Information systems. Computer sciences. Issues of information security
Информационные системы. Информатика. Проблемы информационной безопасности

UDC 004.89
https://doi.org/10.32362/2500-316X-2023-11-4-16-25

RESEARCH ARTICLE

Approach to knowledge management and the development of a multi-agent knowledge representation and processing system

Evgeniy I. Zaytsev [@], **Elena V. Nurmatova**

MIREA – Russian Technological University, Moscow, 119454 Russia
[@] *Corresponding author, e-mail: zajcev@mirea.ru*

Abstract

Objectives. A multi-agent knowledge representation and processing system (MKRPS) comprises a distributed artificial intelligence system designed to solve problems that are difficult or impossible to solve using monolithic systems. Solving complex problems in an MKRPS is accomplished by communities of intelligent software agents that use cognitive data structures, logical inference, and machine learning. Intelligent software agents are able to act rationally under conditions of incompleteness and ambiguity of incoming information. The aim of the present work is to identify models and methods, as well as software modules and tools, for use in developing a highly efficient MKRPS.

Methods. Agent-based modeling methods were used to formally describe and programmatically simulate the rational behavior of intelligent agents, expert evaluation methods, the mathematical apparatus of automata theory, Markov chains, fuzzy logic, neural networks, and reinforcement learning.

Results. An MKRPS structure diagram, a multi-agent solver, and microservices access control diagram were developed. Methods for distribution of intelligent software agents on the MKRPS nodes are proposed along with algorithms for optimizing the logical structure of the distributed knowledge base (DKB) to improve the performance of the MKRPS in terms of volume, cost and time criteria.

Conclusions. The proposed approach to the development and use of intelligent software agents combines knowledge-based reasoning mechanisms with neural network models. The developed MKRPS structure and DKB control diagram includes described methods for optimizing the DKB, determining the availability of microservices used by the agents, ensuring the reliability assurance and coordinated functioning of the computing nodes of the system, as well as instrumental software tools to simplify the design and implementation of the MKRPS. The results demonstrate the effectiveness of the presented approach to knowledge management and the development of a high-performance problem-oriented MKRPS.

Keywords: multi-agent system, intelligent software agents, multi-agent intelligent solver, knowledge representation and processing system, reinforcement learning

• **Submitted:** 24.10.2022 • **Revised:** 27.01.2023 • **Accepted:** 02.05.2023

For citation: Zaytsev E.I., Nurmatova E.V. Approach to knowledge management and the development of a multi-agent knowledge representation and processing system. *Russ. Technol. J.* 2023;11(4):16–25. https://doi.org/10.32362/2500-316X-2023-11-4-16-25

Financial disclosure: The authors have no a financial or property interest in any material or method mentioned.

The authors declare no conflicts of interest.

НАУЧНАЯ СТАТЬЯ

О подходе к управлению знаниями и разработке мультиагентной системы представления и обработки знаний

Е.И. Зайцев [@], **Е.В. Нурматова**

МИРЭА – Российский технологический университет, Москва, 119454 Россия
[@] *Автор для переписки, e-mail: zajcev@mirea.ru*

Резюме

Цели. Мультиагентная система представления и обработки знаний (МСПОЗ) – это распределенная система искусственного интеллекта, предназначенная для решения проблем, которые трудно или невозможно решить с помощью монолитной интеллектуальной системы. Решение сложных проблем в МСПОЗ осуществляется интеллектуальными программными агентами, которые инкапсулируют в программных классах когнитивные структуры данных, методы логического вывода и машинного обучения. Интеллектуальные программные агенты МСПОЗ способны рационально действовать в условиях неполноты и нечеткости поступающей информации. Целями работы являются исследование и разработка моделей, методов, программных модулей и инструментальных программных средств, которые позволяют создать высокоэффективную МСПОЗ.

Методы. В работе использовались методы агентного моделирования, позволяющие формально описывать и программно имитировать рациональное поведение интеллектуальных агентов, методы экспертных оценок, математический аппарат теории автоматов, марковские цепи, нечеткая логика, нейронные сети, алгоритмы машинного обучения с подкреплением.

Результаты. Разработаны структурная схема МСПОЗ, мультиагентный решатель, схема управления доступом к микросервисам. Предложены методы распределения интеллектуальных программных агентов по узлам МСПОЗ, а также алгоритмы оптимизации логической структуры распределенной базы знаний (РБЗ), позволяющие повысить эффективность объемных, стоимостных и временны́х характеристик МСПОЗ.

Выводы. Предложен подход к разработке и использованию интеллектуальных программных агентов, который объединяет механизмы рассуждений на основе знаний с нейросетевыми моделями. Разработаны структура МСПОЗ, схема управления РБЗ, методы оптимизации РБЗ, определения доступности используемых агентами микросервисов, обеспечения надежности и скоординированного функционирования вычислительных узлов системы, а также инструментальные программные средства, позволяющие упростить процесс проектирования и реализации МСПОЗ. Полученные результаты демонстрируют эффективность представленного подхода к управлению знаниями и разработке высокопроизводительной проблемно-ориентированной МСПОЗ.

Ключевые слова: мультиагентная система, интеллектуальные программные агенты, мультиагентный интеллектуальный решатель, система представления и обработки знаний, обучение с подкреплением

• **Поступила:** 24.10.2022 • **Доработана:** 27.01.2023 • **Принята к опубликованию:** 02.05.2023

Для цитирования: Зайцев Е.И., Нурматова Е.В. О подходе к управлению знаниями и разработке мультиагентной системы представления и обработки знаний. *Russ. Technol. J.* 2023;11(4):16–25. https://doi.org/10.32362/2500-316X-2023-11-4-16-25

Прозрачность финансовой деятельности: Авторы не имеют финансовой заинтересованности в представленных материалах или методах.

