ТРУДЫ

ИНСТИТУТА СИСТЕМНОГО ПРОГРАММИРОВАНИЯ РАН

PROCEEDINGS OF THE INSTITUTE FOR SYSTEM PROGRAMMING OF THE RAS

ISSN Print 2079-8156
Том 36 Выпуск 2

ISSN Online 2220-6426
Volume 36 Issue 2

Институт системного
программирования
им. В.П. Иванникова РАН

Москва, 2024

ИСП РАН

ISSN 2079-8156 (Print), ISSN 2220-6426 (Online)

Труды Института системного программирования РАН
Proceedings of the Institute for System Programming of the RAS

Труды ИСП РАН – это издание с двойной анонимной системой рецензирования, публикующее научные статьи, относящиеся ко всем областям системного программирования, технологий программирования и вычислительной техники. Целью издания является формирование научно-информационной среды в этих областях путем публикации высококачественных статей в открытом доступе.
Издание предназначено для исследователей, студентов и аспирантов, а также практиков. Оно охватывает широкий спектр тем, включая, в частности, следующие:

- операционные системы;
- компиляторные технологии;
- базы данных и информационные системы;
- параллельные и распределенные системы;
- автоматизированная разработка программ;
- верификация, валидация и тестирование;
- статический и динамический анализ;
- защита и обеспечение безопасности ПО;
- компьютерные алгоритмы;
- искусственный интеллект.

Журнал издается по одному тому в год, шесть выпусков в каждом томе.
Поддерживается открытый доступ к содержанию издания, обеспечивая доступность результатов исследований для общественности и поддерживая глобальный обмен знаниями.
Труды ИСП РАН реферируются и/или индексируются в:

Proceedings of ISP RAS are a double-blind peer-reviewed journal publishing scientific articles in the areas of system programming, software engineering, and computer science. The journal's goal is to develop a respected network of knowledge in the mentioned above areas by publishing high quality articles on open access.
The journal is intended for researchers, students, and practitioners. It covers a wide variety of topics including (but not limited to):

- Operating Systems.
- Compiler Technology.
- Databases and Information Systems.
- Parallel and Distributed Systems.
- Software Engineering.
- Software Modeling and Design Tools.
- Verification, Validation, and Testing.
- Static and Dynamic Analysis.
- Software Safety and Security.
- Computer Algorithms.
- Artificial Intelligence.

The journal is published one volume per year, six issues in each volume.
Open access to the journal content allows to provide public access to the research results and to support global exchange of knowledge. **Proceedings of ISP RAS** is abstracted and/or indexed in:

УДК004.45

Адрес: 109004, г. Москва, ул. А. Солженицына, дом 25.
Телефон: +7(495) 912-44-25
E-mail: info-isp@ispras.ru
Сайт: http://www.ispras.ru/proceedings/

Address: 25, Alexander Solzhenitsyn st., Moscow, 109004, Russia.
Tel: +7(495) 912-44-25
E-mail: info-isp@ispras.ru
Web: http://www.ispras.ru/en/proceedings

Содержание

Table of Contents

DOI: 10.15514/ISPRAS-2024-36(2)-1

Applicances of Different Kind of Storage Systems for Network Traffic Analysis Results

[1] *V.I. Egorov,* ORCID: 0009-0001-5168-6474 <*unclehook@ispras.ru*>
[1] *R.E. Ponomarenko,* ORCID: 0009-0009-5741-3627 <*rerandom@ispras.ru*>
[1,2,3,4] *A.I. Getman,* ORCID: 0000-0002-6562-9008 <*ever@ispras.ru*>

[1] *Ivannikov Institute for System Programming of the RAS,*
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia.
[2] *Moscow Institute of Physics and Technology,*
9 Institutskiy per., Dolgoprudny, Moscow Region, 141701, Russia.
[3] *National Research University "Higher School of Economics",*
20 Myasnitskaya ulitsa, Moscow, 101000, Russia.
[4] *Lomonosov Moscow State University,*
1, Leninskie Gory, Moscow, 119991, Russia.

Abstract. Network Traffic Analysis (NTA) helps identify security threats, monitor network performance, and plan for future capacity. While real-time analysis is ideal, it can be difficult due to high data volume and complexity. Large amounts of traffic require parsing, and real-time data may miss hidden threats. Post-analysis can address these challenges. It hardly depends on choosing an effective and appropriate storage solutions. A variety of storage systems exist, each employing different approaches and formats to retain data. This article explores the applications of various storage systems for NTA results. Three different types of storage systems considered, including Greenplum, Nebula graph and OpenSearch. A comparative approach is employed, analyzing the same dataset across various storage systems.This allows to examine how different database structures and query capabilities influence the efficiency and accuracy of NTA. The resulting insights will not only provide valuable guidance for selecting the optimal storage solution for specific NTA tasks, but also serve as a foundation for future research in this area.

Keywords: Network Traffic Analysis, storage systems, database analysis, Greenplum, OpenSearch, Nebula graph.

For citation: Egorov V.I., Ponomarenko R.E., Getman A.I. Applicances of different kind of storage systems for network traffic analysis results. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 7-20. DOI: 10.15514/ISPRAS-2024-36(2)-1.

Применение различных систем хранения для результатов анализа сетевого трафика

[1] *В.И. Егоров,* ORCID: 0009-0001-5168-6474 <*unclehook@ispras.ru*>
[1] *Р.Е. Пономаренко,* ORCID: 0009-0009-5741-3627 <*rerandom@ispras.ru*>
[1,2,3,4] *А.И. Гетьман,* ORCID: 0000-0002-6562-9008 <*ever@ispras.ru*>
[1] *Институт системного программирования им. В.П. Иванникова РАН,*
109004, Россия, г. Москва, ул. А. Солженицына, д. 25.
[2] *Московский физико-технический институт,*
141700, Россия, Московская область, г. Долгопрудный, Институтский пер., 9.
[3] *Национальный исследовательский университет «Высшая школа экономики»,*
101978, Россия, г. Москва, ул. Мясницкая, д. 20.
[4] *Московский государственный университет имени М.В. Ломоносова*
119991, Россия, г. Москва, Ленинские горы, д. 1.

Аннотация. Анализ сетевого трафика (NTA) помогает выявлять угрозы безопасности, наблюдать за производительностью сети и планировать емкость сети. Идеальным подходом в такой задаче выступает анализ в реальном времени, однако, он может быть затруднен из-за большого объема данных и сложности обрабатываемых данных. Большие объемы трафика требуют подробного разбора, а анализ данных в реальном времени может привести к пропуску скрытых угроз в трафике. Офлайн анализ может решить эти проблемы. Данный подход во многом зависит от выбора эффективного и подходящего решения для хранения данных. Существует множество систем хранения данных, каждая из которых использует разные подходы и форматы для хранения данных. Эта статья исследует применение различных систем хранения данных для результатов NTA. Рассматриваются три разных типа систем хранения, включая Greenplum, Nebula graph и OpenSearch. Используется сравнительный подход, анализирующий один и тот же набор данных в разных системах хранения. Это позволяет изучить, как различные структуры баз данных и возможности запросов влияют на эффективность и точность NTA. Полученные результаты не только предоставят ценные рекомендации по выбору оптимального решения для хранения данных для конкретных задач NTA, но также послужат основой для будущих исследований в этой области.

Ключевые слова: анализ сетевого трафика, системы хранения, анализ баз данных, Greenplum, OpenSearch, Nebula graph.

Для цитирования: Егоров В.И., Пономаренко Р.Е., Гетьман А.И. Применение различных систем хранения для результатов анализа сетевого трафика. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 7–20 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–1.

1. Introduction

Network Traffic Analysis (NTA) have many appliances such as security threats identification, network performance monitoring and bottlenecks, capacity planning of network infrastructure and different application monitoring. While real time traffic analysis is very important it might be hard to achieve. Moreover, it could be unachievable is some cases. There are two main reasons why analyzing network traffic in real-time is challenging: Volume of Data and Data Complexity. Network traffic can be immense, especially in large organizations or environments with high bandwidth. Another important thing is that not all network traffic is readily interpretable. Raw data packets need to be parsed and analyzed to understand the context and identify anomalies. In other cases information from real-time data is not enough. For example, threat incidents might be identified only then it already appeared. Post-facto analysis is a viable solution to this problem. Stored network traffic analysis results have a wide range of practical applications, here are some key areas:

- Stored data allows for forensic investigation after a security breach. Analysts can examine historical traffic patterns to identify suspicious activity leading up to the incident.

- Stored traffic data can be used to train machine learning models to recognize malicious patterns. These models can then be used for real-time analysis to proactively block threats.
- Historical traffic data helps predict future network demands. This information is crucial for network administrators to plan for infrastructure upgrades or capacity expansion to ensure smooth network operation.
- By analyzing historical traffic patterns, security teams can build baselines for normal network activity. Deviations from these baselines can indicate potential threats like malware or unauthorized access attempts.

The biggest challenge for NTA systems lies in designing a robust data storage solution that is flexible, scalable, and delivers fast performance. The large volume of data generated necessitates this robust infrastructure to ensure efficient storage and analysis. Choosing the optimal database for this demanding environment presents distinct challenges, including:

- Data Variety: Effectively accommodating the diverse data formats generated by NTA systems within a single database.
- Performance: Ensuring efficient data retrieval and analysis, especially for real-time security monitoring.
- Scalability: Adapting the database to accommodate the ever-growing volume of network traffic data.
- Security: Implementing robust security measures to safeguard sensitive network data stored within the database.

There are various types of storage systems, each with its own strengths and weaknesses. Traditional options include relational database management systems which follows SQL standard. They might be categorized into OnLine Analytical Processing (OLAP) and OnLine Transaction Processing (OLTP) systems. NoSQL databases differ from SQL by not adhering to the SQL standard. Examples include document-oriented, key-value, graph, and hierarchical databases. The selection of a storage system depends heavily on the data type, its characteristics, and how it will be used. For our experiment, three database types were chosen. Since Network Traffic Analysis (NTA) results often involve analytical queries, an OLAP database might be sufficient. Greenplum [1], a popular and advanced OLAP database with horizontal scaling and full SQL compatibility [2], is a strong contender. It also supports OLTP type of load. Document-oriented databases with full-text search capabilities are also interesting because they are schema-less and offer advanced search functionalities. Elasticsearch [3] is a well-known document-oriented database, but its license is not entirely open-source. This why, OpenSearch [4], a fork of Elasticsearch, will be considered. Given that NTA results often consist of interconnected data, a graph database could be a suitable option. While Neo4J [5], Dgraph [6], and others exist, Nebula Graph [7] was chosen for its proven horizontal autoscaling capabilities and, more importantly, its ability to handle large amounts of data [8].

Several experiments were conducted to identify the most suitable storage system for NTA results. These experiments focused on data load times, data acquisition metrics, and disk space usage. To ensure a fair comparison, all experiments used the same network traffic data set.

This paper is organized as follows. Section II describes the distinct characteristics of NTA data and the challenges associated with its storage. Then, It describes the general problem statement, structure, and unique characteristics of network traffic data. Section III includes analyzes of existing storage solutions for NTA, their strengths and weaknesses outlined. Section IV provides general design considerations and data schema used in experimental part. Section V contains evaluation results which include comparison of three database types for storing NTA results, evaluating their limitations and advantages. By leveraging a comprehensive analysis of these works and recent advancements in NTA research, we unveil a comparative evaluation of various database types. This in-depth exploration includes relational, NoSQL, and graph databases, meticulously highlighting

their strengths and weaknesses within the context of NTA data. Experimental results, including performance estimations, storage space requirements, and other relevant metrics, are presented. Section VI concludes article by summarizing the findings and highlighting potential areas for future research to improve storage solutions for NTA data.

2. Problem statement

For better understanding problems connected with storing NTA results, key characteristics of this type of stored data must be presented. First of all, storing and processing of network traffic falls under the concept of big data [9] due to its significant volume, variety, and speed of new data acquisition. Data can be generated at a high speed, which makes it difficult to process and store in real time. The volume of network traffic data can reach petabytes in large organizations, requiring scalable storage solutions and efficient processing methods. In article [9], suggested NewSql and NoSQL databases over relational databases, because they inappropriate on big datasets. In NTA systems time of occurring some events plays big part. This kind of systems often relies on temporal context, as patterns and anomalies can be associated with specific times, dates, or intervals. For example, time arrival of packets can be used as classification characteristic for deep learning methods [10-11]. Time stamps and temporal metadata are crucial for effective analysis and storage of network traffic, as they help to evaluate various statistical patterns in the large volume of network traffic. For instance, a burst in network traffic [12] at unusual hours might indicate suspicious activity, while recurring spikes during business hours could point to bandwidth bottlenecks. Therefore, time stamps and precise temporal metadata are crucial for efficient analysis and storage of NTA data. They empower security professionals to not only identify potential threats but also gain valuable insights into network usage patterns and resource allocation needs. By effectively leveraging temporal data, organizations can achieve a deeper understanding of their network behavior and proactively address potential security vulnerabilities and performance issues. Network traffic analysis can be categorized by two main levels of detail [13]: packet-level and flow-level. While packet-level network data offers a highly detailed view of traffic, it necessitates powerful, specialized equipment. This approach becomes impractical for expansive networks due to scalability limitations with a growing number of devices. Additionally, storing such granular data demands significant storage capacity. Flow-level analysis, on the other hand, provides a more scalable and privacy-conscious alternative. By aggregating packets, flow-level data offers a sufficient level of detail for network monitoring in modern environments with vast amounts of traffic and numerous connected devices.

3. Related work

The challenge of storing network traffic analysis results has been around for decades. Early attempts relied on relational databases. TelegraphCQ [14], for example, leveraged PostgreSql [15] for this purpose, with extensions for handling large continuous queries over ever-changing network data streams.

TelegraphCQ focuses on handling large streams of continuous queries over variable data streams. Later, timemachine [16] emerged, offering a cost-effective solution that utilized commodity hardware to buffer high-volume traffic for several days. The core principle behind this time machine lies in the “heavy-tailed” distribution of network traffic. This allows for capturing most connections in their entirety, while strategically skipping less critical data, all within a configurable per-connection byte limit. Some of the suggested approaches [17] based on special data format of network data, such as NetFlow [18] and IpFIX [19]. This kind of solutions facilitate the cost-effective monitoring of high-bandwidth links, using off-the-shelf hardware capabilities. Further this approach improved by NetMemex [20], provide network flow data with full packet payload. More advanced solution [21] separate constraints of static data analysis, exploiting high bandwidth cluster solutions which gain immediate insights into dynamic environments through seamless data

acquisition and analysis. Unveiling historical context with rapid responsiveness, empowering data-driven decision-making at every level. This comprehensive solution, often referred to as streaming data warehouses, represent a paradigm shift in network data analysis, offering unparalleled flexibility and agility. The researchers highlight a key advantage of their solution by contrasting it with existing systems like those [22] based on Apache Hadoop. While Hadoop offers valuable functionalities, it's limited to analyzing data captured at specific points in time (snapshots). This can be a disadvantage when real-time insights are crucial. Many of mentioned solutions didn't exist in free access nowadays. In the next paragraphs, several most recent and interesting technologies introduced.

PcapDB [23] is a distributed, open-source, and search-optimized packet capture system. It is designed to replace expensive commercial appliances with off-the-shelf hardware and a free, easy-to-manage software system. Captured packets are reorganized during capture by flow (an indefinite length sequence of packets with the same src/dst IPs/ports and transport protocol), indexed by flow, and searched (again) by flow. The indexes for the captured packets are relatively tiny (typically less than 1% the size of the captured data). Data captured in Pcap format is stored on-site at each Capture Node, minimizing network traffic. This setup enables cyber incident responders and analysts to swiftly search through indexed data instead of raw Pcap files, significantly cutting down query times, and allowing for searches across various capture locations. PcapDB indexes build on top of the PostgreSql but developers didn't describe them in detail. While indexes in PcapDB allows to fast search networks it strugle of luck of functionality. It allows only to search information through network and transport level of TCP/IP stack.

In 2017 group of researchers presented Moloch [24] an open source, large scale, full packet capturing, indexing, and database system. It builds on top of the Elasticsearch [3] database system. Later, developers of Moloch evolved it to Arkime [25]. Arkime group network packets by logical flows of data, grouping by network addresses information. Arkime store all possible data in one index. Every document in this index may consists of any of supported network protocol records. A key benefit of this approach is the ability to perform full-text searches across captured network traffic. This facilitates easy exploration of large datasets. However, using a full-text search engine like Elasticsearch can introduce performance overhead, especially for structured data like network traffic. Elasticsearch offers horizontal scaling by distributing data across nodes (sharding). While this enhances scalability, it can increase query complexity and impact performance. Replication ensures data redundancy but comes at the cost of higher storage consumption and write latency. Striking a balance between these features can be challenging. Additionally, large and frequently updated datasets can strain the indexing process, affecting search performance.

GRANEF [26] – Graph-based network forensics is a new approach to analyzing network traffic data. GRANEF toolkit utilizes Dgraph database for storing and querying data. Main advantages in links between various network members. It allows analysts to easily navigate and visually identify interesting network traffic. Dmitry Larin's 2023 master's thesis [27] proposed a novel approach for describing network topology using a multilevel graph model. This model leverages data from OSPF and BGP communication between routers to represent the network topology across different layers of the TCP/IP protocol suite. The approach facilitates analysis of the network's state at each level. The storage architecture utilizes two key components: Nebula Graph and Clickhouse. Nebula Graph is a graph database optimized for efficient graph queries, which is particularly well-suited for the type of data being analyzed. ClickHouse [28], an OnLine Analytical Processing (OLAP) database, serves as the storage for timeseries events. The proposed composite data storage approach leverages two types of databases to gain the advantages of each.

4. General design

4.1 Data schema

Main investigated data schema is based on [29] dissertation work. This data model offers a method for analyzing network interactions independent of underlying protocols. It considers network communication as a set of logic connections, there packets transfered between logical entities residing at the same level within the network architecture. This approach allows to don't get attached to specific protocol stacks, allowing for the flexible representation of diverse network interactions. Logic connection described by concrete network protocol type is called Context. Distinct instances of a defined context can be efficiently differentiated by their unique key. Key is usually presented in protocol headers and usually describes some address information. In general, context key is presented in binary format, but may be serialized in different formats. At this moment it serialized into json [30]. Json bring more flexibility in further processing and exporting of data. Set of instances of Context present a tree structure. In this tree each Context connected with it parent. Parent symbolize more lower layer in network stack. Example of context tree presented on Fig. 1.

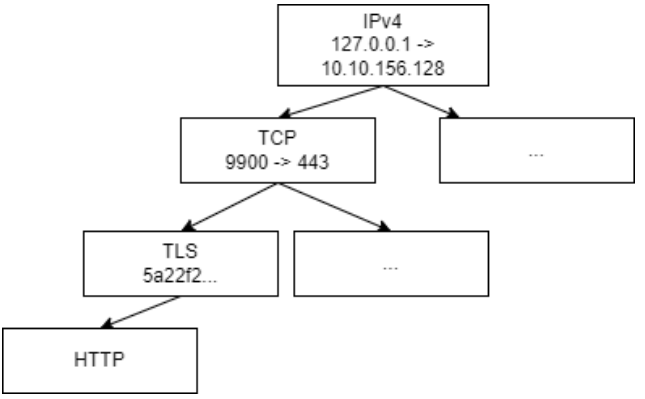


Fig. 1 Context tree

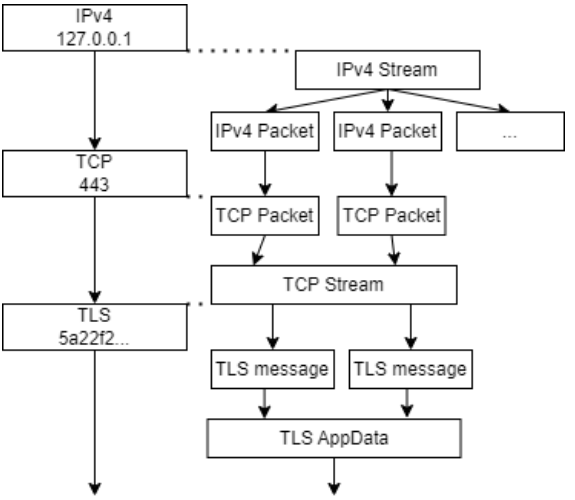


Fig. 2 Block tree

While context describe logical connections, payload of this connections either network streams or packets with header fields, presented by blocks. Block is universal structure which describes sequence of bytes, with some characteristics. Each block is assigned a parse type. Parse type is similar to Context type, and connected with type of network stream, name of network protocol packet or type of field of packet. Proposed data schema semantically separate blocks by two subtypes: stream block and fragment block. The stream block usually acts as a data source for another blocks. It may define dedicated network stream or original network data. Fragment block is simpler thing, it just defines network packet or protocol field. Every fragment block must have offset and size which define range of bytes in data source related to this block. Each block possesses a semantic connection to the specific context in which it appears.

Proposed data schema utilizes a three-tiered approach to represent network connections Fig. 2:

1) Top Level provides metadata describing the overall logical network connection across various network stack layers.
2) Second Level described by stream blocks represent unique network flows, each encapsulating a specific communication stream and it payload.
3) Last level delves into the most granular details, describing individual network packets or specific fields within those packets by Fragment block

4.2 Sql database (greenplum)

Greenplum [1] is an open-source data warehouse software built on top of PostgreSQL [15]. It is designed for handlinglarge datasets and complex queries. Greenplum's MPP architecture distributes data and workloads across multiple servers, enabling efficient processing of large-scale data warehouses and analytics tasks. It also supports some specific data types like net-types [31] which can be useful for network data analysis. Greenplum scales horizontally by adding more servers, this allows to seamlessly grow storage and processing capacity as overall data volumes increase. This advantage is very important for storing always increasing amount of network data. Furthermore, it is fully sql compatible and inherited most of PostgreSql functions. Main feature of greenplum is support of OLAP data storage format. As work with NTA results gets into description of big data there is no big need for OLTP operations like update or delete. OLAP databases are optimized for performing aggregations and complex queries on large datasets. This enables network analysts to quickly identify trends, pinpoint peak traffic periods, and analyze traffic patterns across different user groups or applications. Greenplum allows to recursively fetch rows by recursive Common Table Expression (CTE) [32]. This feature helps to easily work with context and block tree of supposed data schema. One of inherited features of PostgreSql is support of JSONB [33] data format. This kind of data can also be stored as raw text, but jsonb add aditional features like: more compact storage format, support of specialized functions for search. JSONB also supports indexing, which can be a significant advantage. All these features allow to easily search through Context keys of experimental data schema.

4.3 NoSQL database (OpenSearch)

OpenSearch [4] is open-source fork of elasticsearch popular full-text search database. OpenSearch excels at horizontal scaling by adding more nodes to the cluster. This allows to handle growing volumes of network traffic data efficiently without sacrificing performance. Advanced full-text search engine allows to easily find and filter some information across various network traffic fields (IP addresses, URLs, protocols) for in-depth analysis. It also supports of aggregation and extended analytic. OpenSearch's aggregation framework allows to analyze network traffic data from various angles. Moreover, it provides insightful visualizations, which can be used to identify trends, patterns, and anomalies in traffic flows. Unlike traditional databases with rigid schemas, OpenSearch offers a schema-less design. This provides flexibility to store and analyze network traffic data with diverse

formats and structures. While Elasticsearch is great for near real-time analysis, for very long-term historical data analysis, other solutions like data lakes might be more efficient. Index in OpenSearch defined by mappings. Each mapping consists of fields with any of the supported types. One of the supported types is Object [34]. An object field type contains a json object. A value in a json object may be another json object. This kind of fields automatically indexed by search engine and their containment may be used in queries. Big disadvantages of OpenSearch is hard working with related data. There is no some kind of sql join or reference. OpenSearch provide basic relations only between documents of one index by mechanism of parent/child relationship [35]. This approach didn't allow to represent tree like structures.

4.4 Graph database (Nebula graph)

Nebula graph [7] excels at modeling and querying relationships between data points. This could be beneficial for network traffic analysis if you want to understand how different devices, users, or IP addresses interact with each other. For instance, tracking connections within a malware outbreak or visualizing communication patterns within a network. Nebula boasts impressive query performance for interconnected data, allowing for efficient retrieval of specific network traffic flows based on relationships. Nebula can scale horizontally to handle growing volumes of network traffic data. While Nebula Graph Database offers a variety of data types, it doesn't natively support storing and indexing JSON data directly. However, it provides a workaround: JSON data can be dynamically converted (casted) into a specific data type called a "Map" [36]. This "Map" type essentially acts as a representation of a JSON object within Nebula Graph and can be used in complex queries for flexible data manipulation.

4.4 Binary data

Network traffic analysis results come in a variety of formats, and understanding these formats is crucial for effective storage and analysis. One key aspect to consider is that this data can often be presented in binary format. Common storage formats for network traffic captures include pcap (packet capture) and its successor, pcapng. These formats preserve the raw network packets, allowing for in-depth forensic analysis. Additionally, network traffic analysis can involve extracting network flow data, which summarizes conversations between devices rather than individual packets. Furthermore, network traffic analysis may involve processing captured data through specific data changing algorithms like decompression and decryption, which result must be stored in binary format. The only storage which supports raw bytes data is greenplum. It allows to store raw bytes in specific format called binary large objects (BLOB [37]). While Elasticsearch doesn't natively support raw bytes, it can accommodate binary data encoded in base64 format [38]. However, the process of encoding and decoding can introduce significant performance overhead, especially for large datasets. Moreover, binary data in OpenSearch is not searchable. Unlike Greenplum and Elasticsearch, Nebula currently doesn't offer direct support for storing raw byte data types. Binary data often presents indexing challenges for traditional databases. Storing it literally within a database might not be the most performant or scalable solution. Best practices for storing this kind of data lay in usage of external object storages, like CEPH, HDFS and etc. In this research raw bytes data will be ignored.

5. Evaluation

5.1 Experiment design

To ensure a fair comparison of storage systems for NTA results, a single network data set was used throughout the experiments for each of considered databases. This ensured a controlled environment for evaluating performance and facilitated a direct comparison of the systems' capabilities for loading and retrieving data. The first set of experiments investigated data loading times. Minimizing

this metric is crucial for NTA due to the large volumes of data typically collected and the limited storage capacity of collectors. Following successful data loading, storage space usage was analyzed. Subsequent experiments focused on data acquisition performance for loaded data. Here, the specific tree-based data schema, as described in Section IV, played a critical role. The goal was to efficiently extract data with minimal connections between elements. For metrics collection Opentelemetry [39] used. Opentelemetry uses tracing to capture the performance of requests or tasks. Percentiles can then be used to summarize the distribution of these execution times. In this context, p50 (50th percentile) would likely represent the duration at which 50% of requests took less time to complete, and p90 (90th percentile) would represent the duration at which 90% of requests took less time to complete.

5.2 Dataset

Basic experimental dataset it is a network trace in pcap format. Table 1 summarizes key statistics about the trace content. The trace serves as input for a network traffic analysis system. After processing, the results are loaded into one of the storage systems under consideration.

Table 1. Dataset statistics

Statistic	Value
Overall size, Mbytes	5893.71
Number of unique contexts	150355
Number of unique streams	19117
Number of IP packets	6067525
Number of TCP packets	2463840
Number of HTTP streams	2548

5.3 Hardware

The experiments were conducted using servers with specific hardware configurations. These servers included Intel(R) Xeon(R) Silver 4314 CPUs clocked at 2.40 GHz, 256 GB of memory, and Dell EMC VD SSDs. One important property for NTA storage is horizontal scaling. This type of scaling, often achieved through distributed databases, which allows to distribute the data load across multiple nodes. To leverage this benefit, all the considered databases were deployed in a multi-node configuration, with each database running on two nodes.

5.4 Data load

Different databases handle concurrent access in varying ways. For instance, Greenplum utilizes a centralized approach. Client requests are received by the master node, which then forwards them to the appropriate data node for processing. In contrast, OpenSearch employs a distributed client-based load balancing strategy. Here, any node within the cluster can potentially receive and handle client requests. It's important to note that in this experiment, data loading is performed synchronously within a single thread. Statistics of loading data into databases presented in Table 2. In Table 2, C refers to Context and S refers to Stream.

Meaningful characteristic for consider is size of stored data. Table 3 provides an overview of the storage space statistic for all types of databases.

5.5 Data query

For each database had executed queries which return same data. It is set of queries that return address information about HTTP streams by defined source IP address. In case of OpenSearch it is impossible to get connected data in one query. Acquisition of required data might be achieved only

be series of queries. Total number of executed queries for each database is equal 330. Results of HTTP streams acquisition presented in Table 4.

Table 2. Load statistics

	Greenplum		OpenSearch		Nebula	
	C	S	C	S	C	S
Time, s	1087		770		1012	
p50, ms	5.9	5.9	4.2	3.5	5.6	5.7
p90, ms	7.1	7	4.6	3.8	6.1	6.2
p99, ms	7.5	7.5	5.1	4.9	6.4	6.5
min, ms	3.4	4.07	2.74	2.48	3.18	4.65
max, ms	54.25	29.4	217.5	153.4	212.1	8.44
SD, ms	0.73	0.83	0.69	2.06	0.76	0.44
spans/min	7160	910	9400	1190	7520	954

Table 3. Space statistics

Space stats	
Greenplum, MB	181
Nebula, MB	115.7
OpenSearch, MB	20.7

Table 4. Fetch statistics

	Greenplum	OpenSearch	Nebula
p50, ms	80.4	18.9	170
p90, ms	83.1	140.4	176.6
p99, ms	92.8	264.9	181.5
min, ms	68.9	15.5	155.5
max, ms	140.9	348.5	198.7
SD, ms	5.05	62.2	5.7

It’s worth noting that queries to OpenSearch performed in several requests, yet it still achieved impressive read speeds. The exceptional read speed of OpenSearch might be attributed to its handling of the “context key”, the most valuable field for search within the considered data schema. Unlike Greenplum and Nebula, which required special type casts and functions for searching this JSON field, OpenSearch natively stored and indexed it in its original format. This native handling likely contributed to the superior read performance.

