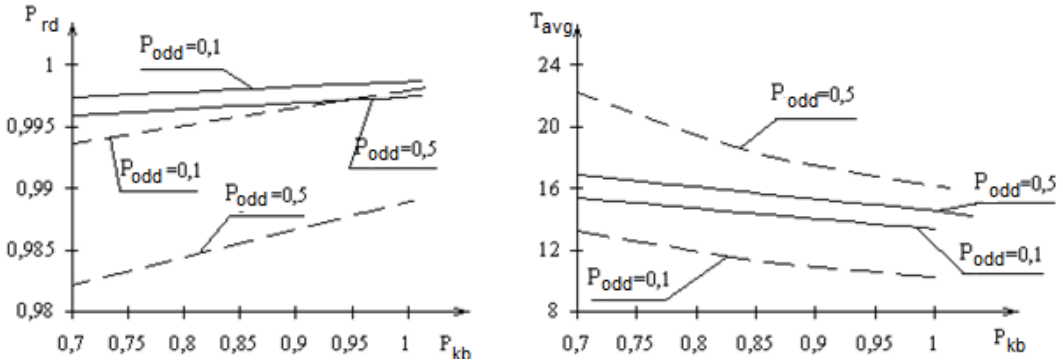
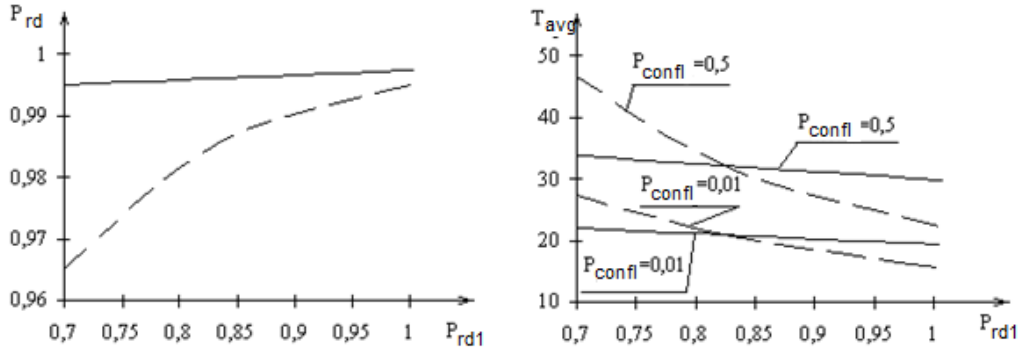


Fig. 8. Dependencies of the probability of a correct MAS task solution P_{rd} and the average solution time T_{avg} on the probability P_{kb}



a b

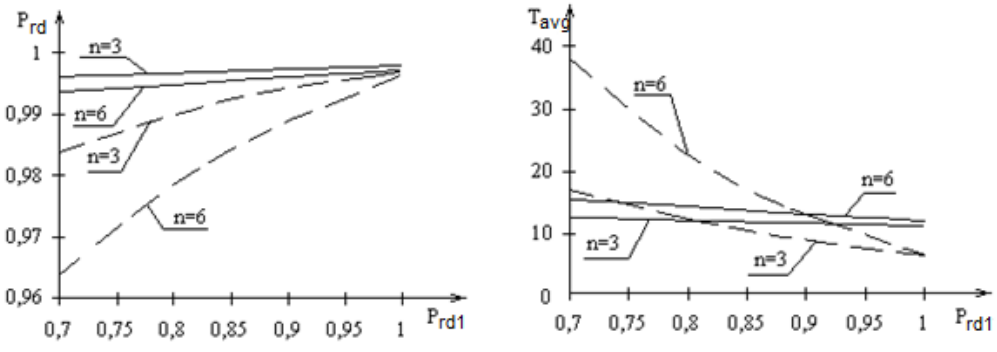
Fig. 9. Dependencies of P_{rd} and T_{avg} on the probability P_{rd1} given $P_{confl} = 0.5$ and $P_{confl} = 0.01$

Investigation of the dependence of the PTC on the number of agents in the MAS (graphs in Fig. 10(a) and 10(b)) shows that as the number of agents increases and when the probability of an individual agent correctly solving its subtask is high ($P_{rd1} > 0.95$), the probability of a correct solution remains approximately the same for both MAS types; however, the solution time for self-organizing MAS is significantly lower.