Авторы заявляют об отсутствии конфликта интересов.

INTRODUCTION

In a multi-agent knowledge representation and processing system (MKRPS), complex, ill-defined problems are solved by intelligent software agents, which are able to act rationally under conditions of uncertainty, including the incompleteness and ambiguity of incoming information [1–5]. Making decisions and carrying out rational actions, software agents use a knowledge base and event-driven microservices, which are designed as separate interacting processes with lightweight inter-process communications. Agents interact with microservices through events, which can be simple notifications or complex state-supported structures.

The solution of complex problems in the MKRPS is performed by decomposing the problems into subtasks, which are jointly solved by reactive and cognitive application software agents. Both horizontal decomposition, which creates a multi-connected system having a flat structure, and vertical decomposition, which creates a hierarchical system with several levels, are used.

Due to the implementation of Reinforcement Learning (RL) machine learning methods in the MKRPS, the behavior of applied software agents becomes more rational when solving problems repeatedly. The Actor-Critic algorithm [6–10] is used to train the applied software agents in the MKRPS.

In order to improve performance, special application programming interfaces (API) and system software modules associated with system software agents are implemented in the MKRPS. System software agents plan and manage the computational resources of the MKRPS, as well as providing mobility for the application software agents. Application agents can roam nodes of the MKRPS that provide the necessary environment for them. In contrast to containers implemented on the basis of namespaces (e.g., by the Docker platform[1]), specialized LibOS (Library Operating System) modules are used to support the technology of mobile agents in the MKRPS.

The performance of an MKRPS is largely determined by the selected approach for structuring, storing and processing knowledge [11, 12]. A high-performance problem-oriented multi-agent solver has been developed, the logical structure of whose DKB has been optimized to support minimal total processing time of queries and transactions.

MKRPS STRUCTURE

The structural diagram of the MKRPS is shown in Fig. 1. There are two types of applied (intelligent) software agents at each computational node of the MKRPS: reactive and cognitive [13]. Application software agents interact with each other, as well as with system software agents that are part of the external shared user-level library LibOS, which is oriented on the exokernel architecture of the OS. System software agents are used for planning and managing computing resources, as well as load balancing and system monitoring.

Special software methods and Cognitive Data Structures (CDS) associated with cognitive software agents are used to represent agent-based abstractions (goals, desires, intentions, beliefs of agents) and implement logical inferences.

Figure 2 shows an example of a state-transition diagram of a cognitive software agent comprising one of the MKRPS nodes.

As follows from the diagram, a cognitive software agent can be in one of five states, two of which are composite, i.e., they have nested states. A change of the state of a software agent occurs as a result of an event. It is possible to switch to a new state without committing an event, which is carried out immediately after performing actions (or activities) associated with the previous state.

From the “Initialization” state, the cognitive agent switches to the “Choice” composite state. In this state, a strategy is generated and the necessary knowledge source is selected, taking into account the links established during the initialization stage and the informative messages received from other software agents. Then, the cognitive software agent enters the “Coordination” composite state, in which new knowledge sources are activated and the actions of reactive software agents are coordinated.

If the reactive software agents do not find a consistent solution (the “No Solution” state), the cognitive software agent returns to the nested partial solution state of the problem. If a solution is found (the “Solution” state), the data obtained at this stage are used to form new queries to the knowledge base.

To work with the knowledge base, four types of methods are implemented to form queries and process the results of these queries:

- analysis (ANS method) for implementing a logical analysis of events;
- association (ASS method) used to get responses to queries about links between objects and events;
- comparison of events or objects (CMP method);
- object specification (VAL method).

Both explicit and fuzzy queries to the knowledge base can be used to specify objects. When implementing fuzzy queries, different types of affiliation functions can be used; these are chosen by the knowledge engineer on the basis of the results of computational experiments.

Cognitive software agents coordinate the work of reactive software agents associated with local knowledge sources. An example of an interaction diagram of reactive software agents of one of the MKRPS nodes is shown in Fig. 3.

[1] https://www.docker.com/. Accessed March 20, 2023.