6. Conclusion

In this paper, appliance of different storage systems for NTA results was considered. The investigation focused on their suitability for storing and facilitating analysis of complex network data. This article evaluated the suitability of three storage systems (OpenSearch, Nebula Graph, Greenplum) for storing and analyzing complex network traffic data (NTA results). While OpenSearch offers the fastest write speeds and space efficiency, it lacks the ability to represent intricate relationships within the data. Nebula Graph excels at modeling these relationships but may not be ideal for very high data volumes. Greenplum provides a traditional relational model, allowing for flexible analysis but potentially consumes more storage space. The key takeaway is that there’s

no one-size-fits-all solution. The optimal storage system depends on specific network needs: High Data Volume: All evaluated databases can horizontally scale to accommodate significant data growth. Intricate Relationships: Nebula Graph excels at modeling complex network connections, making it ideal for scenarios where understanding these relationships is crucial. Loosely Connected Data: Greenplum's relational model with SQL support allows efficient storage of data points with fewer connections. Schema Flexibility: OpenSearch offers the most flexibility for storing various NTA data structures due to its schema-less architecture. The optimal storage solution depends on the specific requirements of the network environment and analysis needs. OpenSearch performed well in write benchmark tests, achieving the lowest time in load experiments. Moreover, OpenSearch impressed with its space efficiency. Impressively, in 50% of fetch experiments, this database outperformed all others in speed. However, it did experience occasional performance spikes. Greenplum showed average results in read tests and more important it had lowest deviation of values. Nebula performs well on load tests but it had some problems with fetching the data. Most inefficient space consumption demonstrate Greenplum. Future research could involve conducting controlled experiments to quantify the performance and scalability of each storage system under varying network traffic loads. Additionally, investigating the integration of these systems into a comprehensive NTA framework could offer valuable insights for network security professionals. By carefully considering the volume, structure, and desired analysis depth of their network data, organizations can select the most suitable storage system for their NTA efforts.

References

[1]. "Greenplum database." (2024), [Online]. Available at: https://greenplum.org, accessed on 03.20.2024.

[2]. Z. Lyu, H. H. Zhang, G. Xiong, et al., "Greenplum: A hybrid database for transactional and analytical workloads," in Proceedings of the 2021 International Conference on Management of Data, 2021, pp. 2530–2542.

[3]. "Elasticsearch." (2024), [Online]. Available at: https://www.elastic.co/elasticsearch, accessed on 03.22.2024.

[4]. "Opensearch." (2024), [Online]. Available at: https://opensearch.org, accessed on 03.20.2024.

[5]. "Neo4j graph database & analytics — graph database management system." (2024), [Online]. Available at: https://neo4j.com, accessed on 03.22.2024.

[6]. "Dgraph — graphql cloud platform, distributed graph engine." (2024), [Online]. Available at: https://dgraph.io, accessed on 03.22.2024.

[7]. "Open source distributed graph database — nebula graph." (2024), [Online]. Available at: https://www . nebula-graph.io, accessed on 03.22.2024.

[8]. M. Wu, X. Yi, H. Yu, Y. Liu, and Y. Wang, "Nebula graph: An open source distributed graph database," arXiv preprint arXiv:2206.07278, 2022.

[9]. A. D'Alconzo, I. Drago, A. Morichetta, M. Mellia, and P. Casas, "A survey on big data for network traffic monitoring and analysis," IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 800–813, 2019.

[10]. Гетьман А. И., Иконникова М. К. Обзор методов классификации сетевого трафика с использованием машинного обучения. Труды ИСП РАН, том 32, вып. 6, 2020 г., стр. 137-154 / GETMAN A. I., Ikonnikova M. K., "A survey of network traffic classification" Trudy ISP RAN/Proc. ISP RAS, vol. 32, no. 6, pp. 137–154, 2021 (in Russian).

[11]. S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," IEEE communications magazine, vol. 57, no. 5, pp. 76–81, 2019.

[12]. D. Wei, F. Shi, and S. Dhelim, "A self-supervised learning model for unknown internet traffic identification based on surge period," Future Internet, vol. 14, no. 10, p. 289, 2022.

[13]. M. Piskozub, R. Spolaor, and I. Martinovic, "Compactflow: A hybrid binary format for network flow data," in Information Security Theory and Practice: 13th IFIP WG 11.2 International Conference, WISTP 2019, Paris, France, December 11–12, 2019, Proceedings 13, Springer, 2020, pp. 185–201.

[14]. S. Chandrasekaran, O. Cooper, A. Deshpande, et al., "Telegraphcq: Continuous dataflow processing," in Proceedings of the 2003 ACM SIGMOD international conference on Management of data, 2003, pp. 668–668.

[15]. “Postgresql: The world’s most advanced open source database.” (2024), [Online]. Available at: https://www. postgresql.org/, accessed on 03.20.2024.
[16]. S. Kornexl, V. Paxson, H. Dreger, A. Feldmann, and R. Sommer, “Building a time machine for efficient recording and retrieval of high-volume network traffic,” in 5th Internet Measurement Conference, USENIX Association, 2005, pp. 267–272.
[17]. A. Ba¨r, P. Casas, L. Golab, and A. Finamore, “Dbstream: An online aggregation, filtering and processing system for network traffic monitoring,” in 2014 International Wireless Communications and Mobile Computing Conference (IWCMC), IEEE, 2014, pp. 611–616.
[18]. B. Claise, “Cisco Systems netflow services export version 9,” RFC Editor, RFC 3954, Oct. 2004. [Online]. Available at: https://www.rfc-editor.org/rfc/rfc3954.txt.
[19]. B. Claise, B. Trammell, and P. Aitken, “Specification of the ip flow information export (ipfix) protocol for the exchange of flow information,” RFC Editor, RFC 7011, Sep. 2013. [Online]. Available at: https://www.rfc- editor.org/rfc/rfc7011.txt.
[20]. H. Lim, V. Sekar, Y. Abe, and D. G. Andersen, “Netmemex: Providing full-fidelity traffic archival,” arXiv preprint arXiv:1603.04387, 2016.
[21]. M. Wullink, G. C. Moura, M. Mu¨ller, and C. Hesselman, “Entrada: A high-performance network traffic data streaming warehouse,” in NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium, IEEE, 2016, pp. 913–918.
[22]. Y. Lee and Y. Lee, “Toward scalable internet traffic measurement and analysis with hadoop,” ACM SIGCOMM Computer Communication Review, vol. 43, no. 1, pp. 5–13, 2012.
[23]. S. I. Steinfadt and P. S. Ferrell, “Packet capture solutions: Pcapdb benchmark for high-bandwidth capture, storage, and searching,” Los Alamos National Laboratory (LANL), Los Alamos, NM (United States), Tech. Rep., 2017.
[24]. J. Uramova´, P. Segecˇ, M. Moravcˇ´ık, J. Papa´n, T. Mokosˇ, and M. Brodec, “Packet capture infrastructure based on moloch,” in 2017 15th International Conference on Emerging eLearning Technologies and Applications (ICETA), IEEE, 2017, pp. 1–7.
[25]. “Arkime.” (2024), [Online]. Available at: https://arkime. com, accessed on 03.20.2024.
[26]. M. Cermak and D. Sramkova, “Granef: Utilization of a graph database for network forensics.,” in SECRYPT, 2021, pp. 785–790.
[27]. D. Larin, “Razrabotka i primenenie modeli opisanija mnogourovnevyh setevyh topologij dlja reshenija zadachi monitoringa i modelirovanija setevoj infrastruktury,” Master’s thesis, Moscow Institute of Physics and Technology (National Research University), Moscow, Jun. 2023.
[28]. “Fast open-source olap dbms clickhouse.” (2024), [Online]. Available at: https://clickhouse.com, accessed on 03.22.2024.
[29]. Маркин Ю.В., “Методы и средства углубленного анализа сетевого трафика”, диссертация на соискание ученой степени кандидата технических наук, год защиты 2017, ИСП РАН, 80 с. / Markin Y.V., “Metody i sredstva uglublennogo analiza setevogo trafika”, dissertacija na soiskanie uchenoj stepeni kandidata tehnicheskih nauk, 2017, ISP RAN, 80 p, (in Russian).
[30]. T. Bray, “The JavaScript Object Notation (JSON) Data Interchange Format,” RFC 7159, Mar. 2014, 16 pp. [Online]. Available at: https://www.rfc- editor.org/info/ rfc7159.
[31]. “Postgresql: Documentation: 16: 8.9. network address types.” (2024), [Online]. Available at: https://www.postgresql.org/docs/current/datatype-net-types.html, accessed on 03.22.2024.
[32]. “Greenplum: With queries (common table expressions).” (2024), [Online]. Available at: https://docs.vmware.com/en/VMware-Greenplum/7/greenplum-database/admin_guide-query-topics-CTE- query.html, accessed on 03.22.2024.
[33]. “Postgresql: Documentation: 12: 8.14. json types.” (2024), [Online]. Available: https://www.postgresql.org/, accessed on 03.21.2024.
[34]. “Object – opensearch documentation”. (2024), [Online]. Available at: https://opensearch.org/docs/latest/field-types/supported-field-types/object/, accessed on 03.21.2024.
[35]. “Join – opensearch documentation”. (2024), [Online]. Available at: https://opensearch.org/docs/latest/field-types/supported-field-types/join/, accessed on 03.22.2024.
[36]. “Map – nebulagraph database manual”. (2024), [Online]. Available at: https://docs.nebula-graph.io/3.6.0/3.ngql-guide/3.data-types/8.map/, accessed on 03.20.2024.
[37]. “Postgresql: Documentation: 12: 8.4. binary data types”. (2024), [Online]. Available at: https://www.postgresql.org/ docs/12/datatype-binary.html, accessed on 03.20.2024.
[38]. “Binary – opensearch documentation”. (2024), [Online]. Available: https://opensearch.org/docs/latest/field-types/supported-field-types/binary/, accessed on 03.20.2024.

[39]. “Opentelemetry.” (2024), [Online]. Available: https://opentelemetry.io, accessed on 03.22.2024.

Appendix A. Acronyms

CTE Common Table Expression. 4
NTA Network Traffic Analysis. 1, 2, 4–6
OLAP OnLine Analytical Processing. 2–4
OLTP Online Transaction Processing. 2, 4

Информация об авторах / Information about authors

Владислав Игоревич ЕГОРОВ – аспирант, стажёр-исследователь ИСП РАН. Научные интересы: обработка, анализ и хранение результатов анализа сетевого трафика.

Vladislav Igorevich EGOROV – postgraduate student, intern researcher at ISP RAS. Research interests: processing, analysis and storage of network traffic analysis results.

Роман Евгеньевич ПОНОМАРЕНКО – аспирант, стажёр-исследователь ИСП РАН. Научные интересы: архитектура программного обеспечения, оптимизация программ, глубокий анализ сетевого трафика.

Roman Evgenevich PONOMARENKO – postgraduate student, intern researcher at ISP RAS. Research interests: software architecture, program optimization, deep packet inspection.

Александр Игоревич ГЕТЬМАН – кандидат физико-математических наук, старший научный сотрудник ИСП РАН, ассистент ВМК МГУ и МФТИ, доцент ВШЭ. Сфера научных интересов: анализ бинарного кода, восстановление форматов данных, анализ и классификация сетевого трафика.

Aleksandr Igorevich GETMAN – Cand. Sci. (Phys.-Math.), senior researcher at ISP RAS, assistant at CMC MSU and MIPT, associate professor at HSE. Research interests: binary code analysis, data format recovery, network traffic analysis and classification.

DOI: 10.15514/ISPRAS-2024-36(2)-2

TQL: a Case Study of Integrating DSL in a Product

A.D. Belousov, ORCID: 0009-0005-3641-1252 <*flygrounder@yandex.ru*>

Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia.
Higher School of Economics,
20, Myasnitskaya st., Moscow, 101000, Russia.

Abstract. Domain-specific languages power numerous modern applications and libraries, including but not limited to: Wolfram Alpha, Microsoft Excel, Graphviz. This work aims to share the experience gathered from developing TQL (Talisman Query Language) – a domain-specific language used in Talisman platform. Talisman platform is a set of tools to automate data processing tasks, developed by Ivannikov Institute for System Programming of RAS. TQL implementation, discussed in this article, supports error-recovery, can be run directly inside a browser as well as on a server, and it also has an interactive playground that visualizes the parse tree while typing. This article describes several techniques and technologies that were used to make these qualities possible while keeping a single, maintainable codebase.

Keywords: programming; compilers; domain-specific languages.

For citation: Belousov A.D. TQL: a case study of integrating DSL in a product. *Trudy ISP RAN/Proc. ISP RAS*, vol.36, issue 2, 2024. pp. 21-32. DOI: 10.15514/ISPRAS-2024-36(2)-2.

TQL: тематическое исследование внедрения предметно-ориентированного языка в продукт

А.Д. Белоусов, ORCID: 0009-0005-3641-1252 <*flygrounder@yandex.ru*>

Институт системного программирования РАН,
Россия, 109004, г. Москва, ул. А. Солженицына, д. 25.
Высшая школа экономики,
Россия, 101000, г. Москва, ул. Мясницкая, д. 20.

Аннотация. Предметно-ориентированные языки используются во многих современных приложениях и программных библиотеках, среди них: Wolfram Alpha, Microsoft Excel, Graphviz. Данная работа описывает опыт, полученный при разработке TQL (Talisman Query Language) – предметно-ориентированного языка, используемого в платформе Talisman. Платформа Talisman – набор инструментов для автоматизации задач обработки данных, разработанный институтом системного программирования РАН. Реализация TQL, описанная в данной статье, поддерживает восстановление после ошибок, может запущена как в браузере, так и на сервере, и имеет интерактивную среду, позволяющую визуализировать дерево разбора во время печати. Данная статья описывает подходы и технологии, которые сделали данные качества возможными при единой, поддерживаемой кодовой базе.

Ключевые слова: программирование; компиляторы; предметно-ориентированные языки.

Для цитирования: Белоусов А.Д. TQL: тематическое исследование внедрения предметно-ориентированного языка в продукт. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 21–32 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–2.

1. Introduction

A domain-specific language (DSL) is a computer programming language of limited expressiveness focused on a particular domain. [1] There are multiple reasons why one might choose to integrate a DSL in their application:

- It can be used by people, who are not familiar with the traditional software engineering, but are familiar with the application domain
- It can be used to limit actions that users can perform in order to preserve security and privacy of other users
- It can be more expressive than general-purpose languages in a particular domain

This work is based on the experience, gathered from developing TQL — a query language for Talisman platform. [2] Talisman is a unified set of tools that automate typical data processing tasks, such as data retrieval, integration, analysis, storage and visualization. It is developed by Ivannikov Institute for System Programming of RAS. One of the core components of the platform is it's knowledge base. It stores documents that can be related to each other, they can be searched by using both relational and full-text patterns. In order to allow such search queries, data is stored in two different systems: in a relational database (PostgreSQL) and a full-text search system (Elasticsearch). Both have their own query languages, which allow only one type of search. In order to combine capabilities of both languages, a new domain-specific language was developed — Talisman Query Language.

User experience for a DSL or any programming language in general depends on the language support and tooling surrounding it. The initial implementation of TQL included only a parser for query execution, lacking language support for users to write queries. This led to the following problems while using the language:

- No syntax error reporting. When a user accidentally forgot to close a bracket or made any other type of syntax error, they would have to find this error manually. This can be improved by reporting which tokens in a query led to a syntax error.
- Difficult to navigate inside complex queries. When having a complex query with nested selectors, it becomes troublesome to identify, which part of the query is nested inside which selector. This can be improved by highlighting keywords and highlighting matching brackets with the same color.
- Having to read the manual to recall the correct keyword. While being powerful, TQL has a lot of keywords that one needs to learn in order to use it efficiently. Suggesting keyword completions can ease that learning curve for users.

A solution based on the existing TQL parser was considered, but it had the following technological limitations:

- Language parser did not include error recovery, so it could only find the first syntax error.
- Parser could only be run in the same environment as the rest of Talisman backend: on a server.

Based on these issues, it was decided to create a new TQL implementation.

The primary focus of this article is to demonstrate how a single implementation of a DSL can offer a seamless editing experience within a browser, while also being utilized on a server for processing user programs. Attempting to meet these requirements poses several challenges, which are outlined in this article along with the solutions employed by the new TQL implementation.

2. Background

2.1 Parser implementation

There are several choices when implementing a parser:

- Implementing it manually
- Using a combinator library
- Using a parser generator

A manual implementation provides developers with the highest degree of flexibility and control over parser behavior. It allows implementing custom error recovery mechanisms and choosing any parsing algorithm. However, this flexibility comes at a cost: manual parser implementations tend to be repetitive and error-prone.

Combinator libraries provide useful building blocks for building a parser. These libraries make parser code easier to read and more composable: each parsing routine is a combinator, that can be then used in other combinators. This approach is less flexible than the manual one: combinator libraries are built on some particular parsing algorithm. Also, they tend to have problems with error recovery, but more on this later.

Parser generators use the most automated approach: they accept a formal grammar as an input and generate a parser automatically. Generators usually have some ways to customize the resulting parser with user-written code, but these capabilities are limited compared to manual parsing and combinator libraries. Error recovery capabilities are usually limited with some set of predefined options, that can be customized only to some degree.

2.2 Error recovery in combinator libraries

Traditionally, combinator libraries and parser generators implemented some variations of LR parsing. This approach gives some flexibility in expressing a language grammar: for example it allows left-recursive rules. However, it makes writing custom error recovery difficult. Usually libraries and generators provide some predefined strategies for error recovery. These strategies are usually enough for error reporting, but not for providing completions. This is one of the reasons, why popular language servers implement manual recursive-descent parsers with custom recovery logic.

There are some combinator libraries, that use recursive descent algorithm and provide capabilities for custom recovery logic: megaparsec [3] for Haskell and chumsky [4] for Rust. Recursive descent imposes some limitations on grammar: for example it does not allow left-recursive rules. Other than that, this approach provides usability of combinator libraries with flexibility of manual parsing.

2.3 CST vs AST

The initial implementation of TQL parsed a query directly into an abstract syntax tree (AST). AST is a tree, representing a program's structure. As an example, if-then-else node would have three children: a condition, a positive branch and a negative branch. This structural information is necessary to analyze and execute the program. AST is an appropriate abstraction for executing a query, but is not appropriate for providing editor support. While editing a query, most of the time, user input won't be syntactically correct, so AST cannot be built. In such contexts, concrete syntax tree or CST is often used. Unlike AST, each token of original input is included in CST with it's coordinates (row, column) from the original input. When a parser cannot parse the program, it might skip some tokens, presenting them as an error node. This way a partial tree can be built and analyzed.

In TQL implementation both CST and AST are used:

- Initially, a parser produces a CST, which can be partial in case of syntax errors. This tree is used to provide language support while editing.

- In case no syntax errors were discovered, CST can be transformed to an AST. It is used by Talisman backend for executing a query.

2.4 Language server protocol

Features like syntax highlighting and autocomplete are provided by code editors and IDEs, in this section we will look into problems that they face and solutions that emerged over time.

Language support has been traditionally provided for each editor individually. Usually, that would require developing a plugin for each editor, using different technologies and architecture. This leads to duplication of efforts: adding a single feature has to be done separately for each supported editor. Inconsistency is another issue with this model: each editor might have different language features available.

These problems led to the creation of Language Server Protocol [5] or LSP for short. It was created by Microsoft for Visual Studio Code editor in 2016 and since then became an industry standard for providing language support. This protocol defines messages that a language client (an editor) and a language server use to communicate with each other. These messages are language-independent, treating a program as a text file and utilizing text coordinates (rows and columns). This protocol allowed editor developers to implement it, benefiting from all language servers that also implement it. Developers working on editor support could now implement language features once and benefit from all the editors implementing a client.

Fig. 1 shows messages defined by LSP, that client and server send each other.

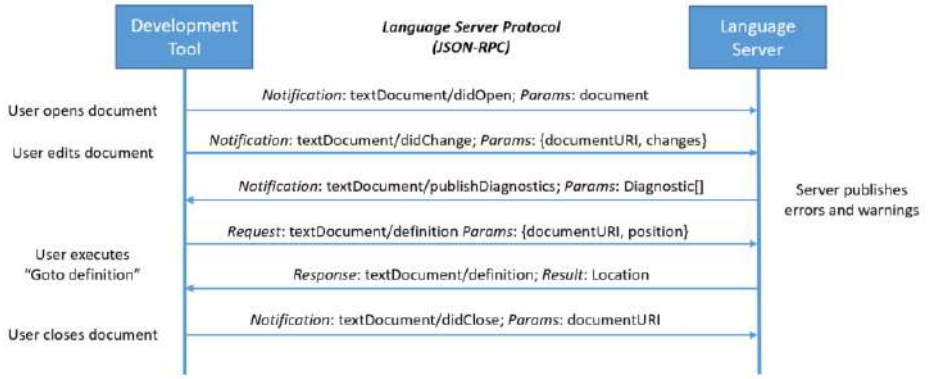


Fig. 1. Language Server Protocol

We've decided to use this protocol to provide language support for TQL. It allows integrating our language with any tooling that also implements this protocol.

3. Related Work

One tool that can be used to create compiler frontends is Bison. [6] In order to use it, one must describe a grammar and semantic actions that correspond to different grammar rules. Bison tool then generates a parser that uses GLR algorithm in order to process inputs. However, LR parsers tend to be less error-resilient than recursive-descent ones, as shown in this [7] article. Error-resilience is an important property when it comes to language servers, that is why this work does not use an LR parser generator.

Not all parser generators use variations of LR algorithm. One prominent example that does not use it is ANTLR [8]. Generated parsers implement LL(*) algorithm, which uses top-down approach. ANTLR has a few built-in error recovery strategies and it allows implementing custom ones. Built-in strategies use generalized algorithm, that can produce decent results, but the best error recovery can be achieved by using custom error strategies for different rules. Each rule in the generated code

is wrapped in a try/catch block that, in case of an error, reports it and attempts to recover. However, recovering from low-level rules can produce undesirable syntax trees as these rules may lack some context that higher-level rules have. In order to disable this behaviour, one must explicitly rethrow an error in all such rules. This clutters the code and makes it less maintainable. In a manual parser, errors might be handled only in rules where it is relevant, avoiding clutter in rules where it is not. While ANTLR theoretically has enough flexibility to implement different error recovery strategies for different rules, it makes it quite difficult in practice, as it is not an intended use case. Xtext [9], that was specifically created to make writing language servers easier, uses ANTLR for parsing and thus suffers from the same problems.

4. Implementation

4.1 Requirements

The new implementation of TQL, that we will call TQLS (Talisman Query Language Server), had to meet the following requirements:

- It had to provide an HTTP API to be used programmatically and a frontend library to be used directly in user's browser
- Language implementation should support:
  - Error reporting
  - Syntax highlighting
  - Autocomplete
- It had to be integrated with the main Talisman backend

These requirements determined technologies and techniques used to implement the language.

4.2 Introduction to TQL

TQL is used to search for entities in Talisman knowledge base. Most common entity types include:

- Document – a piece of information that was retrieved from some source.
- Concept – an entity, that is known to the knowledge base. Concepts can be mentioned in documents.
- Link – represents a connection between two concepts.

Concepts and links form a typed labeled property graph. Each concept and each link has a type, that determines their properties. Consider the following example: document might mention two people, that are known to the knowledge base as concepts. These concepts would have type "person". Two people might have a link of type "friendship", that links them together. Concepts of type "person" might have properties like name, birth date, gender, etc. Links of type "friendship" might have a property that says when this friendship was established.

TQL operators can be divided into the following groups:

- Full-text operators are used to search in the text representation of documents.
- Relational operators are used to find entities based on their properties and their relations to other entities.
- Universal operators can be applied both to full-text and relational expressions.

Queries below illustrate some full-text operators:

- *apple* – searches for documents that contain word "apple".
- *-apple* – searches for documents that do not contain word "apple".
- *apple~* – searches for documents that contain "apple" using fuzzy search.

- *"apple pie"~1* – searches for documents that contain phrase "apple pie" with at most one word between them.

The following queries demonstrate relational operators:

- *concept(apple)* – searches for documents mentioning a concept named "apple".
- *concept.fruit(apple)* – the same as above, but this concept should have a type "fruit".
- *identifier=OK-1* – searches for a document with the identifier "OK-1"'.
- *concept(document(identifier=OK-1))* – searches for documents that mention some concept that is also mentioned in a document with the identifier OK-1. Selectors can be nested inside each other.

Queries below demonstrate universal operators that can be used with both full-text and relational expressions:

- *concept(apple) fruit* – searches for documents mentioning a concept named "apple", while also containing a word "fruit" in them. Logical "and" is represented as a space.
- *concept(apple),fruit* – searches for documents mentioning concept named "apple", or containing a word "fruit" in them. Logical "or" is represented as a comma.
- *concept(apple) (fruit,food)* – searches for documents mentioning a concept named "apple" and containing a word "fruit" or "food" in them. Parentheses change evaluation order. By default, "and" has a higher priority than "or".

Evaluation of TQL queries is performed by the Talisman backend and is beyond the scope of this work. One important consideration is that TQL queries are not compiled ahead-of-time to SQL and Query DSL. Instead, the resulting AST is evaluated by Talisman backend in a bottom-up manner, using the appropriate search mechanism at each step.

4.3 Language as a Service

It might seem natural to include a DSL implementation into the host application's codebase. However, treating a DSL as a separate product provides several benefits:

- One might choose a more appropriate tech stack for implementing a DSL.
- It allows using a DSL in different environments: i.e., inside a native app, on a website and on a server.
- Makes it easier for a different team to work on the project: they can use a different release schedule, version control and tooling.

For the remaining of this work, we will be calling this approach "Language as a Service" or LaaS. LaaS allows one to design and implement a DSL separately from designing and developing the host product. Modern applications tend to be multi-platform: there can be a native application, a website and an HTTP API. By abstracting away platform-specific details, one might focus on making a better DSL. API for integrating with platform-specific code tends to be a thin wrapper of this core abstract API. Later we will discuss possible technology choices, that allow such integrations.

4.4 Running TQLS inside a browser

Editors, providing language features like autocomplete, that were examined during our research made requests to a remote server for providing completions using either HTTP or WebSocket protocol. This approach leads to more delay before providing completions due to network requests, requires a constant Internet connection and creates more load on servers. This is why one of the design goals for TQLS was being able to run it inside a browser.

Traditionally, writing something for the browser assumed a particular tech stack: HTML, CSS and JavaScript. While these technologies still power the majority of the web, there are some new ones

that allow using a different stack. One of the most prominent ones is WebAssembly. It defines a special binary format that is supported in modern browsers, that can be executed safely inside a sandbox. Different languages support compilation in WebAssembly which allows a wider tech stack choice. These binaries can be distributed using conventional npm packages which makes it easy to use for frontend developers.

4.5 Running TQLS on a server

While language features like providing completions can be run inside a browser, there still was a need in providing a server access to parser. Talisman backend had an old implementation of TQL parser. Supporting two different parsers written in two different programming languages would impose a huge burden on the developers and would lead to inconsistencies between language support for the user and for executing queries. TQLS uses GraphQL [10] API for exposing a backend API.

GraphQL has several benefits over traditional REST APIs:

- API clients can be generated from the schema
- Schema can be automatically validated against having breaking changes
- There is no need to write a separate endpoint for each possible query. Server exposes all the data it has available and client selects the exact subset it needs

Tree is represented as a flat list, where each element has a unique identifier. Different element types have different GraphQL types, all of which implement Element interface.

```
type Tree {
    rootId: String
    nodes: [Element!]!
}
interface Element {
    data: Metadata!
}
type Metadata {
    id: String!
    start: Int!
    end: Int!
}
```

4.6 Error recovery

Error recovery is implemented as a set of heuristics, that are based on common usage patterns for TQL. Error recovery may add two additional node types to the resulting CST:

- Skipped tokens – tokens that were skipped by parser. These tokens are still preserved in a tree, providing more context for code completion.
- Recovered tokens – these tokens are not present in the query, yet they were expected and were added in the tree by a parser, preserving a correct tree structure.

The following is an example of a heuristic, used by TQLS: whenever a parser encounters a closing parenthesis after the left part of the equation (an identifier and an equation sign), it does not skip it, but instead adds a recovered operand in a tree and continues parsing as normal. This way, less input is skipped and the resulting tree is more useful for code completions generation.

Figure 2 shows an example of a CST for a query $$$ (test=). Both types of additional nodes were added by the error recovery procedure. Unquoted dollar signs are not allowed, that is why they were

skipped. After an equals sign, a parser encountered a closing parenthesis and it inserted a recovered operand in a tree.

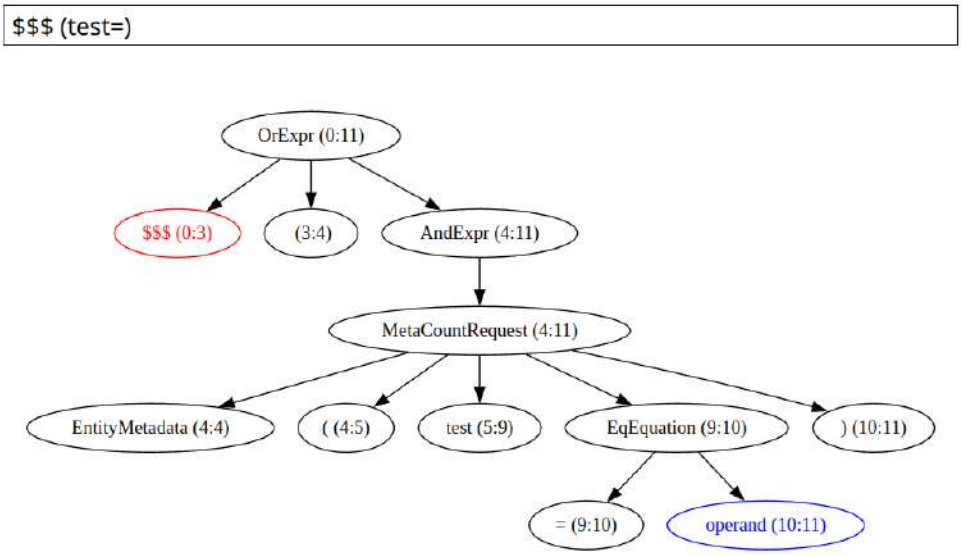


Fig. 2. CST with skipped and recovered tokens

In order to report errors to the user, a recursive procedure searches tree for both types of nodes and combines them in a single list. For the above example, two errors will be reported:

1. *Start: 0 | End: 3 | Found: $$$ | Expected: expression*
2. *Start: 10 | End: 11 | Found:) | Expected: operand*

4.7 Code completion

Code completion procedure takes a query and a cursor position as an input and returns a list of completions as an output. Each completion is a text range and a string, that should replace that range if the completion is accepted. Completion procedure is split into two stages:

- Determine a completion context. Completion context consists of a range inside a query and the current selector. The current selector is the innermost selector, that surrounds the cursor.
- Make a list of completions based on the provided context.