Furthermore, the number of agents in the system has a much stronger impact on the performance of a self-organizing MAS.

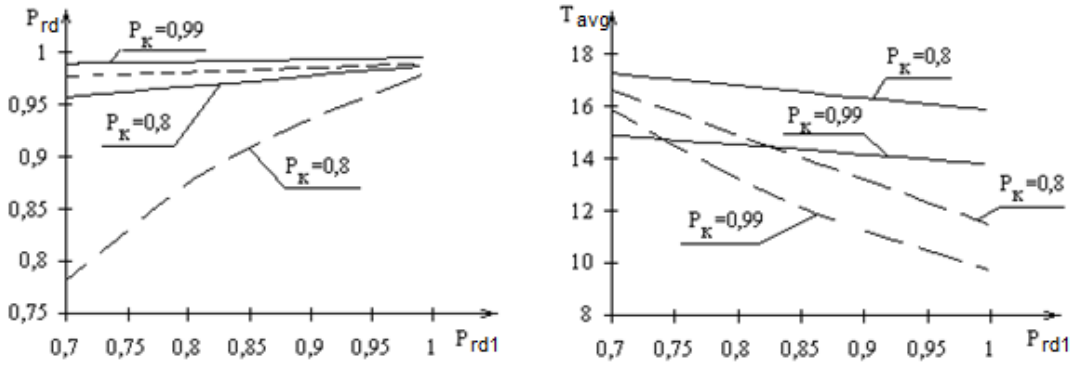
To ensure a high probability of a correct task solution along with a short solution time, it is necessary to limit the number of agents in a self-organizing MAS to 3–4.

An investigation of the dependence of the PTC on the probability of correct monitoring (graphs in Fig. 11(a) and 11 (b)) shows that at high values of correct monitoring probability ($P_k > 0.99$), the probability of correct operation is approximately the same in both cases; however, the task solution time for self-organizing systems is lower.



a b

Fig. 10. Dependencies of P_{rd} and T_{avg} on the probability P_{rd1} when the number of agents $n = 3$ and $n = 6$



a b

Fig. 11. Dependencies of P_{rd} and T_{avg} on the probability P_{rd1} given $P_k = 0.99$ and $P_k = 0.8$

Conclusions

Summarizing the above, it can be concluded that the use of self-organizing MAS is most effective in the following scenarios:

1) when there is a high probability that agents will find the necessary knowledge in their knowledge bases to solve their subtasks.

2) when there is a high probability that agents will solve their subtasks correctly.

3) when the number of managed agents in the system is small (no more than 3–4).

4) when there is a high probability of correct monitoring combined with a high probability of correct subtask execution by the agents.

The recommendations regarding the feasibility of a particular agent coordination option, derived from the analysis described above, should be applied when designing the structure and operational algorithms of multi-agent systems for dynamic SDN management.

Conflicts of interest

The authors declare that they have no conflicts of interest in relation to the current study, including financial, personal, authorship, or any other, that could affect the study, as well as the results reported in this paper.

Use of artificial intelligence

The authors confirm that they did not use artificial intelligence technologies when creating the current work.