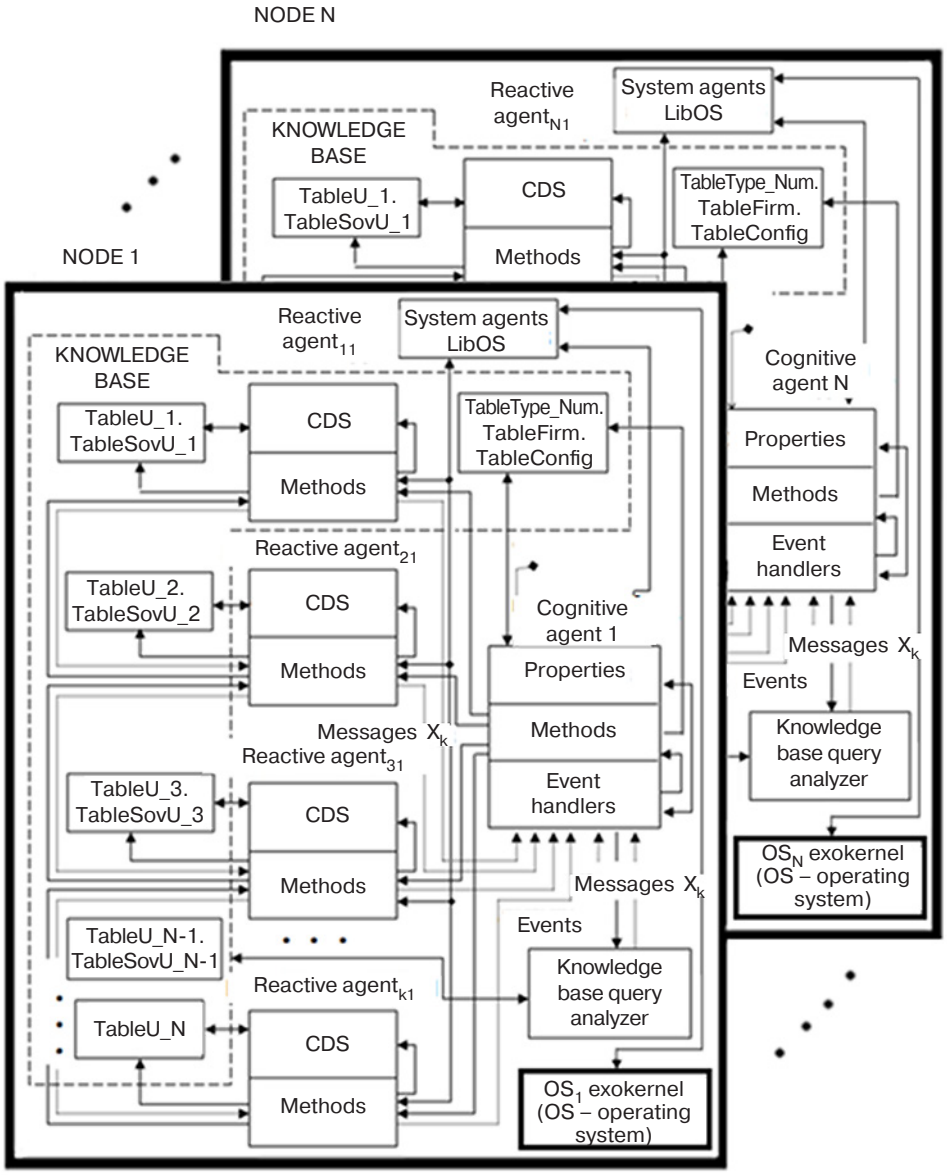

Fig. 1. MKRPS structure

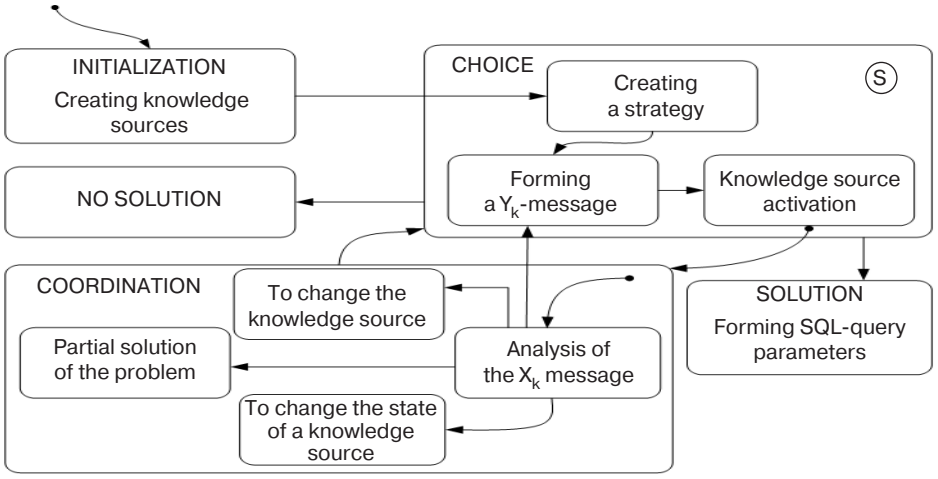

Fig. 2. State-transition diagram of a cognitive software

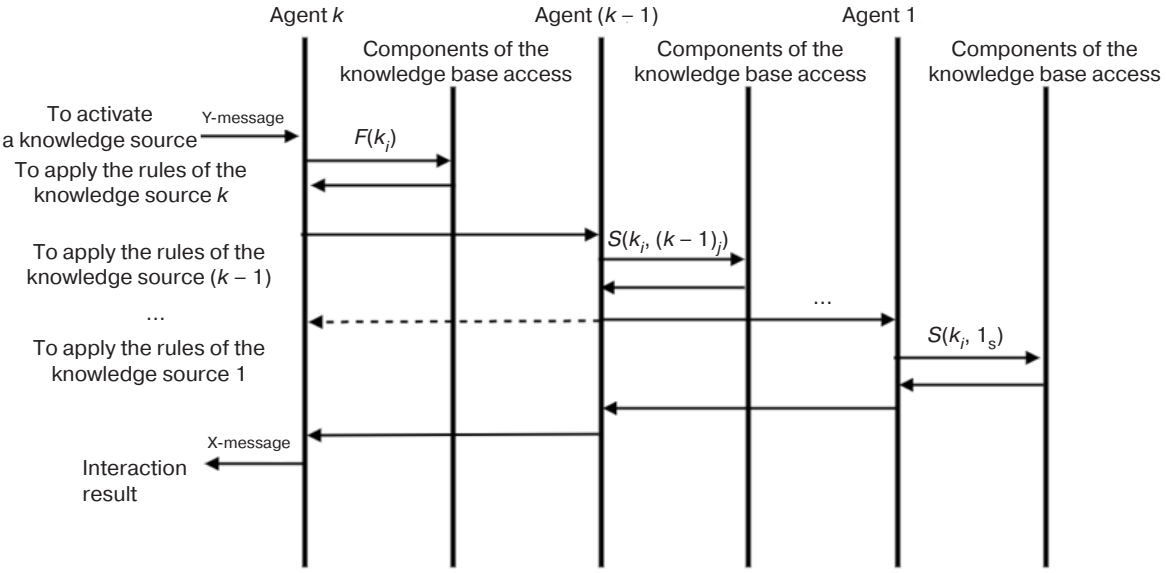

Fig. 3. Diagram of reactive software agent interaction

In the interaction diagram, time moves from top to bottom. Knowledge base access software components and agents are represented by vertical lines. Messages between agents (components) are marked with horizontal arrows. Upon receipt of a message, each agent (source of knowledge) invokes the corresponding program method (a member function of class $F()$ or $S()$) and returns the result. The figure on the left shows the comments. In this case, each reactive software agent with number k interacts only with its nearest neighbor having the number $(k - 1)$. The software agents in this group sequentially perform their assigned tasks within a single process (without switching the context). The priorities of the reactive software agents comprising the associated MKRPS node are set according to the agent's sequence number. The first reactive agent uses high-priority frames associated with TableU_1 and TableSovU_1. The software agent with number k has the lowest priority and is associated with TableU_N.

If the MKRPS node is a multiprocessor system, the agents of this node can act simultaneously. To organize parallel computations, the application software agents of each MKRPS node are distributed into groups using compatibility and inclusion matrices.

The compatibility matrix **S** has the following form:

$$\mathbf{S} = \begin{bmatrix} 0 & s_{12} & s_{13} & \dots & s_{1M} \\ s_{21} & 0 & & \dots & s_{2M} \\ s_{31} & s_{32} & 0 & \dots & s_{3M} \\ \dots & \dots & \dots & \dots & \dots \\ s_{M1} & s_{M2} & s_{M3} & \dots & 0 \end{bmatrix} \begin{matrix} S_1 \\ S_2 \\ S_3 \\ \dots \\ S_M \end{matrix},$$

where $s_{ij} = 1$ if agents A_i and A_j should work in parallel, otherwise $s_{ij} = 0$; S_i is the ith row of the coherence matrix **S**; M is the number of agents.

The inclusion matrix **R** is used to distribute the software agents of the node into groups:

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1M} \\ r_{21} & r_{22} & \dots & r_{2M} \\ \dots & \dots & \dots & \dots \\ r_{N1} & r_{N2} & \dots & r_{NM} \end{bmatrix} \begin{matrix} R_1 \\ R_2 \\ \dots \\ R_N \end{matrix},$$