In order to find a completion context, the first step is to find a token under cursor in a tree. To do that, a recursive procedure iterates through root's children from left to right and makes a recursive call if cursor is contained within a child node's range. This way, if cursor is placed between two tokens, the left one will be chosen by the procedure. This is due to the fact, that completion is expected to finish token before the cursor, not after. Token and it's ancestors are then used to determine a completion context. An example, where the found token is not an entire completion context, is when dealing with multi-word properties, which are written in quotes. Consisting of several tokens, they should be considered a single completion context.

After completion context is found, it's text is compared with known keywords. A list of available keywords is determined by the current selector.

4.8 Programming language

TQLS uses Rust as an implementation language. Rust was chosen due to the following reasons:

- Chumsky combinator library uses recursive descent algorithm and allows custom error recovery strategies.
- Robust support for WebAssembly, allowing it to run in a browser.
- Algebraic Data Types (enums) which are useful for modelling syntax trees.
- Robust GraphQL support, which allows implementing a backend API.

Rust-analyzer [11] was used as a source of inspiration for designing TQLS.

4.9 Type-safe CST API

Parser produces a concrete syntax tree, which reflects grammar rules. Unlike AST, CST is what we call a homogeneous tree: each node uses the same structure in code. Code below demonstrates structures used to represent this tree in a program:

```
struct Node {
  kind: NodeKind,
  meta: NodeMetadata,
  children: Vec<Child>,
}
enum Child {
  Node(Node),
  Token(Token),
}
struct Token {
  kind: TokenKind,
  meta: NodeMetadata,
  text: String,
}
```

Using the same node type for all tree nodes simplifies writing general tree traversal algorithms. However, when one needs to work with a specific node type, this API can be error-prone. On top of this API, there is a type-safe API which is generated from the tree description. This description is written in Ungrammar [12] format. As an example, let's take a look at the following nodes:

```
AndExpr = Factor (' ' Factor)*
Factor = StructOperators
  | FlatFactor
  | ElementParens
```

Custom source generator will generate the following structures:

```
struct AndExpr<'a> {
  node: &'a Node,
}
impl<'a> AndExpr<'a> {
  fn get_factor_list(&self) -> Vec<Factor<'a>> {
  tree::get_children_with_type(self.node)
}
enum Factor<'a> {
```

```
        StructOperators(StructOperators<'a>),
        FlatFactor(FlatFactor<'a>),
        ElementParens(ElementParens<'a>),
    }
```

There are currently over 70 node types in an Ungrammar description and more than 3500 lines of code generated from it. Writing this code manually would be error-prone and since the tree structure can change over time, updating this file manually will be a maintenance burden.

4.10 Interactive playground

In order to debug and test TQLS, an interactive playground application was developed. It runs inside a browser using WebAssembly. This way, it is possible not only to test our application and visually validate the resulting tree, but also to ensure that the exposed API is sufficient and is easy to use. In order to render tree in a browser, TQLS serializes it as a DOT graph and playground renders it using a graphviz library.

Figure 3 shows an AST for a query SYRCoSE, Young Researchers rendered inside a playground.

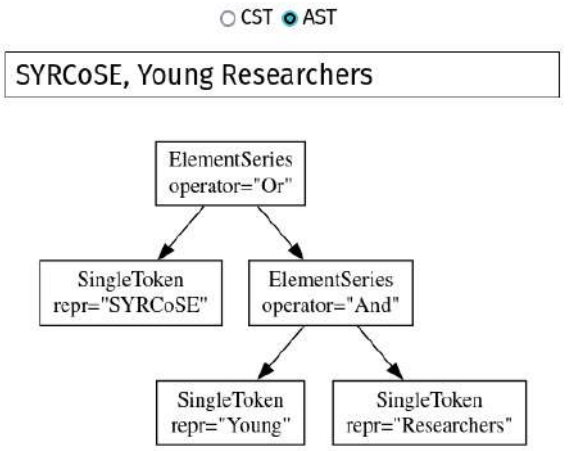


Fig. 3. Interactive playground

5. Conclusion

This work describes our experience developing and integrating a DSL with the rest of our services. Our focus was on making it maintainable and portable, here is a summary of our solutions to this problem:

- Treating a DSL as a separate application with it's own architecture and tech stack
- Using CST for editor support and AST for executing queries
- Implementing Language Server Protocol to integrate with any tools supporting it
- Using WebAssembly to run TQLS in a browser
- Using GraphQL to run TQLS on a server
- Using Rust programming language to implement our design
- Generating type-safe CST API from Ungrammar description
- Using interactive playground to test and debug TQLS

References

[1]. M. Fowler and R. Parsons, Domain-Specific Languages. Boston, MA: Addison-Wesley Educational, 2009.
[2]. “TALISMAN – tracking and learning insights from social media analysis,” Ispras.ru. [Online]. Available: https://talisman.ispras.ru/. [Accessed: 13-May-2024].
[3]. M. Karpov, “megaparsec: Industrial-strength monadic parser combinator library.” [Online]. Available: https://github.com/mrkkrp/megaparsec. [Accessed: 31-Mar-2024].
[4]. J. Barretto, “chumsky: Write expressive, high-performance parsers with ease.” [Online]. Available: https://github.com/zesterer/chumsky. [Accessed: 31-Mar-2024].
[5]. “Language Server Protocol Specification,” Github.io. [Online]. Available: https://microsoft.github.io/language-serverprotocol/specifications/lsp/3.17/specification/. [Accessed: 31-Mar-2024].
[6]. C. Donnelly and R. Stallman, Bison. Samurai Media Limited, 2015.
[7]. A. Kladov, “Challenging LR parsing,” rustanalyzer, 16-Sep-2020. [Online]. Available: https://rustanalyzer.github.io/blog/2020/09/16/challeging-LR-parsing.html. [Accessed: 12-May-2024].
[8]. T. Parr, Definitive ANTLR 4 Reference, 2nd ed. Raleigh, NC: Pragmatic Programmers, 2013
[9]. S. Efftinge and M. Spoenemann, “Xtext - Language Engineering Made Easy!,” Eclipse.dev. [Online]. Available: https://eclipse.dev/Xtext/. [Accessed: 12-May-2024].
[10]. “GraphQL,” Graphql.org, 26-Oct-2021. [Online]. Available: https://spec.graphql.org/October2021/. [Accessed: 02-Jul-2024].
[11]. “rust-analyzer: A Rust compiler front-end for IDEs.” [Online]. Available: https://github.com/rust-lang/rust-analyzer. [Accessed: 31-Mar-2024].
[12]. “Introducing ungrammar,” Github.io, 24-Oct-2020. [Online]. Available: https://rust-analyzer.github.io//blog/2020/10/24/introducing-ungrammar.html. [Accessed: 02-Jul-2024].

Информация об авторах / Information about authors

Артём Дмитриевич БЕЛОУСОВ – старший лаборант, студент Высшей школы экономики. Сфера научных интересов: конструирование компиляторов и облачные вычисления.

Artyom Dmitrievich BELOUSOV – senior laboratory assistant, student of Higher School of Economics. Research interests: compiler design and cloud computing.

DOI: 10.15514/ISPRAS-2024-36(2)-3

Exploring the Taxonomy of Commits in Cyber-Physical Systems for Enhanced Error Fixes Investigation

N.A. Starovoytov, ORCID: 0009-0007-0242-0198 <*nikstarall@gmail.com*>
S.M. Staroletov, ORCID: 0000-0001-5183-9736 <*serg_soft@mail.ru*>

Polzunov Altai State Technical University,
46, prospect Lenina, Barnaul, Altai region, 656038, Russia.

Abstract. Cyber-physical systems are a symbiosis of multi-level control systems that take into account the physical aspects of the functioning of target objects. Errors in such systems can be associated both with incorrect organization of the code and operation of the hardware, as well as with an incorrect understanding of physical laws and their numerical approximation. Continuing our previous work, we apply technologies for analyzing commits in git repositories of some well-known cyber-physical systems, followed by classification of messages from developers. As a result, we discuss the identified strong keywords and generalized fix messages that can reveal the main classes of bugs in these projects. The results of the work can be used in training and consulting on errors and vulnerabilities in complex systems.

Keywords: clustering; fixing commits; errors classification; cyber-physical systems.

For citation: Starovoytov N.A., Staroletov S.M. Exploring the Taxonomy of Commits in Cyber-Physical Systems for Enhanced Error Fixes Investigation. Trudy ISP RAN/Proc. ISP RAS, vol 36, issue 2, 2024., pp. 33-46. DOI: 10.15514/ISPRAS-2024-36(2)-3.

Классификация коммитов в репозиториях киберфизических систем для исследования исправлений ошибок в них

Н.А. Старовойтов, ORCID: 0009-0007-0242-0198 <*nikstarall@gmail.com*>
С.М. Старолетов, ORCID: 0000-0001-5183-9736 <*serg_soft@mail.ru*>

АлтГТУ им. И.И. Ползунова,
Россия, 656038, Алтайский край, г. Барнаул, пр. Ленина, 46.

Аннотация. Киберфизические системы представляют собой симбиоз многоуровневых систем управления и учитывают физические аспекты функционирования целевых объектов. Ошибки в таких системах могут быть связаны как с неправильной организацией кода и работой аппаратных средств, так и с неверным пониманием физических законов и их численной аппроксимацией. Продолжая предыдущую работу, мы применяем технологии автоматизированного анализа коммитов в git-репозиториях некоторых известных киберфизических систем с последующей классификацией собранных сообщений о фиксации изменений, написанных разработчиками таких систем. В работе мы обсуждаем выявленные сильные ключевые слова и обобщенные сообщения об исправлениях, которые способны показать основные классы ошибок в этих проектах. Результаты исследования могут быть использованы при обучении и консультировании по ошибкам и уязвимостям в сложных системах.

Ключевые слова: кластеризация; исправляющие коммиты; классификация ошибок; киберфизические системы.

Для цитирования: Старовойтов Н.А., Старолетов С.М. Классификация коммитов в репозиториях киберфизических систем для исследования исправлений ошибок в них. *Труды ИСП РАН*, том 36, вып. 2, 2024 г., стр. 33-46 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–3.

1. Introduction

Modern software development environments are crucial for improving the efficiency and quality of software development processes. One such valuable tool is the git version control system [1], which is widely used among developers and provides a detailed change history in its repositories. Each change to the code is documented with a message written in natural language by the developer, making it easier to track the evolution of the software.

This work seeks to leverage data analysis methods to identify common types of errors in the source code of software used in cyber-physical systems. By analyzing the messages in git repositories, automated methods can be used to detect patterns of errors fixes for the code. This approach can help developers identify recurring issues and take preventive measures to improve software quality.

By conducting such analysis, software development teams can streamline their processes, reduce the occurrence of errors, and enhance the overall quality of the software being developed. By understanding the common types of errors and their main causes, developers can implement targeted solutions to address these issues and prevent them from occurring in future projects to ensure that software for cyber-physical systems is robust and reliable.

The rest of the paper has the following structure.

In Section 2, we discuss the background on commits to version control systems and cyber-physical systems. Section 3 is about the related works on errors detection in cyber-physical systems. Section 4 is devoted to the implementation plan and solving corresponding issues. In Section 5, we discuss the results we got by running our software to analyze some known repositories of cyber-physical systems. In Conclusion, we point out the generalization of found errors types.

The present paper is an extension of the report presented at SYRCoSE Software Engineering Colloquium 2024 in Stavropol.

2. Background

2.1 The git version control system

A version control system is a system that helps manage software development. It tracks changes, provides multiple version control of files, and allows multiple developers to collaborate [2].

The git version control system was created by Linus Torvalds to manage the development of the Linux kernel. Projects in the git system are called repositories.

A git repository contains a collection of files and folders associated with a project, along with a history of changes to each file. A file history is a list of changes at a specific moment called commits. They can be organized into multiple development lines called branches. Because git is a distributed version control system, anyone with a copy of the repository can access the entire project's codebase and its history. Thus, by analyzing the repository one can track the progress of project development.

Each commit contains the following tracking information [3]: (1) a snapshot of the current state of the files in the repository at the time the changes were committed, a commit stores all the changes that have been made to the files at the time of the commit; (2) author of the commit; (3) the date the commit was created; (4) a commit message written by the developer in natural language that describes the essence of the changes made in this commit; (5) hash sum; and (6) parent commits.

2.2 Commit classification

The three most common reasons for making changes to the code by introducing a commit, are [4-5]:

1) Adding new functionality: this refers to the process of introducing new features to the software application. It could involve creating new modules, implementing new algorithms, or integrating third-party libraries.
2) Improving code quality: this reason includes making changes to the code to enhance its maintainability, readability, and efficiency. It could involve refactoring code to follow coding standards, optimizing algorithms or improving documentation.
3) Fixing bugs: they involve identifying and resolving issues in the code that affect the functionality of the software application.

While bug fixes and adding new functionality are closely related reasons for code changes, it is essential to accurately classify commits in a version control system to distinguish between the two categories. To address this, some research focused on identifying key features that differentiate bug fixes from adding new functionality in commits. By analyzing the patterns, code changes, and contextual information associated with each type of commit, a commit classifier was made [6]. In our work we use almost the same ideas.

2.3 Cyber-physical systems and errors in it

Edward Lee's works on modeling cyber-physical systems [7, 8] provide in-depth exploration of the fundamental concepts underlying these complex systems. Lee points out the importance of a precise definition of cyber-physical systems, highlighting their evolution from the earlier cybernetic systems studied in control theory. One key aspect that sets modern cyber-physical systems apart is their reliance on diverse sensor data, which necessitates a distributed approach to operation. These systems not only interact with their environment but also have direct impacts on human lives, underscoring the need for accurate modeling to ensure their safe and effective functioning.

Raj Rajkumar emphasizes the interconnected nature of cyber-physical systems [9], notes how these systems combine physical processes with communication networks and computing capabilities. Helen Gill expands the definition of cyber-physical systems to include human factors, taking into account the role of human operators in these systems [10]. Janos Sztipanovits highlighting the interdisciplinary nature of cyber-physical systems, pointing how these systems bring together expertise from various fields such as computer science, engineering, and mathematics [11].

Errors in cyber-physical systems can be categorized into various types [12] such as design errors, implementation errors, communication errors, timing errors, and environmental errors. These errors can result in safety, system failures, performance, and security vulnerabilities.

Addressing errors in cyber-physical systems presents numerous challenges [13] due to the intricate nature of system interactions, real-time constraints, resource limitations, and the necessity for interdisciplinary expertise. To enhance error detection and mitigation strategies, researchers are investigating advancements in formal methods, model-based design, runtime monitoring, anomaly detection, and machine learning techniques [14].

2.4 Examples of cyber-physical systems to analyze

In this subsection, we describe some notable projects from our past experience and from the list of projects on the topic "cyber-physical systems" on GitHub, which we chose to analyze error fixes in their repositories.

The Ardupilot project [15] can be considered as a cyber-physical system. Ardupilot is an open-source autopilot software suite that enables autonomous control of drones, unmanned aerial vehicles (UAVs), and other robotic systems. It integrates physical components (such as sensors, actuators, and communication modules, see the example of architectural modeling in [16] with computational elements (software algorithms, control logic) to enable real-time monitoring, control, and navigation of the vehicle.

The Scada-LTS project [17] can also be understood as a cyber-physical system. SCADA (Supervisory Control and Data Acquisition) systems are used to monitor and control industrial processes, infrastructure systems, and other complex systems that involve the interaction between physical components and digital control systems. The Scada-LTS project is an open-source SCADA software system provides monitoring, control, monitoring, and data acquisition capabilities for various industrial applications.

The Modelica project [18-19] is related to the cyber-physical system topic. Modelica is an open-source, object-oriented modeling language used to model complex cyber-physical systems that involve the interaction between physical components and digital control systems. In a cyber-physical system context, Modelica can be used to create models that represent the behavior of physical components such as mechanical systems, electrical systems, thermal systems, and more. These physical component models can then be integrated with control algorithms and other software components to create a comprehensive model of the entire system.

The KeYmaeraX project [20-21] relates to the promising topic of proving the correctness of cyber-physical systems. KeYmaeraX is a theorem prover for hybrid systems, which are systems that exhibit both continuous dynamics (physical processes) and discrete dynamics (digital control algorithms). Cyber-physical systems often fall into the category of hybrid systems. KeYmaeraX supports the verification of safety-critical properties, such as collision avoidance, stability guarantees, reachability analysis, and dynamic logic specifications [22-24], which are essential for ensuring the reliable operation of cyber-physical systems.

3. Related work

To search for errors and vulnerabilities in cyber-physical systems, both static and dynamic methods are applicable. Since software for cyber-physical systems is a set of executable programs (possibly running on different nodes), general methods of analysis, testing and verification specific to distributed software are applicable to it. For such systems, monitoring is preferable, when the system is running and its current state can be obtained and compared with the expected one. The work [25] describes a review of this kind of monitoring for given requirements in the form of specifications. Specific signal analysis methods are applicable for different classes of such systems [26]. For real-time systems, this kind of monitoring can check parameters such as input rate, scheduling delay, and processing time [27]. For systems dependent on network interactions, methods for analyzing network vulnerabilities [28] are applicable.

It has long been discussed that static code analysis techniques are applicable to such systems [29]. Although for special domains, special approaches must be used to generate benchmarks [30].

As of theoretical research, first of all, methods for constructing behavioral models of such systems and testing them [31] as well as logical calculus systems for modeling vulnerabilities [32] are known. Such models are decoupled from real code, which entails the use of abstraction and a potential loss of adequacy, but they can potentially test situations that are difficult to reproduce.

Empirical work by Tan et al. [33] focuses on conducting a comprehensive analysis of bugs in open-source software, with a specific emphasis on the Linux kernel. In the case of Linux, bugs are categorized based on their respective subsystems, such as core, driver, network, FS, arch, or other. The study utilizes BugZilla's message text for various open-source components, employing vectorization techniques for automatic bug classification. In a similar vein, Xiao et al. [34] proceed with a detailed examination of 5741 Linux kernel bug reports. Their analysis delves into bug descriptions, comments, and attached files from the Linux kernel bug tracker, BugZilla. The bugs are further classified as fast-reproducible (Bohrbug), difficult-to-reproduce (Mandelbug), or context-dependent. Additionally, the researchers define specific categories to which bugs can be attributed, such as memory errors or unfreed resources, based on their contextual dependencies. In the present work, we are going to do such kind of analysis. By focusing on cyber-physical systems rather than Linux kernels, we can uncover specific issues and vulnerabilities that are relevant to this

particular domain. Utilizing git repositories as the source for messages provides a wealth of data that can offer insight into the development and maintenance of these systems. By differentiating between fixing commit messages, we can pinpoint common errors and identify patterns that may indicate underlying issues within the codebase.

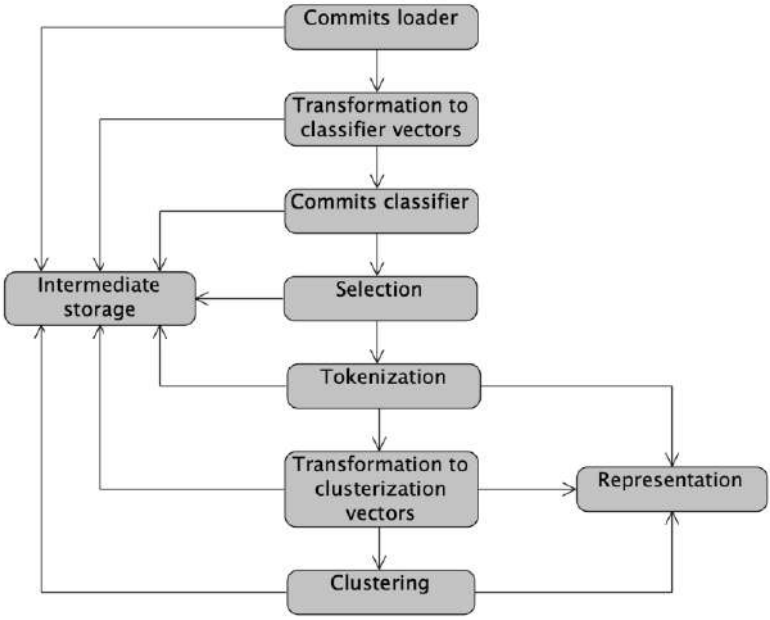


Fig. 1. Architecture of our solution

4. On the implementation

4.1 The structure of the solution

The internal flow and main modules of our solution are presented in Fig. 1. The ideas to implement the approach are presented in our previous paper [35]. To repeat, this is getting the text of commits, identifying commits that commit, representing its text in vector form using a “bag of words”, working with increasing the role of significant words, and further clustering such vectors into generalizing centroids with the ability to restore close messages from commits for each of the found centroids. Les us just describe the logical modules.

The **Intermediate storage** is a module necessary to be able to interrupt the progress of the program and replace other modules without completely losing the result of the work; it can be presented in the form of a small database management system.

The **Commits loader** loads commit information from the git system. This module is necessary because calculating file changes can be a costly operation, and preloading information allows it to be done only once.

The **Transformation to classifier vectors** module obtains commits in a vector representation. For conversion, both data from the commit information in natural language and in a programming language are used. Priority in classification is given to control flow graph representation of the commit diff. The vector data is used only for classification and is not used further in commit clustering.

The **Commit classifier module** determines whether a certain commit is a bug fix.

The **Selection** module makes a selection from the general pool of commits that will be needed for a specific study. For example, the comment belongs to a specific domain and time period.

The **Tokenization** module converts a commit message to a list of keywords. This process involves removing technical information, converting words to a basic form, and removing unimportant words.

The **Transformation to clusterization vectors** module converts a list of keywords to numeric vectors, and also generates a dictionary for the reverse arrangement of vectors.

The **Clustering module** defines a certain cluster for a specific commit or assigns it to an outlier (does not assign it to any of the clusters). It works with the **Distance calculation** module which calculates the matrix of distances between all vectors.

The **Representation** module collects data from different modules, organizes them and presents them in a user-friendly format.

4.2 Covered issues

In this section, we discuss solving the difficulties that followed attempts to directly apply Linux kernel repository analysis approaches [35] to new cyber-physical system repositories.

During the analysis, it was noticed that the selected repositories of cyber-physical systems lack in commit messages a detailed description of the actions taken by developers. Therefore, we have to rely solely on headlines as the source of data for our analysis. Incomplete and uninformative commit messages make it difficult to understand development history and effectively track down the causes of problems. Unfortunately, in the repositories studied, such commits predominate, which complicates the analysis.

A significant problem also arises from the high IDF values observed in dictionaries. This reflects a lack of token diversity that prevents key aspects from being isolated in the codebase. As a consequence, inflated values lead to a number of problems that must be overcome to achieve good storage analysis.

As a solution to the problems, LSA transformations had to be introduced. LSA is a machine learning technique used to identify semantic patterns in large text data sets, making it an effective tool for analyzing changes in domain-specific code repositories.

It was also decided to change the clustering algorithm. DBSCAN offers significant improvements to the commit analysis process. It detects clusters of different densities and shapes, making it easier to adapt to different projects.

Analyzing each repository with a small number of commits is a unique task in selecting clustering parameters (Table 1), and requires careful manual tuning. Manual selection is critical to achieving optimal results. For example, increasing the “LSA units” parameter can help in cases where the number of clusters is not enough or too many.

Table 1. Empirically selected parameters for clustering and LSA transformation

	Scada	Modelica	KeYmaeraX	ardupilot
Min cluster size	5	5	10	15
Eps	0.23	0.23	0.38	0.45
LSA units	100	100	200	300

In cyber-physical systems repositories, the general lack of a sufficiently large number of commits is a major obstacle because it limits the amount of data available for analysis. This limitation makes it difficult to effectively identify patterns. Low commit rates may be due to factors such as small development teams, infrequent updates, or rigorous testing and review processes before committing changes. Overcoming these challenges requires taking a closer look at available data and exploring alternative analytical approaches to extract meaningful information from a limited data set.

Differences in commit message conventions create serious problems for the classifier (deciding whether a fix or a new feature has been committed) when applied across different repositories. As a result, classifiers trained in one repository may struggle to perform effectively in others, resulting in

reduced relevance in identifying patches and clustered data. In this set, a classifier trained on the Linux kernel repository was used, so it was much easier for it to find training data.

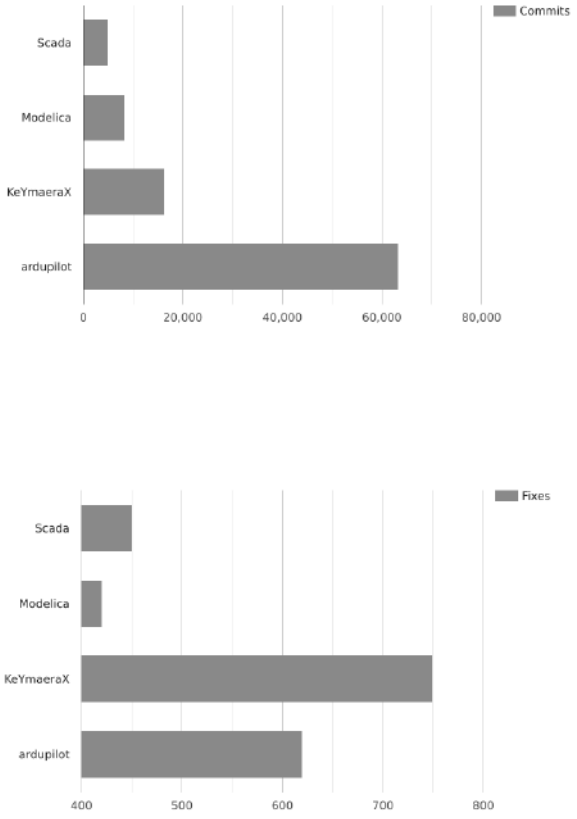


Fig.2. The number of commits (top) vs the number of fixes (bottom) in the selected repositories

The accompanying Fig.2 illustrates an interesting observation: the number of identified fixes does not depend on the total number of commits. This highlights the difficulty of generalizing classifiers and the importance of considering storage-specific nuances to improve classification accuracy.

To mitigate the impact of problems associated with abstract commit descriptions and low commit frequency, several strategies are proposed. These include analyzing relationships between comments, tracking code changes over time, identifying patterns based on sequences of commits, etc. Additionally, using meta-information such as commit author, timestamp, and associated files can enhance the analysis capabilities. Combining data from different sources gives us a more complete understanding of storage dynamics and helps to make more informed decisions.

Every repository, be it Scada, Modelica, KeYmaera X or Ardupilot, has issues that affect the efficient identification of commit clusters (Table 2). These problems include weak commit descriptions, low numbers of commits, difficulties in classifying fixes, and low relevance within the context of the repository's domain. For example, automatic analysis of the Modelica repository is almost impossible with the proposed methods, since each of the proposed problems is present in it.

Interestingly, repositories with fewer inherent problems require more stringent parameter specifications to achieve optimal results for them.

Table 2. Main issues for clustering in selected repositories

	Scada	Modelica	KeYmaeraX	ardupilot
Weak description	+	+	+	+
Low commits count	+	+		
Wrong classification of fixing commits		+		+
Low relevance	+	+	+	

5. Results of analysis of repositories with some known projects of cyber-physical systems

In this section, we show the results of the analysis of the git repositories of the specified projects by our software. First, we demonstrate the generalized vectors found, with the words there arranged in descending order of weight using tf-idf. Next, for each vector, we show examples of the most relevant messages from commits that match this vector.

5.1 The Ardupilot project

Vector #1: ['apvehicle', 'compiler', 'float', 'override', 'keyword', 'old', 'airspeed', 'double', 'gc', 'initialisation', 'serial', 'avoid', 'wrapper', 'older', 'return']

```
AP_PID: compiler warnings: apply is_zero(float)
APMrover2: compiler warnings: apply is_zero(float) or is_equal(float)
Plane: compiler warnings: apply is_zero(float) or is_equal(float)
AP_Compass: compiler warnings: apply is_zero(float) or is_equal(float)
AntennaTracker: compiler warnings: apply is_zero(float) or is_equal(float)
```

This vector represents a fix for the compiler warning when working with floating-point values.

Vector #2: ['hwdef', 'binding', 'copter', 'register', 'match', 'changes', 'update', 'sub', 'optional', 'recent', 'upstream', 'manual', 'i2c', 'avoid', 'lua']

```
AP_HAL_ChibiOS: hwdef add support for Networking
hwdef: add hwdef for SDMODELH7V1
AP_HAL_ChibiOS: update truenav hwdef
AP_HAL_ChibiOS: hwdef for Flywoo F405 Pro
AP_HAL: move defaulting of HAL_DSHOT_ALARM into hwdef
```

This vector represents an improvement to the hardware definition for hardware abstraction components.