REFERENCE

1. Zhong, Qin and Zhang, Zhaohui (2023), “Traffic Load Balancing and Routing Optimization Algorithms in Sdn-Driven Networks”, Available at SSRN: https://ssrn.com/abstract=4576872 or http://dx.doi.org/10.2139/ssrn.4576872
2. Tache, M. D., Păscuțoiu, O., and Borcoci, E. (2024), “Optimization Algorithms in SDN: Routing, Load Balancing, and Delay Optimization”, *Applied Sciences*, vol. 14(14), article number: 5967, doi: https://doi.org/10.3390/app14145967
3. Guo, A., and Yuan, C. (2021), “Network Intelligent Control and Traffic Optimization Based on SDN and Artificial Intelligence”, *Electronics*, vol. 10(6), article number: 700, doi: https://doi.org/10.3390/electronics10060700
4. Hussain, M., Shah, N., Amin, R., Alshamrani, S. S., Alotaibi, A., and Raza, S. M. (2022), “Software-Defined Networking: Categories, Analysis, and Future Directions”, *Sensors*, vol. 22(15), article number: 5551, doi: https://doi.org/10.3390/s22155551
5. Arzo, S. T., Akhavan, Z., Esmaeili, M., Devetsikiotis, M., and Granelli, F. (2022), “Multi-agent-based traffic prediction and traffic classification for autonomic network management systems”, *Future Internet*, vol. 14(8), article number: 230. doi: https://doi.org/10.3390/fi14080230
6. Arzo, S. T., Bassoli, R., and Granelli, F. (2021), “Multi-agent based autonomic network management architecture”, *IEEE Transactions on Network and Service Management*, vol. 18(2), pp. 1092–1106, doi: https://doi.org/10.1109/TNSM.2021.3059752
7. Goteti, D. and Reddy V. K. (2025), “AI-driven routing pipeline in SDN: Federated and multi-agent control”, *Frontiers in Artificial Intelligence*, article number: 1685155. doi: https://doi.org/10.3389/frai.2025.1685155
8. Yuan, T., da Rocha Neto, W., Rothenberg, C. E., Obraczka, K., Barakat, C., and Turletti, T. (2021), “Dynamic controller assignment in SDN through multi-agent deep reinforcement learning”, *IEEE Transactions on Network and Service Management*, vol. 18(2), pp. 1230–1244. doi: https://doi.org/10.1109/TNSM.2020.3047765
9. Kunz, T. and Muthukumar, K. (2017), “Comparing OpenFlow and NETCONF when interconnecting data centers”, *Proc. Int. Conf. on Network Protocols* (ICNP), 2017-October, 8117598, doi: https://doi.org/10.1109/ICNP.2017.8117598
10. Kalinin, Y., Koliesnik, I., Kuchuk, H., Kuchuk, N., Polyashenko, S. and Medvid, M. (2025), “Principles of Diagnostics of Complex Systems using Structural Automaton Algebra and Taking into Account Signal Time Coordination”, *2025 IEEE 6th Khpi Week on Advanced Technology* (Khpiweek 2025), doi: https://doi.org/10.1109/KhPIWeek61436.2025.11288711
11. Aryan, R., Yazidi, A., Bouhoula, A. and Engelstad, P.E. (2025), “A formal technique for automatic resolution of OpenFlow anomalies”, International Journal of Information Security, vol. 24(4), 181, doi: https://doi.org/10.1007/s10207-025-01035-x
12. Rezanov, B., and Kuchuk, H. (2023), “Model of elemental data flow distribution in the Internet of Things supporting Fog platform”, *Innovative Technologies and Scientific Solutions for Industries*, vol. 2023(3), pp. 88–97, doi: https://doi.org/10.30837/ITSSI.2023.25.088

13. Mozhaev, O., Kuchuk, H., Kuchuk, N., Mykhailo, M. and Lohvynenko, M. (2017), “Multiservice network security metric”, *2nd International Conference on Advanced Information and Communication Technologies*, AICT 2017 – Proceedings, pp. 133–136, doi**:** https://doi.org/10.1109/AIACT.2017.8020083
14. Kuchuk, H., Kalinin, Y., Dotsenko, N., Chumachenko, I. and Pakhomov, Y. (2024), “Decomposition of integrated high-density IoT data flow”, *Advanced Information Systems*, vol. 8, no. 3, pp. 77–84, doi: https://doi.org/10.20998/2522-9052.2024.3.09
15. Li, G., Wang, X., & Zhang, Z. (2019), “SDN-based load balancing scheme for multi-controller deployment”, *IEEE Access, 7*, 39612–39622. doi: https://doi.org/10.1109/ACCESS.2019.2906683
16. Kuchuk, N., Kashkevich, S., Radchenko, V., Andrusenko, Y. and Kuchuk, H. (2024), “Applying edge computing in the execution IoT operative transactions”, *Advanced Information Systems*, vol. 8, no. 4, pp. 49–59, doi: https://doi.org/10.20998/2522-9052.2024.4.07
17. Agarwal, S., Kodialam, M., & Lakshman, T. V. (2016), “Multi-controller based software-defined networking: A survey”, *Computer Networks, 103*, 122–135. doi: https://doi.org/10.1016/j.comnet.2016.04.015

Received (Надійшла) 17.12.2025
Acceptedforpublication (Прийнята до друку) 11.03.2026

ВІДОМОСТІ ПРО АВТОРІВ / ABOUT THE AUTHORS

Руккас Кирило Маркович – доктор технічних наук, доцент, професор кафедри теоретичної та прикладної інформатики, Харківський національний університет імені В. Н. Каразіна, Харків, Україна;
Kyrylo Rukkas – Doctor of Technical Sciences, Associate Professor, Professor of Theoretical and Applied Computer Science Department, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine;
e-mail: rukkas@karazin.ua; ORCID Author ID: https://orcid.org/0000-0002-7614-0793;
Scopus Author ID: https://www.scopus.com/authid/detail.uri?authorId=59248988100.