where N is the number of groups; $r_{ij} = 1$ if agent A_i is included in group Y_j. Agent A_i is included in group Y_j if $S_i \cap R_j = \varnothing$, i.e., matrix rows do not intersect.

For optimal partitioning of the set of agents into subsets when using compatibility and inclusion matrices, it is necessary to consider the structure of the MKRPS node, the functional features of software agents, and their requirements to computing resources.

In Multi-Agent Reinforcement Learning (MARL), the environment depends on all software agents. Unlike centralized learning, in which software agents have full control over the computational process (the relevant policies being distributed by a central agent), in the decentralized model used in MKRPS, independent agents can share experiences and policies. In the decentralized model, execution and learning are implemented locally, allowing the application software agents to adapt to the local perception of the environment (Fig. 4).

A decentralized model of multi-agent reinforcement learning allows the use of standard RL algorithms. Applied software agents of MKRPS are trained through a series of rewards and punishments based on the Actor-Critic algorithm, in which a strategy generates actions, while a value function critiques those actions.

Since there is an actor and a critic for each software agent, agents may have different strategies (policies) and

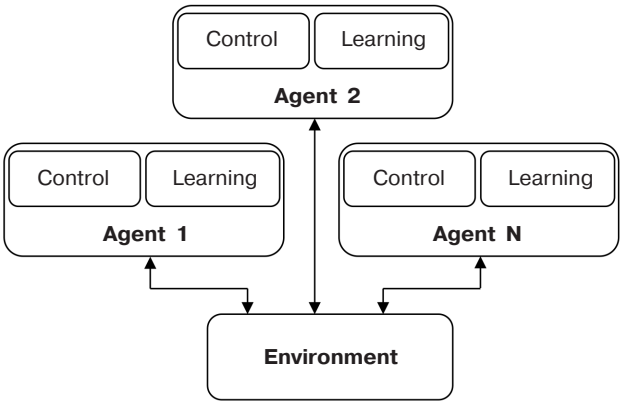

Fig. 4. Decentralized model of multi-agent reinforcement learning

rewards. The agents of MKRPS cooperate to optimize the overall long-term goal. Function approximation in MKRPS is implemented based on neural networks that model both policy and value functions.

For effective implementation of the multi-agent reinforcement learning, problems of multidimensional and multimodal targets, scalability, instability, and optimality must be solved [7].

In the process of solving subtasks, software agents use microservices, which are duplicated on different nodes of the MKRPS to improve reliability and performance. The system software agents distribute the computational load and manage the microservices based on the data provided by the monitor agents (Fig. 5).