Vector #3: ['hil', 'initial', 'wrapper', 'implementation', 'roll', 'tailsitter', 'pitch', 'eliminate', 'transition', 'ap', 'attitude', 'call', 'vtol', 'timing', 'creation']

```
ACM: fixed HIL build again
ArduPlane: remove HIL support
AP_Compass: remove HIL support
Blimp: remove HIL support
GCS_MAVLink: remove HIL support
```

This vector deals with support for the Hardware-in-the-loop simulator, which was gradually removed from the project.

Vector #4: ['define', 'separate', 'send', 'patch', 'stability', 'simple', 'hold', 'alt', 'ability', 'able', 'absolute', 'abstraction', 'ac', 'accel', 'acceleration']

```
AP_RCProtocol: add separate define for AP_RCPROTOCOL_PPMSUM_ENABLED
AP_RCProtocol: add separate define for AP_RCPROTOCOL_ST24_ENABLED
AP_RCProtocol: add separate define for AP_RCPROTOCOL_DSM_ENABLED
AP_RCProtocol: add separate define for AP_RCPROTOCOL_SUMD_ENABLED
AP_RCProtocol: add separate define for AP_RCPROTOCOL_IBUS_ENABLED
```

This vector deals with define macros in code.

Vector #5: ['fence', 'without', 'register', 'attitude', 'pointer', 'must', 'collective', 'avoid', 'quad', 'horizontal', 'next', 'tradheli', 'heli', 'term', 'structure']

```
AC_Fence: add polygon fence check to check_destination_within_fence
AC_Avoid: add support for stopping at polygon fence
AC_Fence: add support for polygon fences
AP_OSD: Add fence indicator panel
Rover: add fence support
```

This vector deals with fixes for obstacle avoidance functions.

Vector #6: ['location', 'adjust', 'vector3f', 'require', 'accepts', 'send', 'home', 'packet', 'specify', 'original', 'unify', 'circle', 'mount', 'support', 'usage']

```
autotest: fixed buildlogs location for *.BIN
AP_Math: move line_path_proportion to Location
ArduPlane: use past_interval_finish_line and line_path_proportion from Location
Tweaks to fix Loiter
Changed save location to int32
added some filtering and smoothing
APMrover2: use past_interval_finish_line and line_path_proportion from Location
```

This vector deals with fixes for location objects.

Vector #7: ['scheduler', 'enum', 'old', 'ability', 'able', 'absolute', 'abstraction', 'ac', 'accel', 'acceleration', 'accelerometer', 'accept', 'accepts', 'access', 'accessor']

```
Rover: windvane update called from scheduler at 20hz
AP_NavEKF: Scheduler Improvements
Plane: remove update_events scheduler shim
AP_HAL_AVR: Scheduler extensions implemented
Copter: remove shims used in scheduler
```

This vector deals with fixes for the scheduler.

Vector #8: ['macro', 'rewrite', 'frontend', 'flag', 'ability', 'able', 'absolute', 'abstraction', 'ac', 'accel', 'acceleration', 'accelerometer', 'accept', 'accepts', 'access']

```
all: use CLASS_NO_COPY() macro
Copter: Obey RANGEFINDER_ENABLED, AUTOTUNE_ENABLED and AC_TERRAIN build macros
AP_HAL_SMACCM: fix to goofed PPM_MAX_CHANNELS macro
HAL_ChibiOS: use EXPECT_DELAY() macro
AP_InertialSensor: use EXPECT_DELAY() macro
```

This vector deals with various code fixes in macros (specific to C/C++ projects).

Vector #9: ['script', 'reset', 'last', 'install', 'readme', 'folder', 'dronecan', 'style', 'multiple', 'optflow', 'byte', 'iofirmware', 'lua', 'jsbsim', 'enumeration']

```
AP_HAL_ChibiOS: fix script for HerePro
AP_HAL_F4Light: fixed some support scripts
AP_Scripting: add arming check for failed scripts
HAL_ChibiOS: moved to generated loader script
AP_Scripting: add checksum of running and loaded scripts with arming check
```

This vector deals with fixes for built-in scripts.

Vector #10: ['quadplane', 'adjust', 'ability', 'able', 'absolute', 'abstraction', 'ac', 'accel', 'acceleration', 'accelerometer', 'accept', 'accepts', 'access', 'accessor', 'accessors']

```
Plane: Quadplane: use uint16_t for output_motor_mask
Plane: added QTUN logging for quadplane
AC_WPNav: converted to use AP_AHRS_View
Plane: Quadplane remove THR_MIN_PWM and THR_MAX_PWM
Plane: allow for NAV_LOITER_UNLIM and NAV_LOITER_TIME in quadplane
```

This vector deals with fixes for working with QuadPlane, which is a combined fixed wing and MultiCopter aircraft.

5.2 The Scada-LTS project

Vector #1 ['settimeout', 'state', 'commonjs', 'pointpropertiesapi', 'eventtextrenderer', 'translate', 'interface', 'messagesms', 'localization', 'validate', 'copy', 'plcalarmsdao', 'messages', 'adapter', 'miscdwrdolongpoll']

```
#2111 Fixed using setTimeout in common.js - added multi thread tests: MiscDwrDoLongPollAlarmsMultiThreadTest, MiscDwrDoLongPollMultiThreadTest; corrected MiscDwr.doLongPoll, without copy state
#2111 Fixed using setTimeout in common.js - use setInterval
#2111 Fixed using setTimeout in common.js 3 - setTimeout functionality ported to java
#2111 Fixed using setTimeout in common.js - validate uiPerformance, added 'Very high' option (1000ms)
#2111 Fixed using setTimeout in common.js - corrected refresh points with statistics, for request that run longer than the Interval Time
```

This vector is a fix for the timer function - executing code by timeout in JS.

Vector #2: ['patch', 'vue', 'views', 'tests', 'ability', 'able', 'abstract', 'abstractbeforeafterworkitem', 'accept', 'access', 'accetable', 'acknowledge', 'acl', 'action', 'active']

```
patch removed newline
patch MangoTextContent.java
Refactor Vue Unit Tests #1533
#1641 ADD [NotFinished] VirtualDataPoint Vue
#1894 Views caching - corrected
```

This vector represents patches for views (that is, representing the states of objects on the screen).

Vector #3: ['correction', 'commit', 'display', 'partial', 'isalive2', 'viewdao', 'clean', 'eventdetectorapi', 'not', 'eventlist', 'component', 'number', 'ability', 'able', 'abstract']

```
Correction of commit number display. #1502
#2051 Visual corrections (component shrink)
EventDetectorAPI corrections #1532
#2051 Add corrections to the IsAlive2
SLTS-40 Add correction to ViewDAO
```

This vector represents various corrections associated with the display of information and events in the system.

Vector #4: ['slts13', 'testdao', 'annotation', 'cleanup', 'scriptdao', 'direct', 'rewrite', 'navigation', 'link', 'controller', 'ability', 'able', 'abstract', 'abstractbeforeafterworkitem', 'accept']

```
SLTS-13 Added FlexProjectRowMapper and remove @SuppressWarning annotation
SLTS-13 move expectedException to TestDAO
SLTS-13 Rewrite ScriptDAO
SLTS-13 controller's clean-up
SLTS-132 Implemented direct link navigation.
```

This vector represents fixes for DAO (data access object).

Vector #5: ['log4j', 'pointdetails', 'slts34', 'logger', 'ability', 'able', 'abstract', 'abstract before after workitem', 'accept', 'access', 'accetable', 'acknowledge', 'acl', 'action', 'active']

```
#1982 - log4j update
#1982 - log4j update
SLTS-97 Add logger for log4j. Correct PointValueService.
WORKSAVE New PointDetails ideas
SLTS-34 Add DataSourceServiceTest
```

This vector represents logging fixes using log4j (log for Java).

5.3 The Modelica project

Vector #1: ['proper', 'word', 'format', 'use', 'number', 'leakage', 'accordingly', 'individual', 'usersguide', 'magnetic', 'electrical', 'work', 'ha', 'implementation', '09']

```
Use proper number formats
Proper word
Use proper number formats
Proper word
SymmetricPolyphaseWinding: moved individual leakage from magnetic to electrical
implementation: it works!
UsersGuide has to be adapted accordingly.
```

This vector represents fixes for formatting messages with the correct number format.

Vector #2: ['unify', 'time', 'ccr', 'event', 'enhancement', 'cleanup', 'drive', 'controller', 'some', 'powerconverters', 'partial', 'interface']

```
refs #1627: Fix detection of (scaled) time events
refs #1627: Fix detection of scaled time events
CCR: corrected error in setState_ps
```

This vector represents fixes for working with time events.

Vector #3: ['electricaldigital', 'correction', 'elementary', 'grid', 'fit', 'mechanicsmultibody', 'layout', 'better', 'info', 'svn', 'revision', 'function']

```
#799 solved for Electrical.Digital
errors fixed in Electrical.Digital
errors fixed in Electrical.Digital
corrections according to #994 in Electrical.Spice3.Internal.Fet
Correction of info of function Matrices.realSchur
```

This vector represents bug fixes in the Electrical.Digital component (this library contains packages to model digital electronic systems based on combinational and sequential logic).

Vector #4: ['due', 'picture', 'complexblocks', 'modification', 'docu', 'referenceair', 'referencemoistair', 'mass', 'energy', 'balance', 'some', 'static']

```
Comments added due to #1475
modifications due to ComplexBlocks
due to #407 bugs 2, 3, 4, 5, 7 fixed (docu, pictures)
Some changes due to renaming of ReferenceMoistAir and ReferenceAir.
medium.preferredMediumStates is now false if both mass and energy balances are
static
Attempt to fix issue #3236
```

This vector represents error fixes for complex blocks with images.

Vector #5: ['dynamicselect', 'boolean', 'comparison', 'real', 'individual', 'stray', 'implementation', '09', '150', '2dtable', '3rdparty', '64bit', 'abbreviation']

```
Avoid Boolean comparison with Real in DynamicSelect (#2862)
Avoid Boolean comparison with Real in DynamicSelect (#2879)
corrected implementation of individual stray permeances
```

This vector represents fixes for proper comparison with real (floating-point) values.

5.4 The KeYmaeraX project

Vector #1: ['implement', 'syntactic', 'derivative', 'index', 'skeleton', 'skolemization', 'counting', 'magic', 'less', '20150824', '20160308', '20160601', '20160802', '20160816', '2sided']

```
implement GetPathAll
implement the CreateProblemRequest
implement BranchRoot
implement skolemization
implement more of skeleton
```

This vector represents the addition of various functionality to the project in the form of large commits, for example, the implementation of skolemization (reduction to Skolem normal form is an approach for removing existential quantifiers from formal logic statements).

Vector #2: ['package', 'private', 'change', 'moves', 'datastructures', 'firstintegrals', 'fol', 'replaceall', 'tooltips', 'dwplus', 'compilability', 'link', 'thanks', 'cert', 'sos']

```
packages
Change package
Change package for compilability.
merge tons of code from kaisar package to experiments package, get bot working
again
Move strategic AxiomIndex to package btactics
```

This vector represents fixes for rebuilding packages in the logical organization of the architecture.

Vector #3: ['solve', 'axiominfo', 'nilpotent', 'search', 'feedback', 'anyarg', 'anything', 'axiombase', 'output', 'rescue', 'diffsolve', 'theme', 'choice', 'consistently', 'speedup']

```
Nilpotent solve (preliminary)
Nilpotent solve speedup
Re-unification to solve p(.)~>.>=0 , p(.)~>2>=0 issues as in
[x':=2]x'>=0<->2>=0 against [':=]
Nilpotent solve: dW only when provable
key/recusor in AxiomInfo
```

This vector represents fixes for Nilpotent solve (for solving algebraic structures).

Vector #4: ['unification', 'derive', 'bidirectional', 'axindex', 'imply', 'monomial', 'projection', 'init', 'sign', 'bit', 'variation', 'dot', 'dbx', 'match', 'dotterm']

```
Colored-dots unification
Improve unification a bit further
Unification support for projection
Unification match: 0-indexed colored dots
DiffHelper derive fix (sign error)
```

This vector represents fixes for the unification algorithms [36] in the theorem prover.

6. Discussion and Conclusion

The results of the clusterer can be considered successful only for the Ardupilot project. For the rest, there are problems with obtaining both the required number of arbitrarily representative clusters, and with the quality of the found vectors, so that they actually represent fixes of repeating types of errors. We see the main problems here in the organization of development of the selected projects, when developers write uninformative messages about commits. In addition, task management suffers: code fixes are committed to solve large problems at once, rather than small changes with a clear justification. The final problem we see is the classification of commits into adding functionality and fixes: the classifier is based on the code as well as messages of Linux kernel commits and for other repositories, as it turned out, it does not work entirely correctly.

As for the processed commits, in the Ardupilot project the main errors were specific to the software domain for unmanned vehicles, such as issues of location processing, obstacle avoidance, abstraction from hardware and scheduling. For SCADA systems, errors specific to Java applications and MVC architecture were found, which is not surprising, because such projects visualize monitoring data of cyber-physical systems. For the Modelica system, the Electrical.digital subsystem was found, which is prone to errors, as well as other issues there are related to the conversion of numeric floating-point values. Finally, in the theorem prover for cyber-physical systems KeYmaeraX (with not a good organization of commits that does not reveal the full complexity of development), the main logical subsystems for proving cyber-physical models in which there were many changes in the code were found: skolemization, nilpotent solve and unification.

The final advantages of our solution are the ability to easily evaluate what is being done in a given repository (with a sufficiently large number of well-organized commits) and what errors were corrected in order to learn from examples of the development of this kind of systems with increased reliability requirements.

References

[1]. git. Available at: https://git-scm.com, accessed Jun. 24, 2024.
[2]. Otte S. Version control systems, 2009. Available at: https://api.semanticscholar.org/CorpusID:7541013, accessed Jun. 24, 2024.
[3]. Chacon S., Straub B. Pro git. Springer Nature, 2014.
[4]. Herzig K, Just S., Zeller A. It's not a bug, it's a feature: how misclassification impacts bug prediction. In Proc. 2013 35th international conference on software engineering (ICSE). IEEE, 2013, pp.392–401.
[5]. Kagdi H., Collard M. L., Maletic J. I. A survey and taxonomy of approaches for mining software repositories in the context of software evolution. Journal of software maintenance and evolution: Research and practice, vol. 19, no. 2, pp. 77–131, 2007.
[6]. Tian Y., Lawall J., Lo D. Identifying Linux bug fixing patches. In Proc. 2012 34th international conference on software engineering (ICSE). IEEE, 2012, pp. 386–396.
[7]. Lee, E. A. The past, present and future of cyber-physical systems: A focus on models. Sensors, vol. 15, no. 3, pp. 4837–4869, 2015.
[8]. Lee, E. A. Plato and the nerd: The creative partnership of humans and technology. MIT Press, 2017.
[9]. Rajkumar R., De Niz D., Klein M., Cyber-physical systems. Addison-Wesley Professional, 2016.
[10]. Gill H. From vision to reality: cyber-physical systems. In Proc. HCSS national workshop on new research directions for high confidence transportation CPS: automotive, aviation, and rail. Austin USA, 2008, pp. 1–29.
[11]. Sztipanovits J., Koutsoukos X., Karsai G., Kottenstette N., Antsaklis P., Gupta V., Goodwine B., Baras J., Wang S. Toward a science of cyber–physical system integration. Proceedings of the IEEE, vol. 100, no. 1, pp. 29–44, 2011.
[12]. Siddesh G.M., Deka G.C., Srinivasa K.G., Patnaik L.M., Cyber-physical systems: a computational perspective. CRC Press, 2015.
[13]. Rajkumar R., Lee I., Sha L., Stankovic J. Cyber-physical systems: the next computing revolution. In Proc. of the 47th design automation conference, 2010, pp. 731–736.
[14]. Luo Y., Xiao Y., Cheng L., Peng G., Yao D., Deep learning-based anomaly detection in cyber-physical systems: Progress and opportunities. ACM Computing Surveys (CSUR), vol. 54, no. 5, pp. 1–36, 2021.
[15]. ArduPilot Project. Available at: https://github.com/ArduPilot/ardupilot, accessed Jun. 24, 2024.
[16]. Staroletov S. Architectural software-hardware co-modeling a real-world cyber-physical system: Arduino-based ardupilot case. In Proc. 2021 30th Conference of Open Innovations Association. IEEE, 2021, pp. 267–278.
[17]. Scada-LTS. Available at: https://github.com/SCADA-LTS/Scada-LTS, accessed Jun. 24, 2024.
[18]. Modelica Standard Library. Available at: https://github.com/modelica/ModelicaStandardLibrary, accessed Jun. 24, 2024.
[19]. Fritzson P. Principles of object-oriented modeling and simulation with Modelica 3.3: a cyber-physical approach. John Wiley & Sons, 2014.
[20]. KeYmaera X, theorem Prover for Hybrid Systems. Available at: https://github.com/LS-Lab/KeYmaeraX-release, accessed Jun. 24, 2024.
[21]. Fulton N, Mitsch S., Quesel J.-D., M. Volp, Platzer A, KeYmaera X: An axiomatic tactical theorem prover for hybrid systems. In Proc. Automated Deduction-CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings 25. Springer, 2015, pp. 527–538.
[22]. Quesel J.-D., Mitsch S., Loos S., Arechiga N., Platzer A. How to model and prove hybrid systems with KeYmaera: a tutorial on safety. International Journal on Software Tools for Technology Transfer, vol. 18, no. 1, pp. 67–91, 2016.
[23]. Staroletov S. Automatic proving of stability of the cyber-physical systems in the sense of Lyapunov with KeYmaera. In Proc. 2021 28th Conference of Open Innovations Association. IEEE, 2021, pp. 431–438.
[24]. Staroletov S., Schulte H., Baar T., Konyukhov I., Shilov N., Rozov A., Liakh T., Zyubin V. Modeling and verification using different notations for CPSs: The one-water-tank case study. In Proc. 2021 16th Conference on Computer Science and Intelligence Systems (FedCSIS). IEEE, 2021, pp. 485–488.

[25]. Bartocci E., Deshmukh J, Donze A., Fainekos G., Maler O., Nickovic D., Sankaranarayanan S. Specification-based monitoring of cyber-physical systems: a survey on theory, tools and applications. Lectures on Runtime Verification: Introductory and Advanced Topics, pp. 135–175, 2018.
[26]. Morgan J., O'Donnell G. E. Cyber physical process monitoring systems. Journal of Intelligent Manufacturing, vol. 29, no. 6, pp. 1317–1328, 2018.
[27]. Canizo M., Conde A., Charramendieta S., Minon R., Cid-Fuentes R. G., Onieva E., "Implementation of a large-scale platform for cyber-physical system real-time monitoring," IEEE Access, vol. 7, pp. 52 455–52 466, 2019.
[28]. Ashibani Y., Mahmoud Q. H. Cyber physical systems security: Analysis, challenges and solutions. Computers & Security, vol. 68, pp. 81–97, 2017.
[29]. Lee E. A. Cyber physical systems: Design challenges. In Proc. 2008 11th IEEE international symposium on object and component-oriented real-time distributed computing (ISORC). IEEE, 2008, pp. 363–369.
[30]. Eichler C., Wagemann P., Schroder-Preikschat W. Genee: A benchmark generator for static analysis tools of energy-constrained cyber-physical systems. In Proc. of the 2nd Workshop on Benchmarking Cyber-Physical Systems and Internet of Things, 2019, pp. 1–6.
[31]. Fabarisov T., Yusupova N., Ding K., Morozov A., Janschek K. Model-based stochastic error propagation analysis for cyber-physical systems. Acta Polytechnica Hungarica, vol. 17, no. 8, pp. 15–28, 2020.
[32]. Lanotte R., Merro M, Munteanu A., Vigano L. A formal approach to physics-based attacks in cyber-physical systems. ACM Transactions on Privacy and Security (TOPS), vol. 23, no. 1, pp. 1– 41, 2020.
[33]. Tan L., Liu C., Li Z., Wang X., Zhou Y., Zhai C., Bug characteristics in open source software. Empirical software engineering, vol. 19, pp. 1665–1705, 2014.
[34]. Xiao G., Zheng Z., Yin B., Trivedi K. S., Du X., Cai K.-Y. An empirical study of fault triggers in the Linux operating system: An evolutionary perspective. IEEE Transactions on Reliability, vol. 68, no. 4, pp. 1356–1383, 2019.
[35]. Staroletov S., Starovoytov N., Golovnev N. Analyzing hot bugs in the Linux kernel by clustering fixing commit messages. Proceedings of the Institute for System Programming of the Russian Academy of Sciences, vol. 35, no. 3, pp. 215-242, 2023.
[36]. Hoder K., Voronkov A. Comparing unification algorithms in first-order theorem proving. In Proc. KI 2009: Advances in Artificial Intelligence: 32nd Annual German Conference on AI, Paderborn, Germany, September 15-18, 2009. Proceedings 32. Springer, 2009, pp. 435–443.

Информация об авторах / Information about authors

Никита Александрович СТАРОВОЙТОВ – магистрант и ассистент кафедры прикладной математики. Сфера научных интересов: кластеризация, анализ текстов.

Nikita Aleksandrovich STAROVOYTOV – master student and assistant at the department of Applied Mathematics. Research interests: clusterization, text analysis.

Сергей Михайлович СТАРОЛЕТОВ – кандидат физико-математических наук, доцент. Сфера научных интересов: формальная верификация, model checking, киберфизические системы, операционные системы.

Sergey Mikhailovich STAROLETOV – Cand. Sci. (Phys.-Math.), associate professor. Research interests: formal verification, model checking, cyber-physical systems, operating systems.

DOI: 10.15514/ISPRAS-2024-36(2)-4

Four-Dimensional ACC Analysis

[1] *N.I. Mustafina*, ORCID: 0000-0001-8617-659X <*nazgul-2003@mail.ru*>
[1,2] *M.A. Plaksin*, ORCID: 0000-0002-6288-8610 <*mapl@list.ru*>
[1] *P.A. Mikisheva*, ORCID: 0009-0003-0246-5418 <*mikisheva.p@yandex.ru*>

[1] *HSE University,*
38 Studencheskaya Ulitsa, Perm, 614070, Russia.
[2] *Perm State University,*
15, Bukireva, Perm, 614068, Russia.

Abstract. The article discusses the issues of planning and resource management in the process of testing software systems. The paper presents the ACC analysis method used at Google to optimize the distribution of efforts for testing different parts of the system. Extending the method by adding a fourth characteristic - actors (roles of system users) – allows for a more flexible assessment of action requirements and user skill levels. Illustrative examples of system attributes and components help understand the principles of the method. The work proposes a new approach to risk management and process improvement in testing software systems in a multidimensional space. The effectiveness of applying the enhanced ACC analysis method using a risk-oriented approach was demonstrated using the example of a control system for technological operations in the repair of electric motors, for which attributes, components, actors were identified, opportunities at their intersection were analyzed, and testing was conducted, which helped improve the system's quality.

Keywords: testing; risk registry; test plan; test cases; system for controlling technological operations; ACC methodology; Google+; risk-oriented approach; risk identification; efficiency improvement; risk assessment; productivity enhancement; process optimization; data analysis; software engineering.

For citation: Mustafina N. I., Plaksin M.A., Mikisheva P.A. Four-dimensional ACC analysis. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 47-58. DOI: 10.15514/ISPRAS-2024-36(2)-4.

Четырёхмерный ACC анализ

[1] *Н.И. Мустафина*, ORCID: 0000-0001-8617-659X <*nazgul-2003@mail.ru*>
[1,2] *М.А. Плаксин,* ORCID: 0000-0002-6288-8610 <*mapl@list.ru*>
[1] *П.А. Микишева,* ORCID: 0009-0003-0246-5418 <*mikisheva.p@yandex.ru*>

[1] *НИУ Высшая школа экономики (Пермский филиал),*
614070, Россия, г. Пермь, ул. Студенческая, 38.
[2] *Пермский государственный национальный исследовательский университет,*
614990, Россия, г. Пермь, ул. Букирева, 15.

Аннотация. В статье обсуждаются вопросы планирования и управления ресурсами в процессе тестирования программных систем. В работе представлен метод анализа ACC, используемый в Google для оптимизации распределения усилий по тестированию различных частей системы. Расширение метода путем добавления четвертой характеристики - актеров (роли пользователей системы) - позволяет более гибко оценивать требования к действиям и уровни навыков пользователей. Иллюстративные примеры атрибутов и компонентов системы помогают понять принципы метода. В работе предлагается новый подход к управлению рисками и улучшению процессов в тестировании программных систем в многомерном пространстве. Эффективность применения улучшенного метода анализа ACC с использованием риско-ориентированного подхода была продемонстрирована на примере системы управления технологическими операциями по ремонту электродвигателей, для которой были определены атрибуты, компоненты, актеры, проанализированы возможности их пересечения, проведено тестирование, что помогло улучшить качество системы.

Ключевые слова: тестирование; реестр рисков; план тестирования; тест-кейсы; система управления технологическими операциями; методология ACC; Google+; риск-ориентированный подход; выявление рисков; улучшение эффективности; оценка рисков; повышение производительности; оптимизация процессов; анализ данных; программная инженерия.

Для цитирования: Мустафина Н.И., Плаксин М.А., Микишева П.А. Four-dimensional ACC analysis. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 47–58 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–4.

1. Introduction

Testing is one of the most important processes in software system development. The main goal of testing is to detect errors and defects in the product being tested, as well as to identify discrepancies between the product's characteristics and the requirements and expectations of users [1-15]. One of the main problems associated with testing is the lack of resources for exhaustive testing. A good test plan should be created at the beginning of a project and may change during the development of the software product. This makes tasks such as planning and management relevant: the task of best resource allocation allocated to the testing phase, and the task of assessing the progress made during testing.

At Google, a special method called ACC analysis [16] is used to address these challenges. It allows ranking different parts of the developing system using a popular risk-oriented approach today, providing recommendations on what percentage of efforts should reasonably be planned for testing specific parts of the system.

The original ACC analysis manipulates three characteristics of a software system: attributes, components, and capabilities. ACC stands for Attribute, Component, Capability. This article proposes to expand the method by including a fourth characteristic – actors (classes of system users). This increases the manageability of the testing process, allows for a different perspective on testing organization, makes the process more flexible, clarifies requirements for implementing specific actions, and also the level of user qualification.

The article describes the ACC analysis method used for software testing and its proposed enhancement. ACC analysis is a risk-oriented approach that ranks different parts of a developing

system based on the percentage of efforts required for testing. The original ACC analysis considers three characteristics: attributes, components, and capabilities. The proposed enhancement adds a fourth characteristic: actors (roles of system users). It increases the manageability of the testing process, provides a different perspective on testing organization, makes the process more flexible, clarifies requirements for implementing specific actions, and considers the level of user qualification. The text also provides an example of applying the improved ACC method for analyzing a Repair Shop system. The enhancement allows for better distribution of testing resources, prioritization of testing based on risk levels, and consideration of user roles and qualifications.

2. The original ACC method

Since ACC analysis is not well known enough, first a description of the traditional methodology is provided, followed by our proposed enhancement. The results of applying ACC to analyze the system for repairing electric motors are then presented.

As an illustration, the application of ACC analysis for testing the system for controlling technological operations in the repair of electric motors (Repair Shop) is described. The system operates according to the following scheme:

- Users of the Maintenance and Repair Shop can be divided into roles: worker, master, workshop supervisor, director, and Maintenance and Repair Shop administrator. Several individuals can belong to each category, except for the director.
- During engine repair, various "operations" are performed. Each operation belongs to one of the "operation groups."
- A broken engine is brought to the company for repairs. The workshop supervisor registers it in the Maintenance and Repair Shop: creates a "card" for it, assigns a unique number to the engine, and determines the list of necessary operations. Subsequently, as the repair progresses, the card will be marked with the completion of each operation.
- The card is placed in the "In Progress" list. The card is visible to all users of all types.
- The workshop supervisor assigns a master responsible for the engine repair.
- The master assigns workers to perform each operation, is responsible for the start and end of the work, makes notes in the engine card about the completion, suspension, and completion of the operations as the repair progresses.
- After the completion of the last operation, the engine card is automatically moved from the "In Progress" list to the "Completed" list.
- The Maintenance and Repair Shop administrator: adds, edits, and deletes users, assigns them a category; adds, edits, and deletes operations and operation groups, assigns an operation to a group; adds, edits, and deletes customers;
- The director has the ability to generate reports that provide information on which worker performed a specific operation, and how much time was spent.

The system model created using ACC analysis significantly differs from traditional models such as functional, structural, flow, and parametric. From an ACC perspective, the system is represented as a matrix, where columns correspond to system attributes, rows to components, and cells contain the capabilities that the system provides to the user. This matrix is constructed as follows.

First, key characteristics of the system are identified, qualities that are important to the user and in which the developed software system should stand out from analogs. In the context of ACC, these are called attributes and are typically expressed as adjectives. Their number is small.

As an illustration, the following list of attributes for the social network Google+ is provided: Social (allows users to exchange information and thoughts), Expressive (users use the product's features for self-expression), Simple (users easily understand how to do what they want), Relevant (shows

only information that interests the user), Expandable (integrates with other Google resources, third-party sites, and applications), Confidential (user data will not be disclosed).

For our illustrative Repair Shop system, the following attributes were identified: "Simple" (offers users only intuitive actions), "Convenient" (minimizes time for frequently performed actions), "Accessible" (allows users with different roles to connect), "Secure" (protects information from external threats).

The second step of ACC analysis involves identifying "components." The concept of components in ACC differs from the traditional understanding. Components are the structural units of the system, not in terms of program structure but from the user's perspective. Components are the key parts of code that make the program what it is.

For the social network Google+, components include Profile, People, Feed, Circles, Notifications, Interests, Posts, Comments, Photos. For our illustrative Repair Shop system, components include Search, Repair Card, In Progress, Completed, Reports, Users, Groups, Operations, Customers.

The third stage of ACC analysis involves describing the "capabilities of the system" - actions that the system can perform at the user's request. As expected for actions, they are expressed using verbs. In the ACC model, capabilities do not exist on their own. They are linked to components and attributes. It is considered that each capability is implemented by a certain component with the aim of providing a certain quality of the product (a certain attribute). For example, for the social network Google+, the component "View Page" interacts with the attribute "Accessible" in three capabilities:

- make the document accessible to employees;
- allow employees to edit the document;
- display the employee's position on the page.