Морозова Анастасія Геннадіївна – кандидат технічних наук, доцент, завідувач кафедри теоретичної та прикладної інформатики, Харківський національний університет імені В. Н. Каразіна, Харків, Україна;
Anastasiia Morozova – Candidate of Technical Sciences, Associate Professor, Head of Theoretical and Applied Computer Science Department, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine;
e-mail: a.morozova@karazin.ua; ORCID Author ID: https://orcid.org/0000-0003-2143-7992;
Scopus Author ID: https://www.scopus.com/authid/detail.uri?authorId=58904589600.

Зарецька Ірина Тимофіївна – кандидат фізико-математичних наук, доцент, доцент кафедри теоретичної та прикладної інформатики, Харківський національний університет імені В. Н. Каразіна, Харків, Україна;
Iryna Zaretska – Candidate of Physical and Mathematical Sciences, Associate Professor, Associate Professor of Theoretical and Applied Computer Science Department, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
e-mail: zaretskaya@karazin.ua ; ORCID Author ID: https://orcid.org/0000-0001-8747-2737;
Scopus Author ID: https://www.scopus.com/authid/detail.uri?authorId=55557172700.

Андрійченко Євгенія Миколаївна – науковий співробітник кафедри інформаційних технологій та бізнес-інформатики, Університет прикладних наук Кампус 02, Грац, Австрія;
Yevheniia Andriichenko – Scientific Researcher at the Department of Information Technology and Business Informatics, Campus 02 UAS, Graz, Austria;
e-mail: yevheniia.andriichenko@campus02.at; ORCID Author ID: https://orcid.org/0009-0005-1568-7292;
Scopus Author ID: https://www.scopus.com/authid/detail.uri?authorId=59252336200.

Чумаченко Дмитро Ігорович – кандидат технічних наук, доцент, афілійований дослідник, Університет Ватерлоо, Ватерлоо, Канада;
Dmytro Chumachenko – Candidate of Technical Sciences, Associate Professor, Affiliated Researcher, University of Waterloo, Waterloo, Canada;
e-mail: dichumachenko@gmail.com; ORCID Author ID: https://orcid.org/0000-0003-2623-3294;
Scopus Author ID: https://www.scopus.com/authid/detail.uri?authorId=58194260300.

Розробка багатоагентної системи динамічного керування SDN мережами

К. М. Руккас, А. Г. Морозова, І. Т. Зарецька, Є. М. Андрійченко, Д. І. Чумаченко

Анотація. Актуальність. На сьогоднішній день зростають обсяги інформації, що передається, та вимоги щодо якості її передачі. Останнім часом набирає популярності технологія SDN, проте виникають труднощі, пов'язані з невизначеністю стану мережі SDN та її елементів, а також інтеграцією різних потоків, які висувають різні вимоги щодо якості доставки інформації. Тому використання інтелектуальної багатоагентної системи (МАС) для управління мережами SDN є актуальною задачею. **Об'єктом дослідження** є процес управління SDN мережами. **Предметом дослідження** є моделі та методи управління SDN мережами. **Мета цієї роботи** – розробка моделі взаємодії інтелектуальних агентів для забезпечення ефективного функціонування МАС динамічного управління SDN. **Результати дослідження**. Розроблено математичні моделі з використанням аналітичного апарату імовірнісно-часових графів двох варіантів координування агентів при динамічному управлінні SDN мережами: системи з координуючим агентом і МАС, що саморегулюється. На основі аналізу отриманих імовірнісно-часових характеристик різних варіантів координування встановлено, що саморегулюючі МАС доцільно використовувати в ситуаціях, коли у агентів достатньо знань для вирішення переважної більшості завдань, що ці рішення з високою ймовірністю виявляються правильними, невелика кількість агентів в системі, а також висока ймовірність здійснення правильного контролю прийнятих рішень.

Ключові слова: SDN; мережа; управління комп'ютерними мережами; багатоагентні системи; імовірнісно-часові графи.