SYNTHESIS OF DISTRIBUTED COGNITIVE DATA STRUCTURES

Due to the large dimensionality of the created cognitive data structures, operations with data structures in MKRPS, whether representing data replenishment or retrieval, are performed by keys. To do this, the entire logical CDS structure must be broken down into a number of clusters that have the smallest interconnection under various constraints. In this case, we will accept restrictions on the dimensionality of clusters and restrictions on the degree of semantic proximity of logical records included in the clusters, taking into account the type of storage systems used.

Let us introduce a binary parameter Z_{kj}^{i} to characterize the use by the *k*th query of the *i*th data group

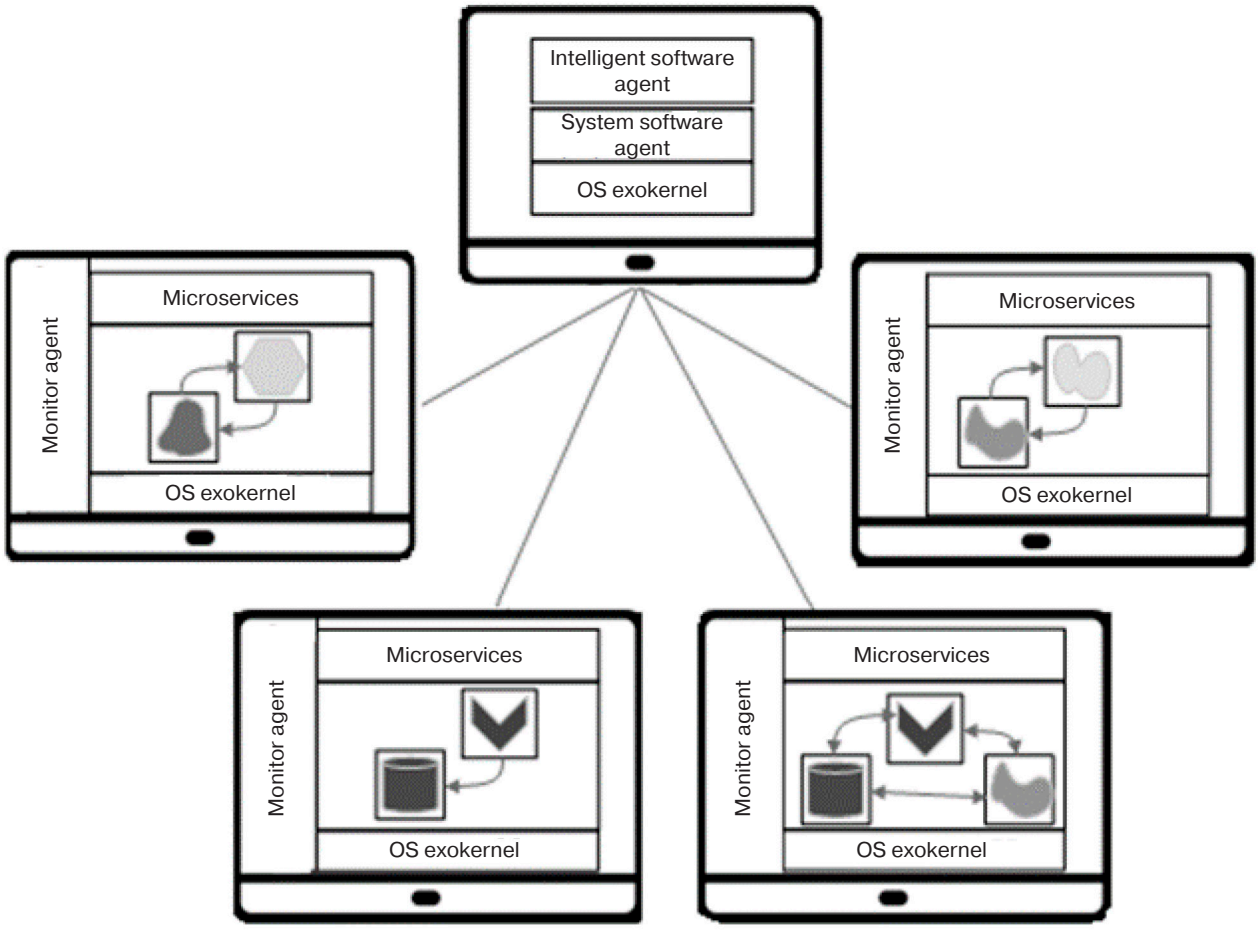

Fig. 5. Microservices and system monitor agents

related to the *j*th logical record. The calculation of this parameter is based on the binary variable a_{ik}, which is equal to one if the *i*th data group is included in the *k*th query, and zero if it is not. This parameter, which is given by the link logic that data groups are included in logical records, is amplified by the variable x_{ij}, which, by analogy, is equal to one if the *i*th data group is included in the *j*th logical record; $x_{ij} = 0$ if it is not.

We have the following calculation of the Z_{kj}^{i} parameter:

$$Z_{kj} = \begin{cases} 1, \text{ at } \sum_{i=1}^{I} a_{ik} x_{ij} \geq 1, \\ 0, \text{ at } \sum_{i=1}^{I} a_{ik} x_{ij} = 0. \end{cases}$$

The complete problem of synthesis of the distributed data structure for MKRPS will be solved by taking into account the criterion of the minimum total time of execution of user requests under such constraints as the uniqueness of the data sets in the record, the length of the logical record, the total number of types of logical records in the structure, the data search time structured by duration of requests, as well as the uniqueness of the input nodes in the structure and their total number.