This results in a matrix where columns correspond to attributes, rows to components, and capabilities are recorded in the cells. Fig. 1 shows the matrix model for Google+ from [16, p.132].

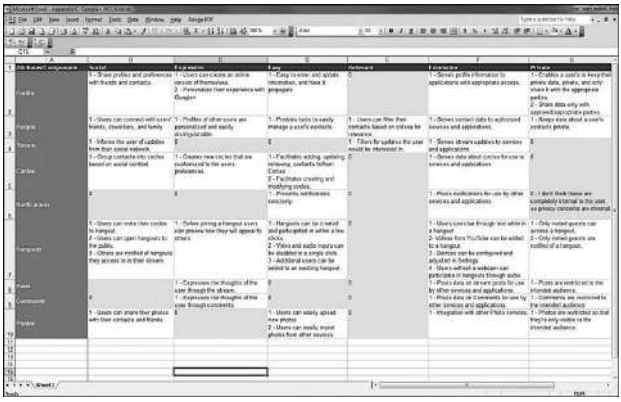


Fig. 1. ACC Table for Google+

A matrix of "attributes-components-capabilities" for our illustrative Repair Shop system is shown in Tables 1 and 2.

There can be many capabilities (tens or hundreds). They provide the result for which the user uses the system. Therefore, the correctness of their implementation should be verified. This means that each capability should be tested at least once.

Already, this matrix is useful as a source of information for building a testing plan "components-attributes-capabilities".

- Each capability requires at least one test. Therefore, the number of capabilities in a table cell indicates the minimum number of tests associated with that cell. It is easy to identify cells, rows, and columns that require maximum testing efforts.

- Each row and each column represent a certain logical integrity. All cells in a row (column) are connected. Therefore, it is logical to test them together. Therefore, each row and each column can serve as a test session assignment. This way, we eliminate duplication and ensure a high level of coverage.

However, the ACC analysis goes further. To increase the informativeness of the model, it involves a risk-oriented approach. This is done as follows.

So far, we have considered all capabilities equal in terms of testing. But in reality, this is not the case. Some capabilities are more significant, while others are less significant. It is necessary to test the more significant capabilities first. (There may not be enough resources to test everything indiscriminately.) The question is: how to determine the significance of capabilities from a testing perspective? ACC proposes to assess the risk of their failure.

Two characteristics are evaluated for each capability: the probability of failures and the degree of criticality of failures. The probability is assessed on a scale of "very rare - rare - sometimes - often." The criticality of failure is assessed on a scale of "minimal (the user may not even notice) - minor - significant - maximum (a blow to the product's reputation; will make the user stop using it)." In both cases, an even number of values is deliberately set on the scales. This is done intentionally to deprive the tester of the opportunity to choose an average option.

There can be many capabilities (tens or hundreds). They provide the result for which the user uses the system. Therefore, the correctness of their implementation should be verified. This means that each capability should be tested at least once.

Already, this matrix is useful as a source of information for building a testing plan "components-attributes-capabilities".

- Each capability requires at least one test. Therefore, the number of capabilities in a table cell indicates the minimum number of tests associated with that cell. It is easy to identify cells, rows, and columns that require maximum testing efforts.
- Each row and each column represent a certain logical integrity. All cells in a row (column) are connected. Therefore, it is logical to test them together. Therefore, each row and each column can serve as a test session assignment. This way, we eliminate duplication and ensure a high level of coverage.

However, the ACC analysis goes further. To increase the informativeness of the model, it involves a risk-oriented approach. This is done as follows.

So far, we have considered all capabilities equal in terms of testing. But in reality, this is not the case. Some capabilities are more significant, while others are less significant. It is necessary to test the more significant capabilities first. (There may not be enough resources to test everything indiscriminately.) The question is: how to determine the significance of capabilities from a testing perspective? ACC proposes to assess the risk of their failure.

Two characteristics are evaluated for each capability: the probability of failures and the degree of criticality of failures. The probability is assessed on a scale of "very rare – rare – sometimes – often." The criticality of failure is assessed on a scale of "minimal (the user may not even notice) – minor – significant – maximum (a blow to the product's reputation; will make the user stop using it)." In both cases, an even number of values is deliberately set on the scales. This is done intentionally to deprive the tester of the opportunity to choose an average option.

After evaluating the probability and criticality of failures for each capability, the risk value (the product of probability and criticality) is calculated and added to the "components-attributes-capabilities" matrix. For better visualization, the matrix is presented as a "heat map": 1-2 – green risks, 3-4 – yellow, 6-9 – orange, 12-16 – red (5, 10, 11 cannot be).

(When there are multiple possibilities in one cell [16], it recommends averaging their risks. In the opinion of the authors of this article, this recommendation is strange. In our opinion, either the maximum or the sum should be taken.)

Table 1. ACC table for the attributes simple and user-friendly for Repair Shop

	1/ Simple: Intuitive actions			*2/ Convenient: minimizing operations for frequently performed actions*	
A/ Search	**Administrator**: 1. Search for employees by name and email *(probability – very rarely, criticality minimal)* *Risk 1*	**Administrator**: 2. Search for a specific order by repair number *(probability – very rarely, criticality minimal)* *Risk 1*		**Administrator**: 1. On each tab, you can search (jobs, groups, operations, reports, customers) by key information *(probability – very rarely, criticality minimal)* *Risk 1*	
B/ Repair Map	**Master**: 1. Adding a description to the order *(probability – very rarely, criticality low)* *Risk 2*	**Master**: 2. Assign workers to order *(probability – often, criticality significant)* *Risk 12*		1. The order automatically moves from the completed list to the finished list *(probability - very rarely, criticality significant)* *Risk 4*	**Master**: 2. Start/finish/suspend repairs *(probability-often, criticality maximum)* *Risk 16*
C/ In progress	**Head of Department:** 1 Review cards of unfinished tasks *(probability – often, criticality maximum)* *Risk 16*			**All**: 1. Display selected number of current tasks *(probability – very rarely, criticality minimal)* *Risk 1*	
D/ Completed	**Head of Department:** 1. Review cards of completed tasks *(probability – often, criticality maximum)* *Risk 16*			**Master**: 1. Change task status from completed to incomplete *(probability – sometimes, criticality maximum)* *Risk 12*	**Director**: 2. Send email notification of task completion *(probability – rarely, criticality minimal)* *Risk 2*
E/ Reports	**Director**: 1. Download report *(probability – rarely, criticality low)* *Risk 4*			**Director**: 1. Check how much time a worker spent on performing the operation *(probability – sometimes, criticality low)* *Risk 6*	**Director**: 2. Review which operations were performed by specific workers during a specified period *(probability – sometimes, criticality low)* *Risk 6*
F/ Users	**Administrator**: 1. Add users *(probability – very rarely, criticality significant)* *Risk 3*	**Administrator**: 2. Delete users *(probability – very rarely, criticality significant)* *Risk 3*	**Administrator**: 3. Edit users *(probability – very rarely, criticality significant)* *Risk 3*	**Administrator**: 1. View selected number of user records *(probability - very rarely, criticality low)* *Risk 2*	**Administrator**: 2. Dismiss users *(probability - very rarely, criticality low)* *Risk 2*
G/ Groups	**Administrator**: 1. Add groups *(probability – very rarely, criticality significant)* *Risk 3*	**Administrator**: 2. Delete groups *(probability – very rarely, criticality significant)* *Risk 3*	**Administrator**: 3. Edit groups *(probability – very rarely, criticality significant)* *Risk 3*	**Administrator**: 1. Adding operations to a group from a pre-formed list *(probability – rarely, criticality significant)* *Risk 6*	
H/ Operations	**Administrator**: 1. Add operations *(probability – very rarely, criticality significant)* *Risk 3*	**Administrator**: 2. Delete operations *(probability – very rarely, criticality significant)* *Risk 3*	**Administrator**: 3. Edit operations *(probability – very rarely, criticality significant)* *Risk 3*		
I/ Customers	**Administrator:** 1. Add customers *(probability – very rarely, criticality minimal)* *Risk 1*	**Administrator**: 2. Delete customers *(probability – very rarely, criticality minimal)* *Risk 1*	**Administrator**: 3. Edit customers *(probability – very rarely, criticality minimal)* *Risk 1*	1. Automatic notifications are automatically sent to mail on email for completed tasks *(probability – rarely, criticality significant)* *Risk 6*	

Table 2. ACC table for the attributes accessible and secure for Repair Shop

	3/ Affordable: allows connecting users with different roles to connect	*4/ Secure: protects information from against various threats*
A/ Search	**All**: 1. Search by job number and customer in current and completed work *(probability - very rarely, criticality minimal)* *Risk 1*	
B/ Repair Map	**All:** 1. View repair completion status *(probability – very rarely, criticality maximum)* *Risk 4*	
C/ In progress	**All**: 1. View all works in active and suspended states *(probability- rarely, criticality maximum)* *Risk 8*	
D/ Completed	**All:** 1. View the list of completed works, their completion date, customer, repair number *(probability - very rarely, criticality minimal)* *Risk 1*	**Administrator**: 1. Cannot change master and workers after work has started, which helps prevent scheduling conflicts *(probability- very rarely, criticality significant)* *Risk 3*
E/ Reports		**Director**: 1. Keep reports confidential *(probability - very rarely, criticality maximum)* *Risk 4*
F/ Users		**Administrator**: 1. Assign roles with limited access rights *(probability- rarely, criticality maximum)* *Risk 8*
G/ Groups		**Administrator**: 1. Keep operation groups confidential *(probability- very rarely, criticality maximum)* *Risk 4*
H/ Operations		**Administrator**: 1. Keep operations confidential *(probability - very rarely, criticality maximum)* *Risk 4*
I/ Customers		**Administrator**: 1. Keep customers confidential *(probability – very rarely, criticality maximum)* *Risk 4*

The informativeness of the matrix sharply increases. It provides information to answer questions such as:

- How to distribute the resources allocated for testing among different functions and components of the system? Which functions and components should receive more attention, and which less? What should be tested first?
- What is the criterion for completing testing? When do we have the right to say, "We have tested everything"?

Further, the description of the enhancement introduced by the authors in the ACC analysis begins.

3. Our suggestions for improvements of ACC analysis

A fourth dimension - actors, classes of system users - was added to the three dimensions of classical ACC analysis (attributes-components-capabilities).

All users of the Repair Shop system are divided into five classes (playing one of five roles): director, workshop manager, master, administrator, worker. Each role has its needs, its goals in using the

Repair Shop system. Each action performed by the system (each capability of the system) is executed upon request of one or several roles. That is, each role has its own set of capabilities.

"Entering the fourth dimension" immediately provided a new perspective on the system, allowing it to be viewed from the user's standpoint. Another basis for grouping capabilities and assessing their risks emerged. What does this provide? Firstly, it is logical to conduct test sessions as the work of a specific role. Secondly, different users may have different qualifications. There should be a correspondence between the user's level of qualification and the level of riskiness of actions performed by them. Performing highly risky actions by low-skilled specialists increases the likelihood of failures. On one hand, there is an opportunity to specify requirements for implementing a specific action of the system (an action intended for low-skilled specialists should have low "riskiness"). On the other hand, "risk assessment" of actions allows determining the "riskiness assessment" of each role. Thus, defining requirements for the level of qualification of users performing that role (highly risky actions should only be performed by highly qualified specialists).

Ideas for further development of the method.

The fourth dimension – it does not necessarily have to be actors. Depending on the project, it can be something else.

We have moved from three-dimensional space to four-dimensional space. The logical next step is multidimensional space. It is possible to introduce consideration of the fifth, sixth, and further dimensions. The limitations here will be associated with the increasing complexity of the model. To combat this complexity, it is logical to use automation.

4. The example of applying improved ACC method for analyzing the repair shop system

As mentioned earlier, a total of 4 attributes, 9 components, and 41 features were identified that intersect attributes and components. The probability and criticality of failures for the features were evaluated, and risk levels were calculated based on these assessments. User roles were assigned to the features that were accessible to them. Most features were accessible to one role, a few to all roles, and one feature had no role assigned as the corresponding action was performed automatically.

The features were sorted in descending order of risk level. Overall, there were 5 red-level risks (three at level 16 and two at level 12), 6 orange-level risks (two at level 8 and four at level 6), 17 yellow-level risks, and 13 green-level risks.

The total sum of all risks was 231. This number can be considered as the overall risk level of the entire system.

The features were grouped by components, attributes, and actors. The total weights of the features by groups are presented in tables 3, 4 and 5 (The column "Sum after testing" will be explained later).

The discipline of session testing was chosen for testing.

Table 3. Component and risk sum table

Component	***Description***	***Sum of risks before testing***	***Sum of risks after testing***
Search	Search strings on different application tabs	4	4
Repair Map	Card with repair information	40	19
Work in	List of active and suspended repair cards	25	17
Completed	List of completed repair cards	34	26
Reports	Tab for viewing working hours and order information	20	20
Users	User data administration tab	30	26
Groups	Work group data administration tab	28	28

Operations	Possible engine operation administration tab	22	22
Customers	Customer work data administration tab	28	28

Table 4. Attribute and risk sum table

Attribute	***Description***	***Sum of risks before testing***	***Sum of risks after testing***
Simple	Intuitive actions	124	99
Convenient	Minimization of operations for frequently performed	66	54
Accessible	Allows connection for users with different roles	14	14
Secure	Protects information from various threats	27	23

Table 5. Role and risk sum table

Role	***Sum of risks before testing***	***Sum of risks after testing***
Administrator	*120*	*116*
Master	*42*	*21*
Workshop Manager	*32*	*16*
Director	*22*	*22*
All	*15*	*15*

The question arose of how to organize sessions based on what principle. Traditional ACC analysis suggests using rows and columns of a table, i.e., conducting testing "by components" or "by attributes". This is convenient for tracking the completeness of testing (it is sufficient to mark "closed" rows and columns). In our case, this order turned out to be not very convenient. The point is that each user should authenticate when logging into the system. This takes time. In the table, capabilities related to different roles often reside in the same row and column. So, to test one row (one column), it will be necessary to log in and out of the system several times. To avoid this, it was more convenient to conduct testing "by roles". Although this complicates tracking the completeness of testing (capabilities of one role are scattered in the table in different places). Another argument in favor of testing "by roles" was the simplicity of building test scenarios. When testing "by roles", it's easy to do this (unlike testing "by components" and "by attributes").

At the same time, the question of the order of checking "actors" arose. It is noticed that Table V leads to an incorrect decision. The thought arises that it should be checked based on the reduction of the sum of risks related to the actor. This is incorrect. A large sum can be obtained not because it includes the most significant risks, but because it includes many less significant risks. This is precisely the situation reflected in Table IV. The role of Administrator carries the greatest weight here. However, the Administrator does not have any "red" risks. In terms of "red" risks, the roles of Master and Workshop Manager take the lead. The former has three "red" risks (16 + 12 + 12 = 40), and the latter has two (16 + 16 = 32). However, another factor intervenes in determining the order of "role testing": the order of filling the information base. According to this factor, the role of Administrator was brought to the forefront. Testing the capabilities of all other actors required a filled information base (operations, operation groups, users performing different roles). Therefore, it was decided to first check the Administrator's actions to fill and adjust the information base, and only after that to check the riskiest capabilities. Thus, the table "components-attributes-capabilities" was used as the basis for building the testing plan.

Another role played by this table is the basis for building the testing completion criterion. The criterion chosen was the change in the level of system riskiness, i.e., the total sum of all risks. We proceeded from the assumption that as a result of testing, the probability of system failures would

decrease. And this means that the magnitude of risks would decrease. (Testing will not be able to affect the criticality of failures.) The criterion for ending testing was chosen as a 15% reduction in system riskiness.

A total of 13 errors were found during testing. As these errors were corrected, the probability of failures was reassessed, and the level of system riskiness was recalculated. The new risk values are shown in Tables 3, 4 and 5 in the column "Risk Sums after Testing". After correcting the thirteenth error, the level of system riskiness decreased by 18%. This means that the criteria for ending testing were satisfied.

Table 5 shows that the most progress was achieved for those roles to which the riskiest capabilities were attributed. The sum of risks for the Master decreased from 42 to 21, and for the Workshop Manager from 32 to 16.

In comparison to other testing methodologies such as RUP and IEEE, which focus more on test formatting advice, ACC addresses both the structure and content of the information system. Additionally, using a risk-oriented approach helps in creating efficient tests due to prioritizing capabilities based on their failure probability and criticality, considering the frequent time constraints.

Conclusion

The article presents proposals for improving the Activity-Components-Component (ACC) analysis method. In addition to the three dimensions of the traditional ACC analysis - "attributes-components-capabilities," it is proposed to add a fourth dimension - "actors" (roles). This provides a new perspective on the system – a user-oriented view, providing another opportunity for organizing testing.

The application of the enhanced ACC method is demonstrated in organizing the testing of a specific software system - a system for monitoring the technological operations of repairing electric motors. The addition of the "actors" dimension facilitated the optimization of test sessions organization. The main focus was on testing the riskiest capabilities. During testing, 13 errors were identified and corrected, leading to an 18% reduction in the overall system risk level.

Further development of the ACC method may involve either replacing the "actors" parameter with another parameter or continuing to increase the number of dimensions, making ACC analysis five-dimensional, six-dimensional, etc. This will make the method more complex and raise questions about its automation.

References

[1]. Kulakov K. A., Dimitrov V. M. Fundamentals of software testing //Electronic textbook for students of the Institute of Mathematics and Information Technologies. /Petrozavodsk: PetrSU. 2018.

[2]. Safiulin R.Z. Development of testing technologies in education // Education management: theory and practice. 2015. №1 (17). URL: https://cyberleninka.ru/article/n/razvitie-tehnologiy-testirovaniya-v-obrazovanii (date of reference: 15.11.2023).

[3]. Karpunin Aleksey Aleksandrovich, Ganev Yuri Mikhailovich, Chernov Maxim Mikhailovich Quality assurance methods in the design of complex software systems // NIKSS. 2015. №2 (10). URL: https://cyberleninka.ru/article/n/metody-obespecheniya-kachestva-pri-proektirovanii-slozhnyh-programmnyh-sistem (date of reference: 15.11.2023).

[4]. Galimova, E. Yu. Methodology for selecting automated, manual and mixed way of testing a software product based on quality criteria // Proceedings of Tula State University. Technical Sciences. 2019. - №. 7. C. 248-256.

[5]. Kulikov S. S. et al. Software testing: textbook. 2019.

[6]. Polevshchikov, I. S., Chirkov, M. S., Levanov, A. A. V. Automated system of test-plans development in software testing // Engineering Gazette of Don. 2019. - №. 8 (59). C. 29.

[7]. Piven A. A., Skorin Yu. I. Software testing // Sistemy obrokobki informatsii. 2012. - №. 4 (1). - C. 56-58.

[8]. Kuvshinova E. A., Glazova V. F. TESTING AS IMPORTANT COMPONENT OF THE SYSTEM OF CONTROL OF SOFTWARE QUALITY // Applied Mathematics and Informatics: Modern Research in Natural and Technical Sciences. 2017. С. 305-308.
[9]. Drobysh A. A., Santsevich S. N. Debugging and testing of software. 2020.
[10]. Shakirova A. I., Khasyanov A. F., Dautov E. F. Software testing time reduction // Modern Science-Intensive Technologies. 2019. - №. 7. С. 104-109.
[11]. Vildanova K. I. Choice of software testing method // Scientific Notes of UlSU. Series" Mathematics and Information Technologies". 2022. - №. 2. - С. 31-37.
[12]. Moiseev D. A. Methodology and process of manual testing // Reliability and quality of complex systems. 2017. - №. 3 (19). С. 107-112.
[13]. Aksenov D. O., Khafizov E. U., Ryabov M. A. Software Testing Management System.
[14]. Ivanov E. C. Development of software testing methodology: master's thesis. 2014.
[15]. Viktorova V. S., Stepanyants A. S. Models and methods for calculating the reliability of technical systems //M.: LENAND. 2016.
[16]. Carollo J., Whittaker J., Arbon J. How testing is done at Google. 2012.
[17]. Plaksin M.A. Testing and Debugging Programs for Future and Present Professionals // Moscow: BINOM, 2023.

Информация об авторах / Information about authors

Назгуль Ибрагимовна МУСТАФИНА - студент 3-го курса Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь). Сфера научных интересов: машинное обучение, анализ данных.

Nazgul Ibragimovna MUSTAFINA - 3rd year student of the National Research University – Higher School of Economics (HSE University, Perm Branch). Research interests: machine learning, data analysis.

Михаил Александрович ПЛАКСИН - кандидат физико-математических наук, доцент кафедры информационных технологий в бизнесе Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь), доцент кафедры математического обеспечения вычислительных систем государственного национального исследовательского университета (ПГНИУ). Сфера научных интересов: системный анализ, ТРИЗ, управление проектами, информатизация образования, преподавание информатики в начальной школе, построение сквозного курса информатики с 1-го по 11-й класс.

Mikhail Alexandrovich PLAKSIN - Candidate of Physical and Mathematical Sciences, Associate Professor of the Department of Information Technologies in Business of the National Research University – Higher School of Economics (HSE University, Perm Branch), Associate Professor of the Department of Computer Science of the Perm State National Research University (PSU). Research interests: system analysis, TRIZ, project management, informatization of education, teaching informatics in elementary school, construction of a cross-curricular informatics course from 1st to 11th grade.

Полина Алексеевна МИКИШЕВА обучается на 3-м курсе на направлении «Программная инженерия» в Пермском филиале НИУ Высшая школа экономики.

Polina Alekseevna MIKISHEVA is a 3rd year student of the Software Engineering program at the Perm branch of the National Research University Higher School of Economics.

DOI: 10.15514/ISPRAS-2024-36(2)-5

On the Automated Unit Tests Generation for Java Applications using Spring Framework

K.A. Shishin, <kirill.a.shishin@gmail.com>
I.V. Muravev, <muravjovilya@gmail.com>
E.K. Kulikov, ORCID: 0000-0002-8313-0227, <egor.k.kulikov@gmail.com>
St. Petersburg State University,
7/9, University Embankment, Saint Petersburg, 199034, Russia

Abstract. This paper considers the automated unit tests generation for programs written in Java using the Spring framework. Although several test generation tools for "pure" Java applications have been developed in recent decades, the features of this framework are mostly not taken into account. However, Spring is used to develop many industrial Java applications. At the same time, the presence of Spring components in the application for which the tests are generated imposes additional requirements not only on the code analysis approaches, but also on the structure of the generated tests. The main source of the information about object types and their properties is the Spring application context. The paper proposes an instrument for analyzing the application context, that in some cases allows generating test scenarios corresponding to real program executions and avoiding excessive mocking. The full initialization of the application context does not occur during this analysis. It makes the test generation safe for user data. The proposed instrument for analyzing the Spring context has been integrated into the UnitTestBot Java automatic test generation tool. We also provide examples of tests generated for real open-source projects.

Keywords: Software testing; automated unit test generation; Spring; mocking; UnitTestBot Java.

For citation: Shishin K.A., Muravev I.V., Kulikov E.K. On the automated unit tests generation for Java applications using Spring. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 59-72. DOI: 10.15514/ISPRAS-2024-36(2)-5.

Об автоматической генерации модульных тестов для Java-приложений, использующих фреймворк Spring

К.А. Шишин, <kirill.a.shishin@gmail.com>
И.В. Муравьёв, <muravjovilya@gmail.com>
Е.К. Куликов, ORCID: 0000-0002-8313-0227, <egor.k.kulikov@gmail.com>
Санкт-Петербургский государственный университет,
Россия, 199034, г. Санкт-Петербург, Университетская наб., д. 7/9.

Аннотация. Данная работа посвящена автоматической генерации модульных тестов для приложений на языке Java, использующих фреймворк Spring. Хотя в последние десятилетия было создано несколько инструментов автоматической генерации тестов для «чистой» Java, специфические особенности этого фреймворка, как правило, не принимались во внимание. Тем не менее, Spring используется при разработке многих промышленных приложений на Java. Использование фреймворка в приложении, для которого необходимо сгенерировать тесты, накладывает дополнительные требования не только к используемым методам анализа кода, но и к виду предлагаемых тестов. Главным источником информации о типах и свойствах объектов в Spring-приложении является его контекст. В данной работе предлагается механизм анализа контекста приложения, который в некоторых случаях позволяет генерировать тестовые сценарии, соответствующие реальному исполнению программы, избегая избыточного мокирования. При этом полная инициализация контекста приложения в процессе анализа не происходит, что делает генерацию тестов безопасной для пользовательских данных. Предложенный инструмент анализа контекста Spring приложения был интегрирован в инструмент автоматической генерации тестов UnitTestBot Java. В заключение приводятся примеры тестов, сгенерированных для некоторых проектов с открытым исходным кодом.

Ключевые слова: Тестирование программного обеспечения; автоматическая генерация модульных тестов; фреймворк Spring; мокирование; UnitTestBot Java.

Для цитирования: Шишин К.А., Муравьёв И.В., Куликов Е.К. Об автоматической генерации модульных тестов для Java-приложений, использующих фреймворк Spring. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 59–72 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–5.

1. Introduction

Software testing is an important and essential part of any project. While manually written tests often cover only a small percentage of program execution paths, their development usually requires significant effort and time from developers and testers. That's why solutions for the automated test generation have been actively developed over the last decades. They are intended to help their users to significantly increase the test coverage of a program, reducing many times the efforts spent on writing tests. There is a well-known experiment [1] on the *Coreutils*[1] project, showing how effective and useful automated test generation can be. Although this project has been developed for many years and tests have been written manually all the time, the test automation tool created by the authors of the experiment managed to significantly increase code coverage with tests in just a few hours and found some defects that had been unknown for more than 15 years. The relevance of automated test generation is also confirmed by the annual competitions of test generation tools [2]. At these competitions, participating tools test real projects (e.g., *Guava*[2], *Seata*[3], *Spoon*[4], etc.). As a result, winners are identified in several categories, including the efficiency of code defect detection, the percentage of program lines covered, and the human-readability of the generated tests.

[1] Coreutils project. Available: https://www.gnu.org/software/coreutils/
[2] Guava: Google Core Libraries for Java. Available: https://github.com/google/guava
[3] Seata: Simple Extensible Autonomous Transaction Architecture. Available: https://github.com/apache/incubator-seata
[4] Spoon. Available: https://github.com/INRIA/spoon

Among the automated test generators showing good results at competitions, we would like to mention the open-source project *UnitTestBot Java*[5], which is being developed at Huawei Russian Research Institute. In 2022, the tool performed well in the competition [3], and in 2023, it was ranked first in the human-readability of generated tests and second overall in all categories [4]. *UnitTestBot Java* is a command-line tool and a plugin for *IntelliJ IDEA*, which aims to generate unit tests for Java applications. The methodological basis of the project is two code analysis techniques: symbolic execution and fuzzing.

UnitTestBot Java is good at generating tests for code in "pure" Java, that is, without the use of special frameworks. However, such frameworks are used quite often and impose additional requirements on the analysis of user code and the format of generated tests. One of the most popular frameworks today [5] is *Spring*[6]. It is used in the development of many Java projects. Therefore, it is important for *UnitTestBot Java* to be capable of generating specialized tests for applications that rely on this framework.

Spring is a diverse framework. However, one of its main features are *Dependency Injection (DI)* and *Inversion of Control (IoC)*. Spring has a *DI/IoC* container[7] that stores managed objects — *beans*. The configuration of this container can be defined in various ways: through annotations in user code, special configuration classes, or XML files. In addition, the user can also choose a *profile* that defines how the application will be configured. More than that, the framework offers a mechanism for implementing the MVC pattern, distinguishing between services and controllers, each of which requires a distinct testing approach as agreed by human testers. It is important that the generated tests must be not only correct, but should also follow the commonly accepted guidelines for testing Spring applications [6-8]. These and other features of the framework make automated testing of Spring applications a very challenging task.

When creating unit tests, we usually test a component in isolation from the external environment (other microservices, databases, authentication mechanisms). Therefore, many complicated features of the Spring framework do not have a significant impact on the unit test generation mechanisms and will be important only in the case of creating integration or end-to-end tests, meaning that "pure" Java testing techniques can be used. However, when it comes to virtual calls, in a "pure" Java program, there is no reliable way to determine which implementation should be called in tests. In contrast, in a Spring application, there is a configuration that determines which concrete classes will be used in place of abstract types at runtime. Using this type substitution information, it is possible to generate tests that are better aligned with the actual behavior of the program and test those scenarios that can occur during application use.

It is also important to note that since unit testing does not involve launching an application and initializing its context, and usually means testing a component in isolation, it is expected that full context initialization will not take place during the automated test generation, which includes analysis of the user configuration. Otherwise, test generation may be dangerous for user data: for example, during context initialization, environment variables may be unexpectedly set, some data may be loaded from third-party services, etc. This creates additional challenges when solving the task of test generation for applications written with Spring.

2. Spring-based test generation

Let's consider a minimalistic example in which the type information from the application configuration allows generating tests better representing the real behavior of the program.

Assume the task is to generate tests for the `getSpecies()` method from the `SpeciesService` class, for example:

[5] UnitTestBot. Available: https://www.utbot.org

[6] Spring framework. Available: https://spring.io

[7] The IoC container. Available: https://docs.spring.io/spring-framework/reference/core/beans.html

```
@Service
public class SpeciesService {
    @Autowired
    @Setter
    private Animal animal;

    public String getSpecies() {
        return animal.getSpecies();
    }
}
```

where `Animal` is the interface having the following form:

```
public interface Animal {
    String getSpecies();
}
```

Suppose that we have two implementations of this interface in the project.

```
public class Cat implements Animal {
    public String getSpecies() {
       return "cat";
    }
}

public class Dog implements Animal {
    public String getSpecies() {
       return "dog";
    }
}
```

However, in the application configuration, only one of them — `Cat` — is used to create the `Animal` bean.

```
public class AnimalConfiguration {
    @Bean
    public Animal animal() {
        return new Cat();
    }
}
```

Assume that we develop tests for this method manually. One possible approach is mocking the virtual call of `getSpecies()` method.

```
public void testGetSpecies() {
    SpeciesService speciesService = new SpeciesService();

    Animal animalMock = mock(Animal.class);
    when(animalMock.getSpecies()).thenReturn("mouse");

    speciesService.setAnimal(animalMock);

    String actual = speciesService.getSpecies();
    assertEquals("mouse", actual);
}
```

Such a test is formally correct, but far from checking the actual behavior of the program and therefore is hardly valuable in practice.