In the present work, we give an approximate algorithm for solving the problem of synthesis of the optimal distributed data structure by the criterion of minimum total query execution time. Thus, in order to determine the distribution of groups by the criterion of the minimum total traffic, we first use an approximate algorithm for the distribution of data clusters between the server and clients of the local network, then reduce the canonical graph of the data structure to an uncoupled graph with the calculation of the weight of each data group.

The data group weight includes the weights of the group itself, as well as the weights of the arcs, taking into account the requirements of MKRPS users:

$$V_i = V_i^{\text{gr}} + V_{ii'}^{\text{cg}},$$

where V_i^{gr} is the total weight of the data group; $V_{ii'}^{\text{cg}}$ is the weight of the arcs of the connected graph of the canonical data structure; *i'* is the index of the group adjacent to the *i*th data group.

$$V_i^{\text{gr}} = \sum_{k=1}^{k_0} \sum_{p=1}^{p_0} \gamma_{kp}^{\text{q}} \delta_{kp}^{\text{q}} \vartheta_{pi},$$

$$V_{ii'}^{\text{cg}} = \sum_{k=1}^{k_0} \sum_{p=1}^{p_0} \gamma_{kp}^{\text{q}} \delta_{kp}^{\text{q}} \vartheta_{pi} \sum_{i' \neq i}^{I} \vartheta_{pi'} a_{ii'}^{\text{G}},$$

where γ_{kp}^{q} is the frequency of query usage by user; δ_{kp}^{q} are elements of user query matrix; $\vartheta_{pi'}$ is the matrix of data groups in query processing; $a_{ii'}^{\text{G}}$ is the matrix of semantic adjacency of the *i*th data group with the data group that has index *i'*.

For a particular *i*th group, the weight will be:

$$V_i = \sum_{k=1}^{k_0} \sum_{p=1}^{p_0} \gamma_{kp}^{\text{q}} \delta_{kp}^{\text{q}} \vartheta_{pi} \left(1 + \sum_{i' \neq i}^{I} \vartheta_{pi'} a_{ii'}^{\text{G}} \right).$$

Then the graph of the computer network is converted into an unconnected graph with calculation of node weight:

$$V_r = t_r + \sum_{r' \neq r}^{R_0} t_{rr'},$$

where t_r is the total average duration of data processing in the *r*th node, consisting of the time of decomposition of the request into subrequests, route selection, and connection establishment; $t_{rr'}$ is the average duration of data transmission between nodes, determined on the basis of the matrix of logical distances between servers of the computer network nodes.

Then a matrix $\mathbf{V} = \|v_{ir}\|$ is formed, the elements of which are defined as the Cartesian product of the weight of each node by the weight of each data group:

$$v_{ir} = V_i \times V_i \text{ for } i = \overline{1, I}, \; r = \overline{1, R_0}.$$

Next, we solve the problem

$$\min_{\{x_{ir}\}} \sum_{i=1}^{I} \sum_{r=1}^{R_0} v_{ir} x_{ir}$$

with the following restrictions:

- by the number of data groups that can be localized on one node:

$$\sum_{i=1}^{I} x_{ir} \leq N_r, \; r = \overline{1, r_0};$$

- on the permissible redundancy of groups by network nodes:

$$\sum_{r=1}^{r_0} x_{ir} \leq M_i, \; i = \overline{1, I};$$

- the capacity of the available external memory of the data storage system:

$$\sum_{i=1}^{I} x_{ir}\rho_i\pi_i \le \eta_r^{\text{DSD}},$$

where ρ_i are values of data group lengths; π_i is the amount of instances in groups; η_r^{DSD} is the amount of available external data storage device (DSD) memory on the node; $x_{ir} = 1$ if the ith data group is included in the rth network node, $x_{ir} = 0$ if it is not.

The solution of the linear integer programming problem is used not only to determine the optimal localization of data groups by network nodes, but also to define the optimal structure of data groups placed on the network nodes.

At the next stage we solve the problem of optimal distribution of node data groups in each node of the network by types of logical records according to the criterion of minimum total time of local data processing. Here, the number of synthesis tasks is determined by the number of network nodes. The initial data are comprised of the subgraphs of the canonical data structure graph, as well as the temporal and volumetric characteristics of the subgraphs of their canonical structure, a set of user requests, and network nodes [14, 15]. The synthesis problem is solved by approximate algorithms with restrictions on the number of groups in a record, on the uniqueness of the inclusion of groups in a record, on the cost of storing information, and on the total time of the request service. As a result, the logical structures of the database are determined for each node of the network along with the formed distribution matrices of the set of data groups by types of logical records and subsequent distribution matrices of the set of record groups by network nodes.

The results of solving the problem of synthesis of distributed cognitive data structures have important practical application in the design of the optimal DKB structure and the possibility of forming specifications for queries and adjustments of distributed data.

CONCLUSIONS

The paper presents a methodological approach to the development of MKRPS. The functional and structural organization of the multi-agent solver are described in terms of models of applied and system software agents, as well as methods of designing and managing DKB. Methods for distributing software agents to the nodes of MKRPS, determining the availability of microservices, as well as providing reliable and coordinated work of computing nodes, are considered. Examples of interaction-, state- and transition diagrams for cognitive and reactive software agents are given. Algorithms for the optimization of logical DKB structure are described. The optimization of DKB improves the efficiency of MKRPS in terms of time-, volume- and cost characteristics to make it more productive, flexible and functional. The obtained results confirm the effectiveness of the presented approach to the development of MKRPS.

Authors' contribution. All authors equally contributed to the research work.

REFERENCES

1. Zaytsev E.I., Khalabiya R.F., Stepanova I.V., Bunina L.V. Multi-Agent System of Knowledge Representation and Processing. In: Kovalev S., Tarassov V., Snasel V., Sukhanov A. (Eds.). *Proceedings of the Fourth International Scientific Conference "Intelligent Information Technologies for Industry"* (*IITI'19*). *Advances in Intelligent Systems and Computing*. Springer; 2020. V. I. P. 131–141. https://doi.org/10.1007/978-3-030-50097-9_14
2. Baranauskas R., Janaviciute A., Jasinevicius R., Jukavicius V. On Multi-Agent Systems Intellectics. *Inf. Technol. Control*. 2015;44(1):112–121. https://doi.org/10.5755/j01.itc.44.1.8768
3. Darweesh S., Shehata H. Performance Evaluation of a Multi-Agent System using Fuzzy Model. *2018 First International Workshop on Deep and Representation Learning* (*IWDRL*). 2018. P. 7–12. https://doi.org/10.1109/IWDRL.2018.8358208

СПИСОК ЛИТЕРАТУРЫ

1. Zaytsev E.I., Khalabiya R.F., Stepanova I.V., Bunina L.V. Multi-Agent System of Knowledge Representation and Processing. In: Kovalev S., Tarassov V., Snasel V., Sukhanov A. (Eds.). *Proceedings of the Fourth International Scientific Conference "Intelligent Information Technologies for Industry"* (*IITI'19*). *Advances in Intelligent Systems and Computing*. Springer; 2020. V. I. P. 131–141. https://doi.org/10.1007/978-3-030-50097-9_14
2. Baranauskas R., Janaviciute A., Jasinevicius R., Jukavicius V. On Multi-Agent Systems Intellectics. *Inf. Technol. Control*. 2015;44(1):112–121. https://doi.org/10.5755/j01.itc.44.1.8768
3. Darweesh S., Shehata H. Performance Evaluation of a Multi-Agent System using Fuzzy Model. *2018 First International Workshop on Deep and Representation Learning* (*IWDRL*). 2018. P. 7–12. https://doi.org/10.1109/IWDRL.2018.8358208

4. Russel S., Norvig P. *Iskusstvennyi intellekt: sovremennyi podkhod.* T. 2. *Znaniya i rassuzhdeniya v usloviyakh neopredelennosti* (*Artificial Intelligence: A Modern Approach.* V. 2. *Knowledge and Reasoning under Uncertainty*): transl. from Engl. St. Petersburg: Dialektika; 2021. 480 p. (in Russ.).
5. Russel S., Norvig P. *Iskusstvennyi intellekt: sovremennyi podkhod.* T. 3. *Obuchenie, vospriyatie i deistvie* (*Artificial Intelligence: A Modern Approach.* V. 3. *Learning, Perception, and Action*): transl. from Engl. St. Petersburg: Dialektika; 2022. 640 p. (in Russ.).
6. Sutton R.S., Barto E.J. *Obuchenie s podkrepleniem* (*Reinforcement Learning*): transl. from Engl. Moscow: DMK Press; 2020. 552 p. (in Russ.).
7. Graesser L., Keng W.L. *Glubokoe obuchenie s podkrepleniem: teoriya i praktika na yazyke Python* (*Reinforcement Learning: Theory and Practice in Python*): transl. from Engl. St. Petersburg: Piter; 2022. 416 p. (in Russ.).
[Graesser L., Keng W.L. *Foundations of Deep Reinforcement Learning: Theory and Practice in Python*. Addison-Wesley Professional; 2020. 416 p.]
8. Pumperla M., Ferguson K. *Glubokoe obuchenie i igra v go* (*Deep Learning and the Game of Go*): transl. from Engl. Moscow: DMK Press; 2020. 372 p. (in Russ.).
9. Red'ko V.G. *Evolyutsiya, neironnye seti, intellekt: Modeli i kontseptsii evolyutsionnoi kibernetiki* (*Evolution, Neural Networks, Intelligence: Models and Concepts of Evolutionary Cybernetics*). Moscow: LIBROKOM; 2013. 224 p. (in Russ.).
10. Winder P. *Reinforcement Learning. Industrial Applications of Intelligent Agents*. O'Reilly Media, Inc.; 2021. 382 p.
11. Houhamdi Z., Athamena B., Abuzaineddin R., Muhairat M. A Multi-Agent System for Course Timetable Generation. *TEM Journal*. 2019;8(1):211–221. https://doi.org/10.18421/TEM81-30
12. Aly S., Badoor H. Performance Evaluation of a Multi-Agent System using Fuzzy Model. *2018 First International Workshop on Deep and Representation Learning* (*IWDRL*). 2018. P. 175–189. https://doi.org/10.1109/IWDRL.2018.8358208
13. Zaytsev E.I. Method of date representation and processing in the distributed intelligence information systems. *Avtomatizatsiya i sovremennye tekhnologii = Automation. Modern Technologies*. 2008;1:29–34 (in Russ.).
14. Batouma N., Sourrouille J.-L. Dynamic adaption of resource aware distributed applications. *Int. J. Grid Distrib. Comput*. 2011;4(2):25–42. Available from URL: http://article.nadiapub.com/IJGDC/vol4_no2/3.pdf
15. Nurmatova E.V., Gusev V.V., Kotliar V.V. Analysis of the features of the optimal logical structure of distributed databases. In: *GRID Workshop Proceedings - GRID 2018 Selected Papers of the 8th International Conference "Distributed Computing and Grid-technologies in Science and Education"*. Dubna; 2018. V. 2267. P. 579–584. Available from URL: https://ceur-ws.org/Vol-2267/579-584-paper-111.pdf