At the same time, when developing tests, we have an opportunity to investigate the application configuration and to find out that only `Cat` implementation is used for a given interface and to write a test that much more closely resembles the real behavior.

```
public void testGetSpecies() {
    SpeciesService speciesService = new SpeciesService();

    Cat animal = new Cat();
    speciesService.setAnimal(animal);

    String actual = speciesService.getSpecies();
    assertEquals("cat", actual);
}
```

Now, let's consider the scenario of automated test generation. If we do not extract type information from the Spring application configuration indicating which of the `Animal` implementations is preferred, we can either generate the already mentioned test with a mock or choose any of the `Animal` interface implementations arbitrarily. In this way, another formally correct but valueless test using the `Dog` implementation can be generated. However, if the application configuration is analyzed and the test generation tool is provided with the information that only the `Cat` implementation is used for the `Animal` interface, then the generated test will accurately represent the actual behavior of the program.

Despite the fact that the type concretization described above cannot always be done (due to possible ambiguity of the possible types choice) and it is not always necessary to do it, in some cases, concretization allows for generating more expressive tests for Spring applications that verify real execution scenarios. Therefore, the ability to generate tests with type concretization is a desirable option for the test generator.

3. Overview

3.1 Existing tools

There are a number of tools that to some extent solve the problem of automated testing of programs in Java. All of them use one or several basic code analysis techniques: symbolic execution, fuzzing or machine learning [9]. The most famous open-source solutions are *EvoSuite*[8], *UnitTestBot Java*[9] and *Randoop*[10]. Among the commercial tools let us mention *Parasoft Jtest*[11], *Diffblue Cover*[12] and *Machinet*[13].

Among these tools, only a small subset can generate tests for Spring applications. For example, *Parasoft Jtest* generates only test method templates. Of course, this reduces the total time required to write tests, but the scope of the covered code depends on the user, who needs to substitute arguments with values in the code of the generated test method templates.

[8] What is EvoSuite? Available: https://github.com/EvoSuite/evosuite

[9] UnitTestBot Java: Automated unit test generation and precise code analysis for Java. Available: https://github.com/UnitTestBot/UTBotJava

[10] Randoop: Automatic unit test generation for Java. Available: https://randoop.github.io/randoop

[11] Parasoft Jtest for Java Unit Testing. Available: https://www.parasoft.com/products/parasoft-jtest/java-unit-testing

[12] What is Diffblue Cover? | Diffblue. Available: https://www.diffblue.com

[13] Machinet: AI Assistant for Developers. Available: https://www.machinet.net

The code snippet below shows an example of a generated test template for a Spring application using *Parasoft Jtest*. It is taken from the official *Parasoft* website[14].

```
@Test
public void testGetPerson() throws Throwable {
    MockedStatic<ExternalPersonService> mocked =
            mockStatic(ExternalPersonService.class);

    mocks.add(mocked);

    Person getPersonResult = null; // UTA: default value
    mocked.when(
            () -> ExternalPersonService.getPerson(anyInt())
    ).thenReturn(getPersonResult);

    // Given
    PeopleController underTest = new PeopleController();

    // When
    int id = 1;
    Model model = mock(Model.class);
    ResponseEntity<Person> result =
            underTest.getPerson(id, model);

    // Then
    assertNotNull(result);
    assertNotNull(result.getBody());
}
```

Another example of a tool that can generate tests for Spring applications is *Diffblue Cover*. It is able to generate tests that take into account the Spring application specifics.
Below is an example of the test generated for a Spring application using *Diffblue Cover*.

```
@ContextConfiguration(classes = {SpeciesService.class})
@ExtendWith(SpringExtension.class)
class SpeciesServiceUnitTests {
    @MockBean
    private Animal animal;

    @Autowired
    private SpeciesService speciesService;
    @Test
    void testGetSpecies() {
        when(animal.getSpecies()).thenReturn("");
        assertEquals("", speciesService.getSpecies());
        verify(animal, atLeast(1)).getSpecies();
    }
}
```

[14] Accelerate Unit Testing of Spring Applications With Parasoft Jtest \& Unit Test Assistant. Available: https://alm.parasoft.com/hubfs/New_Pages/Whitepaper:\%20Accelerate\%20Unit\%20Testing\%20of\%20Spring\%20Applications\%20with\%20Parasoft\%20Jtest\%20and\%20Unit\%20Test\%20Assistant.pdf

However, such tests do not follow the test writing guidelines for Spring applications mentioned in the introduction (for example, it is rather unconventional to use a class under test for context configuration) and the tool does not offer a mechanism to deal with excessive mocking.

Thus, none of the test automation tools we are aware of offer mechanisms for generating tests based on the application configuration.

3.2 UnitTestBot Java

UnitTestBot Java is a part of the *UnitTestBot* tool lineup for automated unit test generation. The tool uses two mechanisms to generate test scenarios: a symbolic engine and a fuzzer.

The symbolic engine is one of the implementations of the symbolic execution paradigm [10]. It performs an analysis of the possible execution paths of a program by mapping a set of path constraints to each branch of execution. These constraints are expressed in terms of the logic of predicates, and then using the SMT solver *Z3*[15] their satisfiability is determined. Obtaining the possible paths of program execution, as well as their prioritization, comes from the control flow graph that is constructed from the byte code of the program. The byte code is preliminarily transformed into *Jimple* representation using the *Soot*[16]. With this transformation, the byte code takes a simpler representation having fewer instructions.

The fuzzer used in *UnitTestBot Java* applies the greybox fuzzing technique, which involves generating random input values for a concrete execution of the methods under test. After each concrete execution, the fuzzer obtains feedback about the change in execution path. Based on this feedback, it mutates the input values for the next iteration of its work.

More detailed information about the implementation of the symbolic engine and fuzzer in *UnitTestBot Java* can be found on the official website of the project or in the documentation in the repository on GitHub[17].

Both of these code analysis techniques are usually combined for maximum efficiency. In *UnitTestBot Java*, by default, 95% of the time allocated for test generation is given to the symbolic engine, and the fuzzer is used as an additional auxiliary tool.

4. Implementation

This section provides a Spring configuration analyzer implementation and describes a modernization of the *UnitTestBot Java* symbolic engine, which allows types to be concretized during test generation based on the information obtained from the configuration analyzer.

4.1 Spring configuration analyzer

The engine obtains Spring-specific information for type concretization from the user application configuration analyzer. This information is collected using Spring's own instruments during the initialization of the application context.

Spring context initialization consists of several steps:

1. Collecting bean definitions. During this phase application configurations are parsed and analyzed. As a result, in particular, bean definitions are created. They include information about the class of the bean, its properties and its relationships with other beans.
2. Configuration of the bean definitions (*BeanFactoryPostProcessor*[18]). Once the information about beans has been collected, Spring can modify these definitions before they are used to

[15] Z3. Available: https://github.com/Z3Prover/z3

[16] Soot. Available: https://github.com/soot-oss/soot

[17] UnitTestBot Java documentation. Available: https://github.com/UnitTestBot/UTBotJava/tree/main/docs

[18] BeanFactoryPostProcessor. Available: https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/beans/factory/config/BeanFactoryPostProcessor.html

create the beans themselves. Configuring bean definitions involves setting dependencies, specifying scope, configuring lifecycle and other parameters specific to the bean.

3. Creating the beans and configuring them. This stage involves creating and further configuring the bean instances based on their definitions. Class instances are created and initialization methods are called.

While the steps of collecting and configuring bean definitions are safe for the user application, the step of creating the beans may cause changes to user data. For this reason, we decided to embed ourselves in the Spring context initialization process and implement our own *BeanFactoryPostProcessor*. It collects all necessary and available to the analyzer information about beans at the stage of setting up the bean definitions, destroys bean definitions, and then stops any further application initialization. Thus, the creation of the beans is prevented. The general pipeline of type information collecting during Spring application context initialization is shown in the Fig. 1.

To analyze beans from a user application with our post processor, we start a "hybrid" Spring application whose classpath combines the classpaths of both the original user application and our Spring analyzer module. It is important to note that the Spring analyzer module has minimal dependencies, which helps avoid dependency conflicts with the user application. In particular, Spring analyzer does not depend on a specific version of Spring and utilizes reflection to handle any popular Spring version bundled with the user application. When starting such a "hybrid" Spring application, we first determine whether Spring Boot is used and, based on that, choose an appropriate application starter class. Furthermore, we dynamically patch annotations to make the started application use the desired Spring configuration (Java- or XML- based) and profiles.

In this way, we get an algorithm for configuring our own Spring application, shown in the Fig. 2, while the entire process of collecting Spring-specific information for type concretization by the configuration analyzer is represented with the chain of actions shown in the Fig. 3.

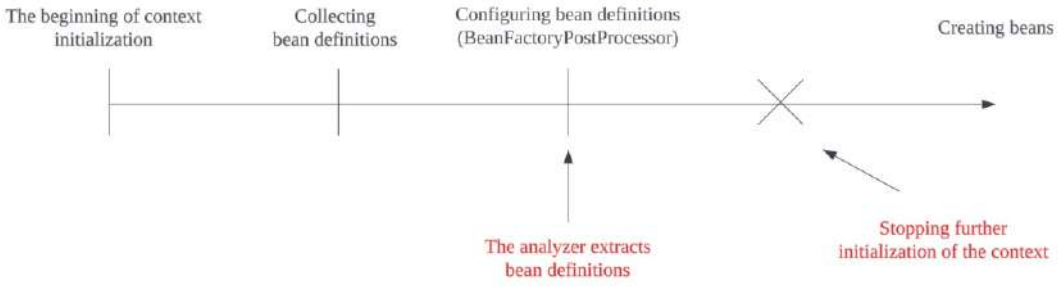


Fig. 1. The initialization stages of the Spring context

4.2 Modernization of symbolic engine

The core idea of the symbolic engine modernization is to change the mechanism used for selecting symbolic object types. Whereas the symbolic engine previously selected an arbitrary type, determined by the SMT solver as satisfying the symbolic path constraints, it now tries to consider only those types that are used in the application configuration chosen for test generation.

Let's discuss the example of test generation for the *getSpecies()* method from the *SpeciesService* class, presented in Section 2. In the past, when generating tests, the *Dog* implementation of the *Animal* interface could be chosen because it satisfied the symbolic path constraints. However, tests using the *Dog* class were not particularly useful because they did not test the actual behavior of the program. In contrast, now, the *Cat* class is deterministically chosen as it is specified in the application configuration.

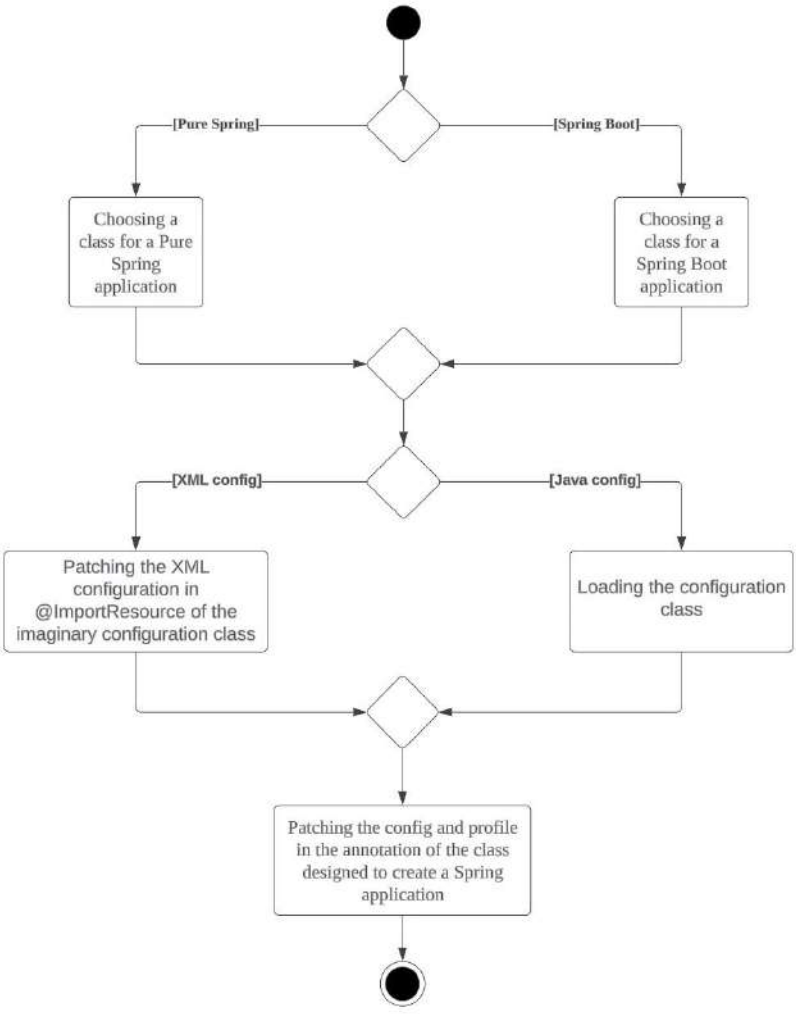


Fig. 2. The activity diagram of creating a Spring application on the UnitTestBot Java side

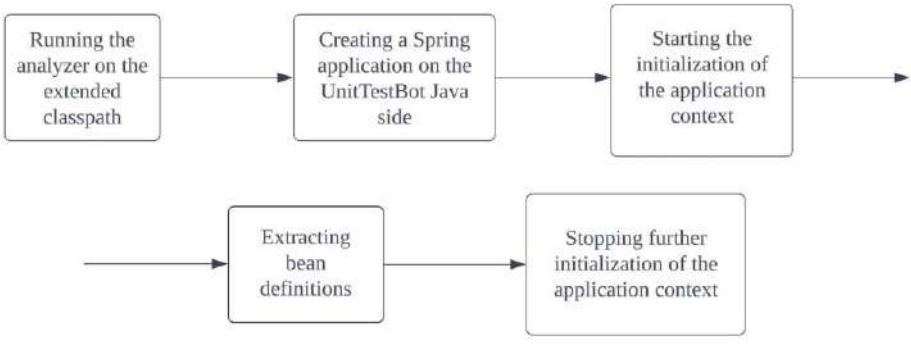


Fig. 3. The Spring-analyzer's work scheme

4.3 Integration of symbolic engine and configuration analyzer

The Spring configuration analyzer is launched in a separate process from the symbolic engine. Firstly, this allows running the configuration analyzer on the extended classpath without any difficulties. After that, it prevents a possible crash of the entire test generation in case of an error during the analysis of custom configurations. For example, running a Java Spring application based on a custom one on the *UnitTestBot* side may cause the JVM crash if the custom application is poorly designed.

The symbolic engine and user configuration analyzer processes, like all other processes in *UnitTestBot Java*, communicate using the *RD*[19] framework.

5. Results

As a result of this research, we managed to propose an approach to the analysis of Spring application configurations, which sometimes allows generating tests that correspond better to the actual behavior of the program. The developed configuration analyzer was integrated into the well-known tool of automated test generation *UnitTestBot Java*.

In particular, the modernized tool is able to generate a test that checks the actual execution of the `getSpecies()` method from the `SpeciesService` class of the running example given in Section 2. To go further, we also provide tests generated with the modernized tool on real open-source projects *Java Blog Aggregator: Boot*[20] and *Mall*[21].

5.1 Java Blog Aggregator: Boot

We paid attention to the *Java Blog Aggregator: Boot* project because it is mentioned, for example, in the article [11] as one of the recommended projects to study for beginners in the Spring framework.

Consider the `AllCategoriesService` class that has the autowired `CategoryService` field.

```
@Service
public class AllCategoriesService {

    @Autowired
    private CategoryService categoryService;

    public Integer[] getAllCategoryIds() {
        List<Category> categories = categoryService.findAll();

        Integer[] result = new Integer[categories.size()];

        for (int i = 0; i < categories.size(); i++) {
            result[i] = categories.get(i).getId();
        }

        return result;
    }
}
```

[19] RD: Reactive Distributed communication framework for .NET, Kotlin and C++ (experimental). Inspired by JetBrains Rider IDE. Available: https://github.com/JetBrains/rd

[20] Java Blog Aggregator: Boot. Available: https://github.com/jirkapinkas/java-blog-aggregator-boot

[21] Mall. Available: https://github.com/macrozheng/mall

The signature of the `CategoryService` class has the form:

```
@Service
public class CategoryService
```

The `CategoryService` class has the annotation `@Service`, which means that this class is used to define a bean.

As a result of test generation using the modernized *UnitTestBot Java* for the `getAllCategoryIds()` method from the `AllCategoriesService` class, we get a test:

```
@Test
public void testGetAllCategoryIds() throws Exception {
    AllCategoriesService allCategoriesService
            = new AllCategoriesService();

    CategoryService categoryService = new CategoryService();

    CategoryRepository categoryRepositoryMock
            = mock(CategoryRepository.class);

    when(categoryRepositoryMock.findAll())
            .thenReturn(new ArrayList<>());

    setField(categoryService,
             "cz.jiripinkas.jba.service.CategoryService",
             "categoryRepository",
             categoryRepositoryMock);

    setField(allCategoriesService,
             "cz.jiripinkas.jba.service.AllCategoriesService",
             "categoryService",
             categoryService);

    Integer[] actual = allCategoriesService.getAllCategoryIds();

    assertEquals(0, actual.length);

    assertEquals(new Integer[0], actual);
}
```

Instead of mocking the `CategoryService` class, its concrete implementation is used in this test. It makes the test more expressive. In particular, we can observe that the `CategoryService` interacts with the database. Also, the user can adjust the behavior of the mock related to database access if necessary.

5.2 Mall

We also generated tests for the *Mall* project, which is very popular, having over one hundred thousand forks and stars on GitHub. Let's discuss the test for the `delAdmin()` method in the `UmsAdminCacheServiceImpl` class, that the modernized *UnitTestBot Java* has generated based on Spring application configuration analysis.

```
@Service
public class UmsAdminCacheServiceImpl
        implements UmsAdminCacheService {
    @Autowired
    private UmsAdminService adminService;
    @Autowired
    private RedisService redisService;
    ...
    @Override
    public void delAdmin(Long adminId) {
        UmsAdmin admin = adminService.getItem(adminId);

        if (admin != null) {
            String key = REDIS_DATABASE + ":" + REDIS_KEY_ADMIN
                    + ":" + admin.getUsername();

            redisService.del(key);
        }
    }

    ...
}
```

This class has two autowired fields: `UmsAdminService` and `RedisService`, which have corresponding beans in the application configuration.

The test generated for the `delAdmin()` method of the `UmsAdminCacheServiceImpl` class is as follows:

```
@Test
public void testDelAdmin() throws Exception
{
    UmsAdminCacheServiceImpl umsAdminCacheServiceImpl
            = new UmsAdminCacheServiceImpl();

    UmsAdminServiceImpl adminService = new UmsAdminServiceImpl();

    UmsAdminMapper adminMapperMock = mock(UmsAdminMapper.class);
    when (adminMapperMock.selectByPrimaryKey(any()))
            .thenReturn(null);

    setField(adminService,
             "com.macro.mall.service.impl.UmsAdminServiceImpl",
             "adminMapper",
             adminMapperMock);

    setField(umsAdminCacheServiceImpl,
           "com.macro.mall.service.impl.UmsAdminCacheServiceImpl",
           "adminService",
           adminService);

    umsAdminCacheServiceImpl.delAdmin(null);
}
```

In this test, instead of mocking the abstract type `UmsAdminService`, its concrete implementation `UmsAdminServiceImpl` is substituted according to the application configuration. Initialization of the second autowired field did not occur because it is not required in the tested program execution path. Although there are no assertions in this test because the method has `void` return type, it is still valuable. Since the tested method takes a nullable value as an argument, a scenario in which `adminId` is `null` is possible and is a kind of edge case that often causes `NullPointerException`. The generated test ensures that no such exception actually occurs in the method under test. When writing tests manually, similar scenarios are often not taken into account.

6. Future work

Unit tests are often used to verify the logic of Spring application components, so high-quality automatic generation of such tests is important. However, some bugs can only be detected by integration and end-to-end tests that interact with real data storage and other microservices, as well as take into account the diverse features of the Spring framework (e.g., authorization and authentication). For this reason, developing an integration test generation tool is a prominent direction for future work. Such a tool will likely also need to initialize a modified Spring application, meaning that the "hybrid" Spring application starter developed in this work may find additional uses.

References

[1]. Cristian C., Daniel D., Dawson E. KLEE: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs. Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (OSDI'08) USA: USENIX Association, 2008, pp. 209-224.

[2]. Workshop on Search-Based and Fuzz Testing, Available at: https://sbft24.github.io, accessed 24.06.2024.

[3]. Ivanov D. et al. UTBot Java at the SBST2022 Tool Competition. 2022 IEEE/ACM 15th International Workshop on Search-Based Software Testing (SBFT), 2022, pp. 39-40. DOI: 10.1145/3526072.3527529.

[4]. Ivanov D., Menshutin A., Pelevin M. et al. UTBot at the SBFT 2023 Java Tool Competition. 2023 IEEE/ACM International Workshop on Search-Based and Fuzz Testing (SBFT), 2023, pp. 68-69. DOI: 10.1109/SBFT59156.2023.00019.

[5]. Java Programming – The State of Developer Ecosystem in 2022 Infographic, JetBrains: Developer Tools for Professionals and Teams, Available at: https://www.jetbrains.com/lp/devecosystem-2022/, accessed 24.06.2024.

[6]. Fadatare R. Spring Boot Unit Testing Service Layer using JUnit and Mockito, Available at: https://www.javaguides.net/2022/03/spring-boot-unit-testing-service-layer.html, accessed 24.06.2024.

[7]. Overview: Spring Framework, Available at: https://docs.spring.io/spring-framework/reference/testing/spring-mvc-test-framework/server.html, accessed 24.06.2024.

[8]. Chathuranga S. Unit and Integration Testing in Spring Boot Micro Service, Available at: https://salithachathuranga94.medium.com/unit-and-integration-testing-in-spring-boot-micro-service-901fc53b0dff, accessed 24.06.2024.

[9]. Kim M., Xin Q., Sinha S., Orso A. Automated test generation for REST APIs: no time to rest yet. Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2022), Association for Computing Machinery, New York, USA, 2022, pp. 289-301. DOI: 10.1145/3533767.3534401.

[10]. Baldoni R., Coppa E., Cono D'elia D., Demetrescu C., Finocchi, I. A Survey of Symbolic Execution Techniques. ACM Comput. Surv. 51, 3, Article 50, 2019. 39 p. DOI: 10.1145/3182657.

[11]. Fadatare R. 10+ Free Open Source Projects Using Spring Boot, Available at: https://www.javaguides.net/2018/10/free-open-source-projects-using-spring-boot.html, accessed 24.06.2024.

Информация об авторах / Information about authors

Кирилл Александрович ШИШИН – студент кафедры системного программирования СПбГУ. Сфера научных интересов: задачи статического анализа кода, автоматическая генерация тестов.

Kirill Alexandrovich SHISHIN – student of the software engineering department of SPbU. Research interests: static code analysis tools, automated tests generation.

Илья Владимирович МУРАВЬЁВ – инженер-исследователь кафедры системного программирования СПбГУ. Сфера научных интересов: задачи статического анализа кода, автоматическая генерация тестов, контекстно-свободная достижимость.

Ilia Vladimirovich MURAVEV – researcher of the software engineering department of SPbU. Research interests: static code analysis tools, automated tests generation, context-free grammars.

Егор Константинович КУЛИКОВ – кандидат физико-математических наук, доцент кафедры системного программирования СПбГУ. Сфера научных интересов: задачи статического анализа кода, автоматическая генерация тестов; методы локальной аппроксимации и их распараллеливание.

Egor Konstantinovich KULIKOV – Cand. Sci. (Phys.-Math.), associate professor of the software engineering department of SPbU. Research interests: static code analysis tools, automated tests generation; local approximation methods and their parallelization.

DOI: 10.15514/ISPRAS-2024-36(2)-6

Application of Generative Artificial Intelligence for Risk Management of Software Projects

[1] *A.D. Dzheiranian*, ORCID: 0009-0000-8916-2855 <*ADDzheyranyan@edu.hse.ru*>
[1,2] *M.A. Plaksin*, ORCID: 0000-0002-6288-8610 <*MAPl@list.ru*>

[1] *HSE University,*
38, Studencheskaia St., Perm, 614070, Russia.
[2] *Perm State University,*
15, Bukirev St., Perm, 614990, Russia.

Abstract. The article highlights an innovative approach to risk management in software projects using generative artificial intelligence. It describes a methodology that involves the use of publicly available chatbots to identify, analyze, and prioritize risks. The Crawford method is used as a basis for risk identification. The authors propose specific formulations of requests to chatbots (instructs, prompts) that facilitate obtaining the necessary information. The effectiveness of the methodology has been demonstrated on five small software projects and a dozen of economic and organizational projects of significantly different scales, from small to federal. This confirms its applicability and practical value.

Keywords: generative artificial intelligence; chatbot; risk management; risk identification; Crawford cards; software projects.

For citation: Dzheiranian A.D., Plaksin M.A. Application of generative artificial intelligence for risk management of software projects. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 73-82. DOI: 10.15514/ISPRAS-2024-36(2)-6

Применение генеративного искусственного интеллекта для управления рисками программных проектов

[1] *А.Д. Джейранян,* ORCID: 0009-0000-8916-2855 <*ADDzheyranyan@edu.hse.ru*>
[1,2] *М.А. Плаксин*, ORCID: 0000-0002-6288-8610 <*MAPl@list.ru*>

[1] *Национальный исследовательский университет «Высшая школа экономики», Россия, 614070, г. Пермь, ул. Студенческая, д. 38.*
[2] *Пермский государственный национальный исследовательский университет, Россия, 614990, г. Пермь, ул. Букирева, д. 15.*

Аннотация. Статья освещает новаторский подход к управлению рисками в программных проектах с использованием генеративного искусственного интеллекта. Описывается методика, которая включает в себя применение общедоступных чат-ботов для выявления, анализа и приоритизации рисков. В качестве основы для выявления рисков используется метод Кроуфорда. Авторы предлагают конкретные формулировки запросов к чат-ботам (инстрактов, промптов), способствующие получению необходимой информации. Эффективность методики продемонстрирована на пяти небольших программных проектах и дюжине проектов экономических и организационных, причем существенно разного масштаба, от малых до федеральных. Это подтверждает ее применимость и практическую ценность.

Ключевые слова: генеративный искусственный интеллект; чатбот; управление рисками; идентификация рисков; карточки Кроуфорда; программные проекты.

Для цитирования: Джейранян А.Д., Плаксин М.А. Применение генеративного искусственного интеллекта для управления рисками программных проектов. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 73–82 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–6.

1. Introduction

The object of the proposed research is a generative model of artificial intelligence (GAI). The subject is the use of GAI for risk management of software projects.

The purpose of the research is to study the possibility of using GAI to manage risks arising during the implementation of software projects.

The study has the following results:

1) A conclusion on the possibility of using GAI in risk management of software projects has been provided.
2) A methodology for using GAI has been developed, and a sequence of prompts that can be used for this aim has been proposed.
3) A comparison of several publicly available GAI chatbots has been conducted.

The relevance of the research is determined by two factors:

1) In the modern world, neural networks are becoming an integral part of everyday life. Generative artificial intelligence stands out as a powerful tool capable of significantly simplifying and accelerating the solution of many tasks, the scope of which is yet to be defined. At the same time, the use of GAI tools fundamentally differs from the use of all other software systems. All other – "traditional" – software systems are initially created to solve some specific tasks. A "traditional" computing system behaves like an automaton, controlled by a set of commands known in advance (although it can be quite complex). There are instructions that describe how exactly to control such an automaton. The use of these software tools is stable in the sense that the same human actions cause the same reaction and lead to the same results. In contrast to "traditional" software systems, the behavior of GAI is extremely unstable. Repeating the same human actions can lead to very different consequences. There is no instruction that would define in advance what exactly should be done with GAI systems to achieve this or that result. Humanity has invented a new entity that behaves in some independent way and communication with which is yet to be learned.
2) Risk management is an important part of information system development management. To assess the relevance of improving risk management methods in informatics, it is enough to look at the results of research on the success of software projects [1]–[5]. The success statistics of software development projects can be roughly represented as a (slightly skewed) normal distribution curve. At one end will be successful projects, i.e., those that were completed on time, stayed within the planned budget, and implemented all the declared capabilities. The share of such projects according to the results of various studies fluctuates around 25-35%. On the other side will be projects that failed and were never completed. There are about 20% of such projects. In the middle are "controversial" projects that were completed, but either exceeded the planned deadlines, did not fit into the budget, or did not implement all the planned capabilities. The share of such projects is about half. This picture changes somewhat in studies of different years. Especially if the study takes into account the size of the project. But overall, it remains quite stable.

The low level of success indicates that the management of most projects is not able to identify all the risks threatening the project and react correctly to them. Risk management remains more of an art than a craft. The identification and analysis of risks are largely the result of using expert methods,

such as the Crawford method [6], affinity diagrams [7], Ishikawa diagrams [8], fail stories, diversion analysis [9], rough models (Fermi method) [10], etc. In computer science, there are no yet accepted standardized methods, similar to the FMEA method [11], which has proven itself in mechanical engineering and several other industries. Therefore, any research aimed at turning risk management of software projects from art into technology is considered useful.

2. The MSF risk management process

This study uses a risk management methodology that is a part of the Microsoft Solution Framework for Agile (MSF) technology [12].

According to MSF, the continuous risk management process consists of six stages:

1) Identifying risks.
2) Analyzing and prioritizing risks.
3) Planning risks (assigning prevention and response plans for risks and comparing different plans for the same risk).
4) Monitoring risks.
5) Implementing risk responses (adjusting the project in response to realized risks).
6) Learning lessons (learning about risks).

We will be interested in the first three stages: risks identification, the analysis and prioritization of risks, and developing plans.

MSF defines risk as "any event or condition that can have a positive or negative impact on the project outcome". Risk is characterized by the probability of a risk event occurring and the impact that the fact of this event will have on the project. To combine these two characteristics together, the concept of risk magnitude is introduced. The risk magnitude is calculated as the product of the risk probability and its impact.

MSF technology provides for the construction of two plans (a risk prevention plan and a risk response plan):

- the prevention plan includes actions aimed at reducing the likelihood of a risk occurring and reducing its potential threat to an acceptable level. These actions must be performed in advance;
- the mitigation plan includes responsive actions that need to be taken if the risk could not be prevented and the risk event occurred. This plan is put into action if a certain predetermined condition (risk trigger) is met.

3. Requirements and project selection

It was necessary to select software projects to conduct the research. There were only two requirements for the projects: they must have a brief description and they must be noticeably different from each other. The brevity of the description was determined by the quantitative limitations of the listed chatbots. The need for noticeable differences in projects was dictated by the fact that we were interested in the extent to which the GAI chatbots would consider the features of a particular project in their recommendations.

Thus, the study examined five small software projects:

- "Designing data visualization tools based on a language-oriented approach";
- "Trade and information software system for managing vending machines";
- "System for printing photographs and magnets from company Lomobil";
- "Mobile application "Virtual parking";
- "Searching for optimal advertising for business according to specified parameters".

The selection of projects for the study was conducted almost by chance ("they happened to be at hand"). Two descriptions were compiled for the projects: a brief one and a more detailed one. It turned out that the level of detail of the project description plays a role. In the case of more detailed descriptions, the chatbots used information about the project details in their responses. As a result, chatbot responses were more exact and meaningful.

4. Methodology of using GAI for risk management

4.1 Overview of the Crawford's method

The Crawford method [6] is used to identify risks. This is a modification of brainstorming in which special measures for avoiding the anchoring effect are taken. In real life it looks like this.

A group of experts (7-10 people) is formed. Each of them receives a pack of numbered cards. On the first card, each expert writes down the risk that he considers the most important for the project being analyzed. The presenter collects the completed cards.

After this, on the second card, each expert writes down the most important of the remaining risks (the second most important risk). The completed cards are collected again by the presenter. And so on for a specified number of times.

After this, risks are discussed and grouped. Different experts may describe the same risk in different words. These facts are revealed during the discussion. Same and similar risks are grouped. Physically, this is expressed in the formation of so-called affinity diagrams (the grouped cards are fastened with tape into a vertical strip) [7]. The length of the tape clearly demonstrates the importance of this risk from the point of view of the expert community. First, all risks of the first rank are discussed, then all risks of the second rank, etc. As a result, a set of risks is formed and sorted by importance. The importance of a risk is determined by the number of times that risk is mentioned by different experts.

4.2 Assigning a chatbot a role

The GAI chatbot acts as the manager of project that needs to be analyzed. The first instruction looks like: "You are an experienced project manager. You have been entrusted with the leadership of the next project…" The following is a description of the project.

4.3 Providing a description of the project

The project description can be given at different levels of detail. The level of detail plays a role. The more detailed the project is, the more details the "experts" will be able to use when evaluating it.

4.4 Formation of a group of experts

Experts are generated by the chatbot as members of a team, which it manages as a project manager.

For each expert, his specialization is indicated. The composition of the team of experts is determined by the subject area to which the project relates. For software projects, these were business analysts, programmers, testers (quality assurance specialists), "human-computer" interaction specialists, and logisticians (specialists in the deployment of software systems). The specifics of the project may require the involvement of particular experts from other fields.

The qualifications of an expert can be described at different levels. One may limit oneself to phrases such as "experienced professional", "has work experience of more than ten years" (or vice versa "less than three years"). Alternatively, one can provide a detailed listing of the expert's knowledge, skills, certificates, etc.

NB! A more detailed description, as a rule, does not provide any benefit compared to a brief description such as "more than ten years of work experience". There is a fundamental difference

with the project description. There the detail of the description plays a role, here it does not. The reason is unknown.

To obtain a detailed description of the expert's qualifications, one can use the same chatbot or another. It is enough to give it a request: "You are the project manager. You need to hire three business analysts: a junior, a middle and a senior. List what knowledge, skills and abilities each of them should have". But – we repeat once again – we did not find any sense in such detail.

An instruction for generating a team of experts may look, for example, like this: "Your team consists of experts: two business analysts, two programmers, two economists, two lawyers. The first expert in each pair has at least ten years of experience, the second – no more than three".

Experts can be given names: "Anna is an experienced business analyst", "Boris is an experienced programmer", etc. After that, one can call them by name: "Let Anna do this", "Let Boris do this", etc. This can be useful since one have to contact the same expert several times.

4.5 Survey of experts

After describing the experts, they are interviewed according to the following scheme:

1) "Let the expert name the characteristics of the project that seem significant to him".
2) "Let the expert name the risks of project implementation that are associated with these characteristics and justify his opinion".
3) "Let the expert evaluate the probability of the risk occurring on a given scale and the damage from the realization of the risk on a given scale" (at the same time, specific numbers in the chatbot's responses cannot be trusted).
4) "Let the expert name the means to reduce the likelihood of the risk occurring and to reduce the damage if it occurs".
5) "Let the expert name indicators by which it can be judged that the danger of this risk occurring is increasing" (it has not yet been possible to get a good answer to this question from the chatbot).

4.6 Grouping of risks

Since in Crawford's method the importance of a risk is determined by the number of times it was named by experts, one can ask the chatbot to classify risks by similarity of wording: "Group similar risks. Indicate which expert named each risk and how exactly it was formulated".

5. Interesting points in building a methodology of using GAI for risk management

Next, some interesting aspects of communicating with GAI chatbots are discussed.

5.1 Ways to organize a collective survey of experts

The Crawford card method is a method of collective examination. The question about how to organize the collective work of a group of experts using one GAI chatbot arose. One chatbot had to speak on behalf of several (in our case, first nine, and then ten) experts. A contradiction arose: it was necessary to communicate with one chatbot and at the same time communicate with ten experts.

This contradiction could be resolved in three ways:

- division in time;
- division in space;
- division in structure.

The time separation implied that the dialogue with the chatbot would be repeated several times. Moreover, each time the bot will be given the task of presenting itself as a new expert. The spatial

separation meant that the dialogue would be run on several different computers with multiple instances of the bot. It was not clear whether these several instances would behave differently "on their own" or whether each of them would have to be given some kind of input. The division in the structure assumed that a certain structure would be built. That structure would allow to communicate with one interlocutor (assign only one role to the chatbot), but at the same time receive answers from several different experts.

The latter option was attractive due to its efficiency. It was the one chosen. We took advantage of the fact that the examination process is conducted collectively, and this process has a leader. It was the presenter who was chosen as our interlocutor. He was asked to carry out the entire procedure "inside the chatbot" and immediately provide us with its results.

5.2 A way to achieve a variety of expert opinions

After this, the question arose about the formation of an "expert group". Initially, the request simply referred to "experts" without any specification. But this approach did not provide a diversity of opinions. The answers given by different experts were often repeated (more precisely, the first three experts named nine different risks, and all the remaining experts repeated the same nine risks in different combinations). And the basis of the Crawford method is precisely that different experts will have different views on the same problem. An attempt was made to solve this problem head-on. The chatbot was asked to name thirty different risks (more precisely, the request was supplemented with the following requirement: "Consider the fact that each expert has different professional experience and knowledge. During brainstorming, they generate a variety of ideas. That is, risk formulations cannot be the same among experts, but may imply one and same risk"). Which is what the bot did. And thus, again came into conflict with Crawford's method. According to Crawford, the importance of a risk is determined by how many experts name it. That is, it is impossible to prohibit different experts from calling the same risk (possibly describing it in different words). But each expert had to name the risks in accordance with his expert vision.

The next solution was found. We abandoned the idea of "experts in general" and decided to clarify the specialization and experience of each expert. As a basis for determining the nomenclature of experts, a development team based on the Microsoft Solution Framework for Agile technology [12] was taken. For greater variety, the work experience of each expert has been added.

As a result, the beginning of the request to the chatbot took the form: "You are the manager of a software project that is planned to be implemented. It is necessary to identify the risks that exist during its implementation. For this you have a group of 10 experts. Two of them are business analysts, two are programmers, two are QA specialists, two are user experience specialists, two are software product deployment specialists. The first specialist in a pair is experienced (at least 10 years of work experience), the second is a beginner (up to three years of work experience). Let each expert name the characteristics of the project that seem important to him. The following is a description of the project".

5.3 Instability of responses

In ordinary consciousness, a computer is perceived as an automaton that operates according to a certain algorithm, converting input data into output data. Under these conditions, it is logical to expect that the same answer will be given to the same request (or – taking into account the experience of searching for information on the Internet – the answer received when repeating the request will be close to the answer to the first request).

It turned out that this is not the case for the GAI. The answer received when repeating the same request could differ significantly both in content (the same experts began to list other risks) and in form. For example, a couple of times the YandexGPT [13] responded to a request with the following phrase: "I'm not in the mood today. Come back another time!"

5.4 Chatbots are confident in their knowledge

GAI chatbots are sure in their know-it-all nature. In several dialogues, a following sentence was added to the request: “If any expert does not have enough information to make a decision, let him ask for the missing information”. None of the chatbots responded to this proposal. There was always enough information for all the “chatbot experts”.

A traditional expert system is able to understand that it is missing some information and request it. A search engine (a la Google) may indicate that during a search it did not find certain words from the query or did not find an answer to the question at all. GAI chatbots are confident that they have all the necessary knowledge to answer any request. With the exception of the boundaries of political correctness that are unacceptable from the point of view of the chatbots’ authors (but this is another aspect of GAI).

5.5 Loss of dialogue context

Theoretically, each chatbot undertakes to remember a certain number of previous replicas and respond to the next replica, considering this context (e.g., one session of dialogue with a BING AI [14] can consist of no more than 30 requests). In practice this is not always the case. Such problems arose especially often when communicating with ChatGPT [15].

Related to the issue of loss of context is the issue of the size of the information provided by the chatbot. The output volume of all chatbots is limited. Some chatbots can request “continued issuance”. Some of them must be asked to do this “manually”. At this moment, a loss of context and a violation of the form of information delivery often occurs. To bypass these restrictions, it makes sense to change the request to the chatbot and split it into several replicas.

5.6 The degree to which project specifics are reflected in risks

Links that reflect the features of the project are quite rare. Here is an example for a project “Designing data visualization tools based on a language-oriented approach” (quoted from a conversation with BING AI, risks identified by a developer, links to project features are in italics):

1) “Lack of time to implement the project. (Rationale: The project is complex and takes a long time to develop, especially since there is only one person on the team)”.
2) “Lack of experience in working with domain-specific languages. (Rationale: Working with domain-specific languages requires specialized skills and knowledge that may be in short supply)”.
3) “Difficulty in creating interactive chart customization. (Rationale: Creating interactive charts is a complex task that requires a deep understanding of user needs and data visualization technologies)”.

But from the point of view of “area fire” chatbots work well. As a criterion, we took the classic list of risks of software projects [16]. It turned out that the GAI closes this list by 70%.

In one of the experiments, risk identification was carried out first by people “manually”, and then using a chatbot. The relationship turned out to be as follows. A total of 14 risks were identified: 9 were identified by both humans and the chatbot, 1 by only humans, 4 by only the chatbot.

5.7 Correctness of risk grouping

How correctly does the GAI group the responses of different experts? Differently. The association was purely formal, almost meaningless. There were collections of risks that were identical in meaning, but with different names. There also was a grouping of risks associated with one problem but understood by different experts from different points of view. This is another example of chatbot instability.

6. Chatbot Overview

In this study, four publicly available chatbots were used as GAI tools: YandexGPT 2 [13], BING AI (Copilot) [14], ChatGPT [15], and GigaChat [17]. All of these are capable of generating human-like text based on context and past conversations.

As a result of the analysis, no significant differences were found among the chatbots in terms of the tasks we solve. Subjectively, BING AI appeared to be more effective, followed by ChatGPT in second place, YandexGPT in third, and GigaChat in last place. Therefore, further research will be based mainly on the example of BING AI.

Nevertheless, the reader may wonder why the most popular ChatGPT is not ranked first. The reason is that it often lost context in our experiments. In contrast, BING AI showed more stability in its responses, possibly due to the use of the exact generation mode. However, it should be emphasized again that this rating is subjective. Under other conditions, the opinion could be different, as bots show variability, instability in their answers, which also leads to the impossibility of creating objective criteria.

7. Evaluation of Approach effectiveness

The main limitations that must be considered when applying GAI (both in risk management and in any other area):

1) GAI does not understand the meaning of the task proposed to it. Its answers are the result of some probabilistic algorithms. This is important to keep in mind, since during a dialogue with a chatbot it is easy to create an illusion of the reasonableness of some of its answers. The probabilistic nature of the GAI's responses leads to very high instability of its work. The same human actions can generate completely different reactions from the chatbot.
2) In particular, this means that the effectiveness of using any communication methods with GAI chatbots is not guaranteed.
3) GAI knowledge is very limited (despite trillions of parameters and a phenomenal number of texts loaded into the neural network). In reality, a person receives a huge amount of information in non-verbal form. This information is not available to neural networks.
4) GAI is not aware of the boundaries of its knowledge and ignorance. It will never understand that it doesn't know something. A chatbot is sure that it knows everything.
5) GAI is not able to rationally explain the answers it produces.
6) Moreover, GAI can give completely incorrect answers ("hallucinate"), but categorically insist on their truth.
7) GAI has some information "in general" but does not have information about each specific project. This means that the neural network is not able to consider the specifics of each certain project. More precisely, this accounting depends on three factors:
    - the details of the project description that a person will provide to the chatbot;
    - the size of the context that the chatbot takes into account when preparing a response;
    - the probabilistic (i.e., unpredictable) nature of developing an answer. Therefore, the chatbot's responses can be perceived in terms of "area fire", but not in terms of describing a specific situation.

The main advantages of GAI in terms of identifying and analyzing risks:

1) Extensive knowledge in a given subject area (taking into account the lack of understanding of the specifics of a particular project, a specific situation, and the fact that this knowledge still needs to be able to be extracted). Experience has shown that chatbots do a good job of identifying risks at a "fundamental level".

2) Extensive knowledge of subject areas related to the project. Opportunity to evaluate the project from different points of view.
3) The ability to quickly involve many experts of different specializations in the work and obtain group expertise of the project.
4) The methodology, originally developed for software projects, was adapted to apply it to economic and organizational projects of significantly different scales, from small to federal. To assess the latter, indicators of national security of the Russian Federation were used. The transfer was successful [18]. The technique works in other subject areas too. Naturally, each field requires its own set of experts.

8. Conclusion

During the study, the use of GAI in the process of risk management of software projects was tested. The study demonstrated the fundamental suitability of the GAI for this task, despite the disadvantage of operational instability. However, the authors have not yet ensured that the GAI takes the features of a particular software project seriously enough into account. It is more about "area fire". And the GAI copes well with this and is not inferior to other sources of risks information.

A methodology for organizing a group examination, simultaneous survey of a "group of GAI-experts" using the GAI chatbot was proposed. The article provided the texts of the relevant requests and examples of their execution. Their advantages and disadvantages were discussed.

In addition, the proposed methodology turned out to be suitable not only in relation to software projects, but also in other subject areas (e.g., risk assessment of economic projects).

Directions for further research have been identified:

- to ensure that the specifics of the projects are taken into account when identifying risks;
- to improve the virtual "community of experts";
- to ensure that the expert provides a verifiable justification for the likelihood and impact of the risk;
- to improve the quality of risk indicators proposed by the experts;
- to continue expanding the proposed methodology to other subject fields.

References

[1]. Prabhakar R., Seetharamaiah P. Organizational Strategies and Social Interaction Influence in Software Development Effort Estimation. IOSR Journal of Computer Engineering, 2014, vol. 16, no. 2, pp. 29-40.
[2]. Quan Y., Zi-quan L. Research on Risk Evaluation and Risk Optimization of IT Projects. 2008 International Conference on Information Management, Innovation Management and Industrial Engineering, Taipei, Taiwan, 2008, pp. 119-123. DOI: 10.1109/ICIII.2008.270.
[3]. Stern R., Arias J.C. Review of Risk Management Methods. Business Intelligence Journal, 2011, vol. 4, no. 1, pp. 51-78.
[4]. Andriole S.J. Editorial: Where did IT go. International Journal of Technology Management, 2022, vol. 89, no. 1/2, pp. 1-8.
[5]. Khatavakhotan A.S., Ow S.H. Improving IT Risk Management Process by an Embedded Dynamic Verifier Core: Towards Reducing IT Projects Failure. 2012 Third International Conference on Intelligent Systems Modelling and Simulation, Kota Kinabalu, Malaysia, 2012, pp. 684-687. DOI: 10.1109/ISMS.2012.47.
[6]. Куликов Ю. Метод Кроуфорда. / Kulikov U. The Crawford Method (in Russian). Available at: http://pmpro.ru/metod-krouforda.html, published 19.12.2010.
[7]. Affinity Diagram – Kawakita Jiro or KJ Method. Available at: https://project-management.com/affinity-diagram-kawakita-jiro-or-kj-method/, published 29.01.2017.
[8]. Ishikawa K. What is Total Quality Control? The Japanese Way. Prentice Hall, London, 1985, 215 p.
[9]. Anticipatory Failure Determination (AFD). Available at: https://www.whereinnovationbegins.net/, accessed 25.12.2023.
[10]. Poundstone W. Are You Smart Enough to Work at Google? Little, Brown Spark, New York, 2012, 290 p.

[11]. Mikulak R.J., McDermott R.E., Beauregard M.R. The Basics of FMEA. Productivity Press, New York, 1996, 76 p.
[12]. Microsoft Corporation. Microsoft Solutions Framework MSF Risk Management Discipline. Seattle, 2002, vol. 1.1., 54 p.
[13]. Chatbot YandexGPT 2. Available at: https://yandex.ru/project/alice/yagpt, accessed 02.03.2024.
[14]. Chatbot Bing AI. Available at: https://www.bing.com/, accessed 02.03.2024.
[15]. Chatbot ChatGPT. Available at: https://gpt-chatbot.ru/, accessed 02.03.2024.
[16]. Boehm B.W. Software Risk Management: Principles and Practices. IEEE Software, 1991, vol. 8, no. 1, pp. 32-41.
[17]. Chatbot GigaChat. Available at: https://developers.sber.ru/gigachat/login, accessed 02.03.2024.
[18]. Джейранян, А.Д., Плаксин М.А. Применение генеративного искусственного интеллекта для оценки рисков и безопасности федеральных проектов. Международная научно-методическая конференция Инженерное образование в цифровом обществе, Минск, Беларусь, 2024 г., ч. 2, стр. 294-298. / Dzheiranian A.D., Plaksin M.A. The use of generative artificial intelligence to assess risks and security of federal projects. International scientific and methodological conference Engineering education in a digital society, Minsk, Belarus, 2024, vol. 2, pp. 294-298 (in Russian).

Информация об авторах / Information about authors

Анна Даниеловна ДЖЕЙРАНЯН – студент бакалавриата Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь), образовательная программа «Программная инженерия». Сфера научных интересов: анализ и визуализация данных, генеративные модели, предметно-ориентированное моделирование.

Anna Danielovna DZHEIRANIAN – undergraduate student at the National Research University – Higher School of Economics (HSE University, Perm Branch), educational program “Software Engineering”. Research interests: data analysis and visualization, generative models, domain-specific modeling.

Михаил Александрович ПЛАКСИН – кандидат физико-математических наук, доцент кафедры информационных технологий в бизнесе Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь), доцент кафедры математического обеспечения вычислительных систем государственного национального исследовательского университета (ПГНИУ). Сфера научных интересов: системный анализ, ТРИЗ, управление проектами, информатизация образования, преподавание информатики в начальной школе, построение сквозного курса информатики с 1-го по 11-й класс.

Mikhail Alexandrovich PLAKSIN – Cand. Sci. (Phys.-Math.), Associate Professor of the Department of Information Technologies in Business of the National Research University – Higher School of Economics (HSE University, Perm Branch), Associate Professor of the Department of Computer Science of the Perm State National Research University (PSU). Research interests: system analysis, TRIZ, project management, informatization of education, teaching informatics in elementary school, construction of a cross-curricular informatics course from 1st to 11th grade.

DOI: 10.15514/ISPRAS-2024-36(2)-7

Data-Driven Approach to Curriculum Analysis

Iu. Nasu, ORCID: 0009-0005-3852-0022 <*yunas2002@iCloud.com*>
M.S. Drobinin, ORCID: 0009-0002-0704-3532 <*msdrobinin@yandex.ru*>
M.S. Efanov, ORCID: 0009-0003-5824-3259 <*msefanovv@gmail.com*>
V.V. Lanin, ORCID: 0000-0002-0650-2314 <*vlanin@hse.ru*>

HSE University,
38, Studencheskaia St., Perm, 614070, Russia.

Abstract. The choice of an educational program is momentous in young people's lives. Given the shortage of time after exams, applicants usually do not have time to analyze possible educational tracks. Furthermore, it requires a thorough study of learning plans. This research addresses the problem proposing the algorithm to data-driven curriculum analysis based on natural language processing of course names or competences listed in learning plans. Moreover, the intelligent software system architecture is described. The method is tested on the curricula scraped from university websites. In order to store the content a data warehouse has been developed. At this time, there are few studies on this topic. The existing ones are either on the early stages of development or scarce on implementation details. They are briefly discussed in this paper.

Keywords: competences, curricula data analysis, NLP, data warehouse.

For citation: Nasu Iu., Drobinin M.S., Efanov M.S., Lanin V.V. Data-Driven Approach to Curriculum Analysis. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 83-90. DOI: 10.15514/ISPRAS-2024-36(2)-7.

Подход к исследованию учебных планов, основанный на данных

Ю. Насу, ORCID: 0009-0005-3852-0022 <*yunas2002@iCloud.com*>
М.С. Дробинин, ORCID: 0009-0002-0704-3532 <*msdrobinin@yandex.ru*>
М.С. Ефанов, ORCID: 0009-0003-5824-3259 <*msefanovv@gmail.com*>
В.В. Ланин, ORCID: 0000-0002-0650-2314 <*vlanin@hse.ru*>

Национальный исследовательский университет «Высшая школа экономики»,
Россия, 614070, г. Пермь, ул. Студенческая, д. 38.

Аннотация. Выбор образовательной программы — важный момент в жизни молодых людей. В условиях дефицита времени после экзаменов абитуриенты обычно не успевают проанализировать возможные образовательные маршруты. Кроме того, это требует тщательного изучения учебных планов. Данное исследование посвящено этой проблеме и предлагает алгоритм анализа учебных планов на основе обработки естественного языка названий курсов или компетенций, перечисленных в учебных планах. Описана архитектура интеллектуальной программной системы. Используемый метод протестирован на учебных планах, взятых с университетских сайтов. Для хранения содержания учебных планов было разработано хранилище данных. На данный момент тема исследования плохо изучена. Существующие статьи либо описывают ранние стадии разработки, либо скудны на детали реализации. Они кратко рассмотрены в данной статье.

Ключевые слова: компетенции, анализ учебных планов, NLP, хранилища данных.

Для цитирования: Насу Ю., Дробинин М.С., Ефанов М.С., Ланин В.В. Подход к исследованию учебных планов, основанный на данных. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 83–90 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–7.

1. Introduction

Adolescence is a pivotal point in life. It is during this period that people shape their talents [1]. Therefore, after finishing secondary school, young people need a pathway to follow their calling. For the majority of teenage Russians, a typical way to do so is to obtain a university education.

Due to time pressure and stress, school-leavers can make an incorrect choice regarding an educational program due to a lack of information. The situation is common—it is estimated that nearly half of school-leavers are said to experience difficulties in choosing an educational program [2].

As a consequence, students are often dissatisfied with higher education [3]. This could be avoided if university entrants were more familiar with undergraduate programs, especially their curricula.

Data-driven decision making has already proven its usefulness in the business world, but is not widespread in the education sector. An intellectual system designed to decompose and visualize different parts of learning plans could alleviate the situation and bring about a change.

The curricula of HSE University and ITMO University were scrapped for the research. To store the retrieved data, a data warehouse was implemented. After that, the algorithm for the detection of similar courses was invented and tested.

2. Problem Statement

The issue addressed in this paper is the lack of digestible information about university programs. Curricula are not easily digestible for those unfamiliar with academic management because they contain a large amount of heterogeneous information. The main hypothesis is that the situation might be soothed by a business intelligence software system devised to provide consultancy to young people choosing an academic program.

The objective of the study is the construction of a research prototype of an intelligent system for inquiry into university curricula analysis. To achieve it Natural Language Processing and Data Warehousing methods are to be implemented. Course names and competence definitions listed in learning plans might be used to find similar courses of different programs.

The functional purpose of the system is to allow the user to compare curricula of educational programs by searching for the most similar courses, highlighting the competencies developed during the training, calculating and comparing the number of classroom hours in different programs.

3. Related Works

Currently, Russia applies a competency-based approach to higher education [4]. Modern Russian educational standards do not contain lists of compulsory courses (except for liberal arts and physical education). This gives universities relative freedom in designing curricula, but imposes the responsibility for proper planning of students' learning activities. The ability to compare educational programs can be a useful tool in the implementation of this concept.

In previous research a combination of Bag of Words and Support Vector Machine was proposed for legal document classification [5]. This approach was subsequently criticized for its insufficient accuracy.

The current task has similarities with the previous one, both focusing on Natural Language Processing, however the main distinction lies in the training data. Legal document classification was based on a predefined dataset whereas the analysis of curricula requires data scraping. It is also a similarity detection task rather than a classification task.

To improve the quality of text processing vector embeddings [6] could be used. They represent the parts of speech in the numerical form, preserving the semantics for effective text comparison and summarization. Among the existing models which could be used to produce the embeddings are Word2Vec [6] and BERT [7], both based on neural networks. The key difference is that the former generates context-independent vectors, whereas the latter produces contextualized embeddings [8].

The Graph Theory methods were implemented to evaluate learning plans [9-11]. The model proposed consists of nodes (courses) and edges that measure the distance between vertices. To calculate the distance, the following formula (1) is provided:

$$w(v,u) = \frac{(\sum cred)}{2N} \times \big(cred(v) + cred(u)\big) \times |comp(v) \cap comp(u)|, \quad (1)$$

where: *w* is the *distance* between courses *v* and *u*,
$\sum cred$ is the overall *sum of credits* in learning plan,
N is the *number of courses* in curriculum,
cred(x) is the *number of credits* for course x,
{comp(x)} is the *set of competences* for course x.

The quality of the curriculum is then evaluated using graph density and modularity. It is assumed that the optimal model should have moderate modularity and high density.

However, the proposed distance measurement is not a proper algebraic metric since *w (x, x)* is not equal to zero and the triangle inequality is violated. Even though the zero distance is interpreted as the absence of link between courses the motivation behind the formula is ambiguous.

One of the references of the previously explored articles is the study [12], which proposes a method for the formation of an individual educational trajectory using a dynamic equation model. The model is applicable to a student who has already chosen a program.

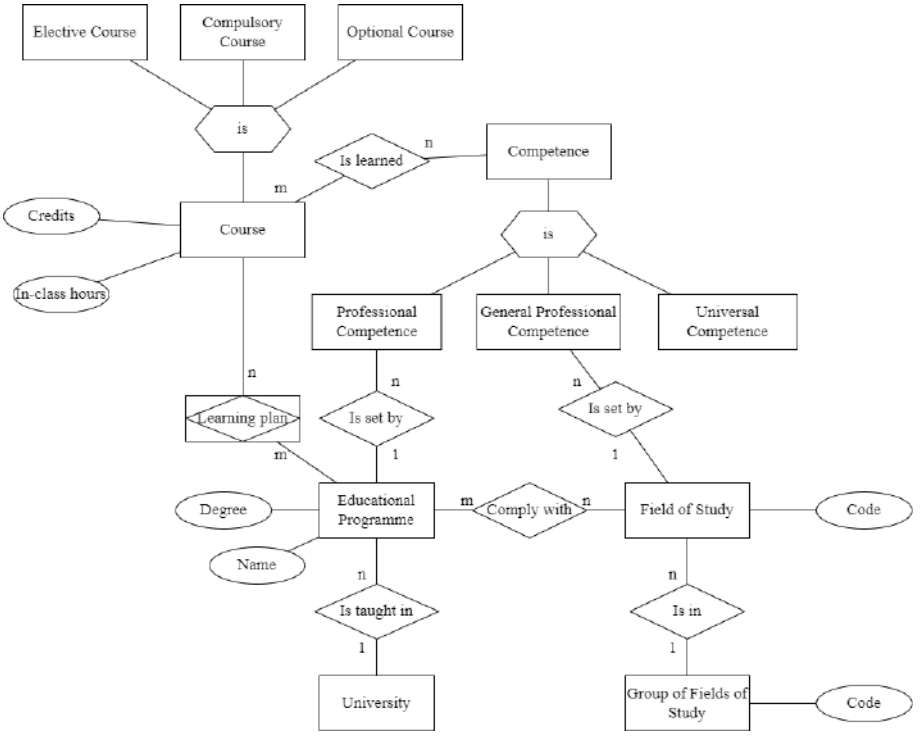


Fig. 1. ER-Diagram of a Learning Plan

Therefore, it is not appropriate for the research objectives. Furthermore, the paper does not provide test cases which verify the model's performance.