4. Рассел С., Норвиг П. *Искусственный интеллект: современный подход*. Т. 2. *Знания и рассуждения в условиях неопределенности*: пер. с англ. СПб.: Диалектика; 2021. 480 с.
5. Рассел С., Норвиг П. *Искусственный интеллект: современный подход*. Т. 3. *Обучение, восприятие и действие*: пер. с англ. СПб.: Диалектика; 2022. 640 с.
6. Саттон Р.С., Барто Э.Дж. *Обучение с подкреплением*: пер. с англ. М.: ДМК Пресс; 2020. 552 с.
7. Грессер Л., Кенг В.Л. *Глубокое обучение с подкреплением: теория и практика на языке Python:* пер. с англ. СПб.: Питер; 2022. 416 с.
[Graesser L., Keng W.L. *Foundations of Deep Reinforcement Learning: Theory and Practice in Python*. Addison-Wesley Professional; 2020. 416 p.]
8. Памперла М. Фергюсон К. *Глубокое обучение и игра в го:* пер. с англ. М.: ДМК Пресс; 2020. 372 с.
[Pumperla M., Ferguson K. *Deep Learning and the Game of Go.* Manning; 2019. ISBN 978-1-6172-9532-4. 384 p.]
9. Редько В.Г. *Эволюция, нейронные сети, интеллект: Модели и концепции эволюционной кибернетики*. М.: ЛИБРОКОМ; 2013. 224 с.
10. Winder P. *Reinforcement Learning. Industrial Applications of Intelligent Agents*. O'Reilly Media, Inc.; 2021. 382 p.
11. Houhamdi Z., Athamena B., Abuzaineddin R., Muhairat M. A Multi-Agent System for Course Timetable Generation. *TEM Journal*. 2019;8(1):211–221. https://doi.org/10.18421/TEM81-30
12. Aly S., Badoor H. Performance Evaluation of a Multi-Agent System using Fuzzy Model. *2018 First International Workshop on Deep and Representation Learning* (*IWDRL*). 2018. P. 175–189. https://doi.org/10.1109/IWDRL.2018.8358208
13. Зайцев Е.И. Методология представления и обработки знаний в распределенных интеллектуальных информационных системах. *Автоматизация и современные технологии*. 2008;1:29–34.
14. Batouma N., Sourrouille J.-L. Dynamic adaption of resource aware distributed applications. *Int. J. Grid Distrib. Comput*. 2011;4(2):25–42. URL: http://article.nadiapub.com/IJGDC/vol4_no2/3.pdf
15. Nurmatova E.V., Gusev V.V., Kotliar V.V. Analysis of the features of the optimal logical structure of distributed databases. In: *GRID Workshop Proceedings – GRID 2018 Selected Papers of the 8th International Conference "Distributed Computing and Grid-technologies in Science and Education."* Dubna: 2018. V. 2267. P. 579–584. URL: https://ceur-ws.org/Vol-2267/579-584-paper-111.pdf

About the authors

Evgeniy I. Zaytsev, Cand. Sci. (Eng.), Associate Professor, Department of Hardware Software and Mathematical Support of Computing System, Institute for Cybersecurity and Digital Technologies, MIREA – Russian Technological University (20, Stromynka ul., Moscow, 107996 Russia). E-mail: zajcev@mirea.ru. Scopus Author ID 57218190023, ResearcherID ABA-4823-2020, RSCI SPIN-code 9662-7658, https://orcid.org/0000-0002-1979-5611

Elena V. Nurmatova, Cand. Sci. (Eng.), Associate Professor, Department of Hardware Software and Mathematical Support of Computing System, Institute for Cybersecurity and Digital Technologies, MIREA – Russian Technological University (20, Stromynka ul., Moscow, 107996 Russia). E-mail: nurmatova@mirea.ru. Scopus Author ID 57205460003, ResearcherID GQI-3212-2022, RSCI SPIN-code 7036-3661, https://orcid.org/0000-0001-8511-0978

Об авторах

Зайцев Евгений Игоревич, к.т.н., доцент, кафедра «Аппаратное, программное и математическое обеспечение вычислительных систем» Института кибербезопасности и цифровых технологий ФГБОУ ВО «МИРЭА – Российский технологический университет» (107996, Россия, Москва, ул. Стромынка, д. 20). E-mail: zajcev@mirea.ru. Scopus Author ID 57218190023, ResearcherID ABA-4823-2020, SPIN-код РИНЦ 9662-7658, https://orcid.org/0000-0002-1979-5611

Нурматова Елена Вячеславовна, к.т.н., доцент, кафедра «Аппаратное, программное и математическое обеспечение вычислительных систем» Института кибербезопасности и цифровых технологий ФГБОУ ВО «МИРЭА – Российский технологический университет» (107996, Россия, Москва, ул. Стромынка, д. 20). E-mail: nurmatova@mirea.ru. Scopus Author ID 57205460003, ResearcherID GQI-3212-2022, SPIN-код РИНЦ 7036-3661, https://orcid.org/0000-0001-8511-0978

Translated from Russian into English by Lyudmila O. Bychkova
Edited for English language and spelling by Thomas A. Beavitt