4. Curricula Structure and Data Warehouse Modelling

In Russia, Higher Education is regulated by the respective Federal Law [13] and Education Standards [14]. These documents were analyzed to create the Entity-Relationship model (Fig. 1). Then it was transformed to a Data Warehouse Star-Schema (Fig. 2).

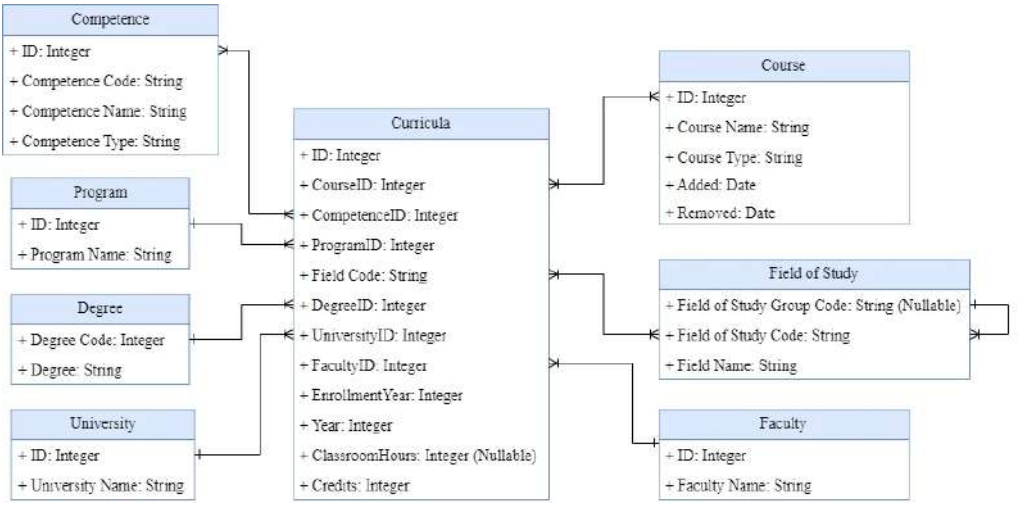


Fig. 2. Data Warehouse schema

Table 1. Modules and functionality

Module	Functionality
Webpage	User Interaction
API Gateway	Load distribution
Data Access Layer	Request handling and access to learning plans data
Data Warehouse	Storage for learning plans data
ETL Module	Data scraping, processing, transformation to SQL and loading
Machine Learning Model	Advanced analytics

Since the graphical interface is requires for user's convenience, a Single Page Web Application is implemented. In order to access data from the warehouse REST API [15] was developed and deployed in Docker Container [16].

Furthermore Nginx [17] proxy server is used for load distribution since the scalability is required due to the seasonality of university enrollment. ETL module is devised for data scraping. Due to different formats of curricula (pictures, tables, pdf-files) the process is not unified yet.

5. Course Similarity Measurement

In order to find similar courses, the following algorithms might be used:

1. Choose two educational programs and obtain their learning plans.

2. For each course in the learning plans assign the competences it improves, find their description (the code will not suffice) in the educational standard.
3. Using pretrained model, vectorize each competence description.
4. For each course compute the mean vector using formula.
5. Normalize the mean vector to unit length.
6. Find nearest neighbors using L2-norm or cosine similarity.

The course names themselves could be used instead of or combined with competence descriptions for steps 2-4 to find the similar courses.

For the experiment the SBERT model [18] was used. As an example, the Software Engineering and the Business Informatics programs taught at the Perm Campus of HSE University were selected. Their curricula were scraped using Selenium [19] and used for the test (Table 2).

The cosine measurements computed for course names and the sets of competences assigned to them are uncorrelated (Pearson and Spearman correlation coefficients equal to -0.11 and -0.13, respectively).

Table 2. Most similar courses

Software Engineering Course	**Business Informatics course**	**Cosine Similarity (names)**	**Cosine Similarity (comp.)**
Top 5 similar by competences			
Group Dynamics	Strategic Management	0.08	0.69
Group Dynamics	Theory and History of Management	0.14	0.69
Group Dynamics	Decision-making	0.04	0.69
Software Design	IT-business infrastructure	0.40	0.65
Databases	Accounting	0.10	0.65
Top 5 similar by names			
Web Programming	Web Programming	1.00	0.44
Safe Living Basics	Safe Living Basics	1.00	0.35
Discrete Mathematics	Discrete Mathematics	1.00	0.10
Research Seminar	Research Seminar	1.00	0.46
Programming	Programming	1.00	0.11

To evaluate the models, expert assessments are used. For that 150 random samples are chosen and ranked manually. Competence-based measurements correlate weakly with expert assessments (Pearson coefficient 0.12), while Name-based measurements show a moderate correlation (Pearson coefficient 0.65).

It is evident that comparison of courses by name gives an imprecise match due to lack of detail, as it favors the general courses (such as History, Law and Economics) or the course with the same name. The name may change over the course of time, but the model would mark them as two different courses. In similar way, it could be taught under various names (e.g., on several educational programs). Though it might be used to provide the overall rough estimate of resemblance.

For this the ratio between the quantity of similar courses (we consider the course similar if the cosine similarity between respective embeddings is higher than 0.65) and the total number of courses for two programs could be computed.

It should be noted that the competence-based approach is inadequate if the data contains inconsistently assigned competences. For example, the above-mentioned Business Informatics' Linear Algebra course is said to develop "positioning products in the global marketplace" (which certainly does not reflect the essence of Linear Algebra).

6. Conclusion

The scientific novelty of the study lies in the use of information technology to analyze curricula. It should be stressed that the project is an investigation into an area of study which is still under-researched. Preliminary research shows that the average number of lessons decreases with the year of study.

Natural language processing methods have been used to compare courses using their names and competences assigned to them.

There were 200 learning plans scraped in purpose of the research. It is planned to scale the project using data from other universities, such as Moscow State University. In future work the other data (such as descriptions of courses and historical data) and methods (such as expert systems) could be used to identify similar courses.

Future work should focus on developing an intuitive user interface based for exploratory search [20]. A guide summarizing the domestic education system and its regulation could be created to improve the user experience.

References

[1]. Kırdök O., Harman E. High School Students' Career Decision-making Difficulties According to Locus of Control. Universal Journal of Educational Research, vol 6, No. 2, pp. 242-248, 2018. doi:10.13189/ujer.2018.060205.

[2]. Kılıç S., Günal Y. University Students' Career Decision Regret: A Mixed-Method Research. IJERE, 2023, vol. 8, No. 3, pp. 521-531. doi:10.24331/ijere.1257601.

[3]. Prakhov I., Rozhkova K., Travkin P. Osnovnye strategii vybora vuza i bar'ery, ogranichivayushchie dostup k vysshemu obrazovaniyu [Main strategies for choosing a higher education institution and barriers limiting access to higher education]. HSE University, Moscow, Rep. 17. Available at: https://www.hse.ru/data/2022/01/25/ 1755651210/ib_17_2021.pdf, accessed 01 Apr. 2024 (In Russian).

[4]. Kelchevskaya N., Shirinkina E. Integraciya obrazovatel'nyh i professional'nyh standartov v usloviyah reformirovaniya: problemy i puti [Integration of educational and professional standards under conditions of reform: problems and ways of solution]. Universitetskoe upravlenie: praktika i analiz [University Governance: Practice and Analysis], issue 1, 2018. pp. 16-25. DOI 10.15826 (In Russian).

[5]. Nasu Iu., Lanin V. Development of Legal Document Classification System Based on Support Vector Machine. Trudy ISP RAN/Proc. ISP RAS, vol. 35, issue 2, 2023. pp. 49-56. DOI: 10.15514/ISPRAS2023-35(2)-4.

[6]. Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector Space. 2013. Available at: arXiv:1301.3781, accessed 01 Apr. 2024.

[7]. Devlin J.; Chang M.-W., Lee K., Toutanova K. BERT: Pretraining of Deep Bidirectional Transformers for Language Understanding. 2018. Available at: https://doi.org/ 10.48550/arXiv.1810.04805, accessed 01 Apr. 2024.

[8]. Shen Y., Liu J. Comparison of Text Sentiment Analysis based on Bert and Word2vec. In Conf. IEEE 3rd ICFTIC, 2021, pp. 144-147. Available at: https://ieeexplore. ieee.org/document/9647258, accessed 01 Apr. 2024.
[9]. Zykova T., Kytmanov A., Khalturin E. Ob analize uchebnogo plana podgotovki bakalavrov v oblasti informacionnyh tekhnologij [On the analysis of the curriculum for training bachelors in information technology]. In Conf. New Educational Strategies in the Modern Information Space, St. Petersburg, 2022, pp. 338- 343. Available at: https://www.elibrary.ru/download/ elibrary_49318813_30209023.pdf, accessed 01 Apr. 2024 (In Russian).
[10]. Zykova T., Kytmanov A., Khalturin E. Analiz formirovaniya kompetencij uchebnogo plana obrazovatel'noj programmy [Analysis of the formation of competences of the educational program curriculum]. In Conf. Transformation of Mechanics and Mathematics and IT-Education in Terms of Digitalization, Minsk, 2023, vol. 1, pp. 176-181. Available at: https://www.elibrary.ru/ download/elibrary_54184628_52556588.pdf, accessed 01 Apr. 2024 (In Russian).
[11]. Zykova T., Kytmanov A., Khalturin E., Noskov M. O sravnenii grafovyh modelej uchebnyh planov podgotovki inzhenerov v oblasti informacionnyh tekhnologij [On the Comparison of Graph Models of Curricula for Training Engineers in the Field of Information Technology]. In Conf. Informatization of Education and E-learning Methods: Digital Technologies in Education, Krasnoyarsk, 2023, pp. 1105-1109. Available at: https://www.elibrary.ru/ download/elibrary_54778536_72713754.pdf, accessed 01 Apr. 2024. (In Russian).
[12]. Mitsel A., Cherniaeva N. Methods for control over learning individual trajectory. IOP Conf. Ser.: Mater. Sci. Eng., 2015, vol. 91, № 012069. doi:10.1088/1757- 899X/91/1/012069.
[13]. Federal Assembly of Russia. (2012). No. 273-FZ, Federal Act on Education. Available at: https://www.consultant.ru/ document/cons_doc_LAW_140174/, accessed 01 Apr. 2024. (In Russian).
[14]. FGOS. Federal Education Standards (Russia). Available at: https://fgos.ru/, accessed 01 Apr. 2024. (In Russian).
[15]. Fielding R. Architectural Styles and the Design of Network-based Software Architectures, Ph.D. dissertation. University of California, Irvine. 2000. Available at: https://ics.uci.edu/~fielding/pubs/dissertation/top.htm, accessed 02 Apr. 2024.
[16]. Docker (2023). Docker: Accelerated Container Application Development. Available at: https://www.docker.com/, accessed 01 Feb. 2024.
[17]. Nginx (2023). Nginx. Available at: https://www. nginx.com/, accessed 01 Feb. 2024.
[18]. SBERT. (2023). SentenceTransformers Documentation. Available at: https://www.sbert.net/, accessed 01 Apr. 2024.
[19]. Selenium. (2023). Selenium Webdriver. Available at: https://www.selenium.dev/documentation/webdriver/, accessed 01 Apr. 2024.
[20]. Ruotsalo T., Peltonen J., Eugster M., Głowacka D., Floréen P., Myllymäki P., Jacucci G., Kaski S. Interactive Intent Modeling for Exploratory Search. ACM Transaction on Information Systems, vol. 36, issue 4, article 44, 2018. 46 pages. DOI:10.1145/3231593.

Информация об авторах / Information about authors

Юри НАСУ – выпускник бакалавриата Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь), образовательная программа «Программная инженерия». Сфера научных интересов: прикладная и вычислительная линейная алгебра, анализ текстовых данных, системный анализ.

Iuri NASU – alumnus of National Research University – Higher School of Economics (HSE University, Perm), educational program “Software Engineering”. Research interests: applied and computational linear algebra, natural language processing, systems analysis.

Михаил Сергеевич ДРОБИНИН – выпускник бакалавриата Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь), образовательная программа «Программная инженерия». Сфера научных интересов: проектирование архитектуры программных систем, объектно-ориентированное программирование.

Mikhail Sergeevich DROBININ – alumnus of National Research University – Higher School of Economics (HSE University, Perm), educational program “Software Engineering”. Research interests: software design and architecture, object-oriented-programming.

Марк Станиславович ЕФАНОВ – выпускник бакалавриата Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь), образовательная программа «Программная инженерия». Сфера научных интересов: business intelligence, системный анализ.

Mark Stanislavovich EFANOV – alumnus of National Research University – Higher School of Economics (HSE University, Perm), educational program “Software Engineering”. Research interests: business intelligence, systems analysis.

Вячеслав Владимирович ЛАНИН – старший преподаватель кафедры информационных технологий в бизнесе Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь). Сфера научных интересов: языки моделирования, предметно-ориентированное моделирование, языковые инструментарии, CASE-средства, системы имитационного моделирования.

Viacheslav Vladimirovich LANIN – Senior Lecturer of the Department of Information Technologies in Business of the National Research University – Higher School of Economics (HSE University, Perm Branch). Research interests: modeling languages, domain specific modeling, language toolkits, CASE tools, simulation systems.

DOI: 10.15514/ISPRAS-2024-36(2)-8

Ontology-based Neurointerface IoT Integration Approach

I.A. Labutin, ORCID: 0000-0001-6858-1479 <*linuxrf@gmail.com*>
S.I. Chuprina, ORCID: 0000-0002-2103-3771 <*chuprinas@inbox.ru*>
Perm State University,
15 Bukireva Str., Perm, 614068, Russia.

Abstract. Recently, there is a surge of interest in employing neurocomputer interfaces for a control contours implementation, especially for different infrastructures of Internet of Things. However, due to a low-level nature of such devices and related software tools, neurointerface integration with a large variety of IoT devices is quite a tedious task, and the one that requires a lot of knowledge in the neuroscience and signal processing to boot. In the paper, we propose an ontology-driven solution for facing the upcoming challenges of unified integration of brain-computer interfaces into IoT ecosystems. We demonstrate an adaptable mechanism for integrating brain-computer interfaces into the Internet of Things infrastructure by introducing an intermediate layer – a smart mediator that will be responsible for communication between the environment and the neurointerface. The mediator's software is generated automatically, and this process is driven by a managing ontology. The proposed formal model and the system's implementation are described. The approach we have developed enables researchers and engineers without strong background in brain–computer interface to automate the integration neurointerfaces with different infrastructures of Internet of Things.

Keywords: Internet of Things; brain–computer interface; ontology engineering; ontology-driven solution; smart mediator.

For citation: Labutin I.A., Chuprina S.I. Ontology-based neurointerface IoT integration approach. Trudy ISP RAN/Proc. ISP RAS, vol. 36, issue 2, 2024. pp. 91-108. DOI: 10.15514/ISPRAS-2024-36(2)-8

Acknowledgements. The authors express their deep gratitude to the Educational and Scientific Laboratory of Sociocognitive and Computer Linguistics, Faculty of Philology, Perm State University, for the provided equipment and support for the research.

Онтологический подход к интеграции нейроинтерфейсов в инфраструктуру интернета вещей

И.А. Лабутин, ORCID: 0000-0001-6858-1479 <*linuxrf@gmail.com*>
С.И. Чуприна, ORCID: 0000-0002-2103-3771 <*chuprinas@inbox.ru*>
Пермский государственный национальный исследовательский университет,
Россия, 614068, г. Пермь, ул. Букирева, д. 15.

Аннотация. В последнее время наблюдается всплеск интереса к использованию нейрокомпьютерных интерфейсов для реализации контуров управления, особенно для различных устройств в инфраструктуре интернета вещей. Однако из-за низкоуровневой природы таких устройств и соответствующих программных средств интеграция нейроинтерфейсов со множеством разнообразных IoT-устройств является довольно трудоемкой задачей, требующей определенных профессиональных

знаний в области нейронауки и обработки сигналов. В данной работе мы предлагаем онтологически управляемое решение для реализации инструментальных средств унифицированной интеграции интерфейсов мозг-компьютер в экосистему интернета вещей. Мы демонстрируем как достигается адаптация к особенностям процесса конкретной интеграции за счет введения в систему промежуточного уровня – интеллектуального посредника, который отвечает за взаимодействие между окружающей средой и нейроинтерфейсом. Программное обеспечение посредника генерируется автоматически, и этот процесс управляется онтологией. Описываются предлагаемая формальная модель и реализация системы. Разработанный нами онтологически управляемый высокоуровневый подход позволяет исследователям и инженерам, не имеющим большого опыта работы с интерфейсом мозг-компьютер, автоматизировать интеграцию нейроинтерфейсов с различной инфраструктурой интернета вещей.

Ключевые слова: интернет вещей; интерфейс мозг–компьютер; онтологический инжиниринг; онтологически управляемое решение; интеллектуальный посредник.

Для цитирования: Лабутин И.А., Чуприна С.И. Онтологический подход к интеграции нейроинтерфейсов в инфраструктуру интернета вещей. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 91–108 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–8.

Благодарности. Авторы выражают глубокую признательность учебно-научной лаборатории социокогнитивной и компьютерной лингвистики филологического факультета Пермского государственного национального исследовательского университета за предоставленное оборудование и поддержку.

1. Introduction

Due to the widespread digitization and an active expansion of the application areas of Internet of Things (IoT), virtual reality and augmented reality, methods and tools for managing software systems based on neurointerfaces are developing rapidly. Of crucial concern here is that there is no universal standard for the integration of different IoT infrastructures nowadays. The diversity of existing protocols and standards for device interaction in the IoT, as well as their lack of compatibility, leads to communication problems between devices within a single infrastructure. Therefore, the call for development and implementation of a new unified concepts and refinement of existing ones to increase the level of interoperability of devices seems to be quite an essential one, especially when it comes to the task of embedding neurointerfaces into already existing software systems.

In the current state of the art, issues related to creating unified methods and tools to automate the process of integration of neurointerfaces into the IoT ecosystem with a goal of controlling its components (target systems and subsystems) are still insufficiently studied, although the literature recognizes the need to develop such methods and approaches [1-4].

In most cases, neurointerface equipment is able to function properly only with a narrow range of proprietary software provided by the manufacturer. Therefore, employment of neurointerfaces in the scenarios and pipelines not accounted by a manufacturer poses quite a challenge, if ever possible, and requires a deep knowledge in fields of both neuroscience and computer science.

In this paper we present the concept of ontology-driven system for integration of neurointerfaces into IoT ecosystems and the approach to its implementation. The general idea – automated generation of a firmware for a smart mediator connecting together an IoT infrastructure and a neurointerface – was introduced in our previous work [5]; in this paper, we focus on proposing a formal model and describe its implementation.

We shall note here that a general overview of neurocomputer interfaces is out of scope of this work; interested readers can be advised to familiarize themselves with an excellent summary paper [6].

2. Use case scenarios

Before building the system model in question, it is necessary to understand the user's general portrait and possible scenarios of his interaction with the system.

It shall be noted that there are two categories of potential users of our system in relation to the fields of neurointerfaces, IoT and ontological engineering: non-specialists and specialists.

Non-specialists will use the system in the form provided to them by specialists; non-specialists don't have the expertise to (and, ideally, should not) configure and extend the system as they may have no relevant competencies. In essence, non-specialists will employ the system "as is" as a set of available tools for solving their own personal problems related to integrating neurointerfaces into specific IoT infrastructure.

Specialists, on the other hand, in addition to the aforementioned employment of the system, can also extend and reconfigure it by adding new modules and their ontological descriptions.

It is important to distinguish these two groups of users as the former ones may need a complete and convenient high-level interface; while the latter can use a lower-level means of interaction with the system to increase flexibility and efficiency of task solving.

The main non-specialists' scenario for the platform's employment will be using a high-level visual user interface to create a formalized description of a "smart assistant's" internal composition and employing some special means to automatically generate its source code based on this formal description. The generated code then becomes a part of the software of such assistant and unifies the integration of user selected modules. Based on our previous work [7] we propose to use the tools of SciVi platform to tackle this problem.

SciVi is Russian scientific visualization and analytics platform enabled describing the pipeline of analytical data processing by means of data flow diagrams and generate the source code in related programming language. This was made possible due to SciVi being an ontology-driven solution.

This scenario is presented in Fig. 1 as a use-case diagram, where two usage scenarios related to employing the platform by non-specialists are illustrated. First scenario – composing DFD diagram in SciVi environment. Second scenario – generating code for the smart assistant based on previously created DFD diagram.

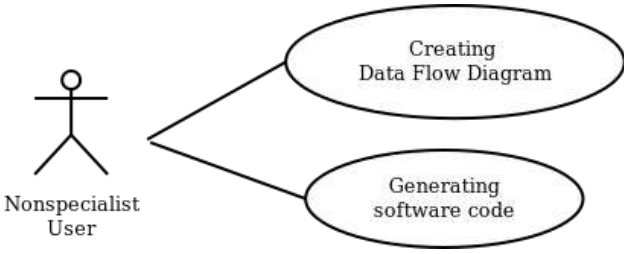


Fig. 1. Use cases for non-specialists

For specialists, an additional scenario is added that involves extending the system with new components. Aside from components of interest they also need to add their ontological descriptions (manually crafted or generated by some tooling) into the system's repository. This is illustrated on Fig. 2 with three use cases related to employing the platform by specialists. Compared with Fig. 1, there is an additional third scenario – filling the system repository with ontological descriptions of new modules available for potential inclusion in code for the smart mediator.

Note here that this is by no means is not an exhaustive list of potential ways to interact with the system; on practice, other interaction scenarios are possible, which are not covered in the paper. Their discovery and analysis are the topic of a further study. In this paper, we focus on providing users who belong to the category of "specialists" with the necessary tools for integrating neurointerfaces into arbitrary IoT infrastructure.

To this end, we propose the ontology-driven solution in which the so-called "smart mediator" plays a key role.

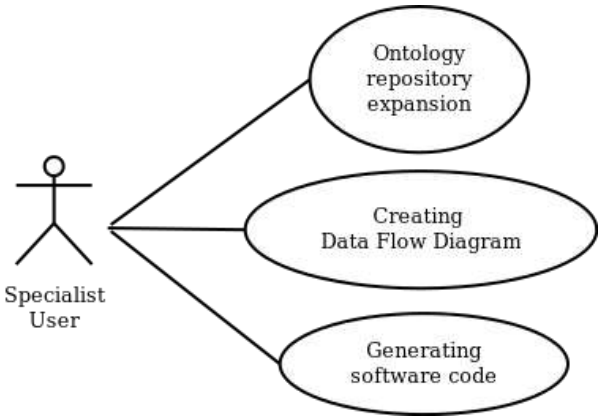


Fig. 2. Use cases for specialists

3. Ontological approach towards integration

In the context of computer science, ontology is seen as a formal representation of knowledge about some domain in a form of sets of concepts and relations between them, as well as associated axiomatic. One of the founders of ontological engineering, Thomas Gruber, defined ontology as "explicit specification of conceptualization" [8].

Following [9] we will define a formal model of ontology O an ordered triplet of the form:

$$O = \langle X, R, \Phi \rangle, \quad (1)$$

where:

- X is a finite non-empty set of concepts (notions, terms) in the domain area represented by ontology O;
- R is a finite set of relationships between concepts;
- Φ is a finite set of interpretation functions defined on the concepts and relations of ontology O (axiomatic).

Furthermore, [9] points out that in an ontology-based system model, there are three ontological components:

1) Metaontology, which contains general concepts and notions independent of the domain area.
2) Domain ontology (one or several), which describes a specific domain area.
3) Task ontology (one or several), which contains types of tasks to be solved and their decomposition into subtasks.

Regarding our goal, we will be mainly interested in domain ontologies. Metaontology for our purpose is not strictly required due to a clearly defined (at least on this stage of research) domain area. For similar reasons, we do not see a need for task ontologies: as it was noted above, the system shall support only a narrow range of use cases, therefore instead of an ontological description of tasks, knowledge about them and ways of interpreting supported types of relationships should be built into the inference engine.

This seems to be the most effective solution in the context of current research, because in such a case it is possible to use so-called "lightweight" ontologies [10].

Here and further, if nothing else is specified explicitly, under the term "ontology" will be understood only "lightweight" ontologies, which do not contain axiomatic (that is, in 1 $\Phi = \emptyset$). Their interpretation falls entirely on the implementation of inference engine. We suggest to create ontologies using environment of our visual ontology editor named ONTOLIS [11-13]. ONTOLIS is

aimed at ordinary users and at the same time allows developers uniformly storing a wide range of attributes in the nodes and arcs of the ontology graph in a unified form which is necessary to take into account the specifics of solving specific tasks during inference [13]. This allows the user to expand the possibilities of interpreting ontologies to some extent, providing an inference engine with additional information in simple machine-readable format, which gives the opportunity to modify its behavior (including referencing external tools).

4. Requirements for the system and the "smart mediator"

Before describing the formal model of the system, it is necessary to define a set of requirements that both the system itself and its results should meet.

Functional requirements for the "smart mediator" logically follow from the purpose of his existence.

1) The smart mediator must receive a signal from the neurocomputer interface. Consequently, the smart mediator should include a module responsible for communication with the neurointerface.
2) The smart mediator must process the received signal. Processing signals is a nontrivial issue per se and is not the subject of this work. It is assumed that the mediator should include several modules responsible for this task. The question of suitability of these modules lies on the shoulders of the user of the system (as was said above, we are talking about a competent user). Within the framework of this work, a set of modules is implemented, sufficient for demonstrating the proof-of-concept of suggested approach to unified integration of the smart mediator into the IoT infrastructure.
3) The smart mediator must provide access to the result of signal processing to other devices in the IoT infrastructure. This can be done in at least two ways (see section V); the choice of a specific mechanism also depends on requirements imposed by the user of the system.

The main non-functional requirements for a smart mediator – flexibility, reliability, predictability – are described below.

1) Flexibility means that the algorithm of the "smart mediator" is not cut in stone once and for all: it can adapt according with changing user needs or surrounding infrastructure. Within the proposed solution this means the need for regenerating the code of the mediator based on a formalized description of the changed infrastructure.
2) Reliability implies that the mediator should have a certain degree of robustness to errors that inevitably occur during interaction between software systems. A detailed error typology is beyond the scope of this work, so from here we assume that the requirement for reliability is inherently implemented at the design stage of modules from which the "mediator" is composed of[1].
3) Predictability guarantees that the mediator will perform only the task prescribed to it by the formal model. This requirement can be received "for free" to some degree if we strictly stick to the paradigm of open-source and free software[2].

5. System development concept

Based on our experience in the field of ontology-driven human-machine interfaces [11-14], we propose a mechanism for integrating brain-computer interfaces (BCI) into the Internet of Things

[1] Strictly speaking, it was also worth to take into account problems arising during interaction between modules of the mediator with each other. The problem of organizing interaction between modules of the mediator among themselves is not a key issue in this paper and is left for the specialist who composes formal representation for further generation of code of the mediator.

[2] Of course, openness and freedom per se do not guarantee full absence of random or deliberate defects leading to unexpected behaviour, but they at least minimize risks associated with them through the "thousand eyes effect" that declares a positive correlation between the quality of the software and the number of people who have access to its code.

infrastructure by introducing an intermediate layer – a smart mediator that will be responsible for communication between the environment (bci:Context in the terminology of [15]) and neurointerface (bci:Device in [15]). The mediator's software is generated automatically, and this process is driven by a managing ontology. In addition, to perform such a generation some extra resources will be required:

- Ontology of semantic filters [16] that describes the necessary transformations of data received from the neurointerface. Parts of this ontology were borrowed from SciVi and used in this work. In addition, it was extended with additional modules specific to neuroscience (for example, with a description of a module for inverse Fourier transform).
- Ontology of target platform that reflects the characteristics of the environment or device that will support the execution of the mediator. This ontology was created from the scratch but taking into account the quirks of SciVi platform.
- Component ontology [17] that provides a complete description of the characteristics of the neurointerface sufficient for establishing communication with it and exchanging data. This ontology was also created from the scratch.

It is important to emphasize in the proposed approach the key role of the mediator, which is considered and implemented as a software module running on one of the devices in the infrastructure of the Internet of Things. In essence, it represents a microservice that receives raw data of a biological nature from a neurointerface and provides access to the results of processing that data to other devices in the IoT infrastructure in one of two possible ways:

1) The smart mediator publishes results of data processing by some protocol, and other network nodes independently and periodically call in to receive information about some aspects of cognitive state of a human. For example, a "smart" lighting bulb can automatically obtain fatigue level data and regulate the light intensity taking into account this information.
2) The smart mediator generates a direct control signal affecting one of IoT infrastructure devices. For example, mediator may define the state of concentration of a person and send it via MPRIS protocol[3] to media device playing multimedia (video / audio) command to pause playback.

The mediator itself may execute either on a specialized device (such as development board for ESP32 microcontroller), fully utilizing all its resources, or on a preexisting infrastructure node (for example, a router or even a personal computer). In the latter case such a mediator can be considered "virtual". The only significant requirement is a physical ability to connect the neurointerface to an equipment on which the mediator works.

As noted above, software code of mediator is being generated following the managing ontology, contents of which is vital for correctness of the mediator functioning. A very important features of ontologies are their transparency, documentability and readability both for human and program agent. This is a significant trait that allows reusing external ontological resources.

Despite of existence of automated tools for creation and validation of correctness of ontologies [18], [19], in reality most practical approaches to choose and quality assessment of ontologies still rely on experts and knowledge engineers (ontology engineers). The other popular approach is to deduce a correctness of built ontology from the result of correctness evaluation of ontologically driven solution.

From our point of view ontological descriptions of various neurointerfaces are not obliged to be integrated in a framework of one domain ontology. It seems viable to develop an interface for so-called smart repository for conveniently search and reuse ontologies of other neurointerfaces or/and

[3] https://specifications.freedesktop.org/mpris-spec/latest/

their parts. Work of such a repository shall be controlled by metaontology (highest-level ontology), i.e., ontology that is storing knowledge about all domain ontologies stored in the repository. Naturally, at same time it is required to ensure coherence and consistency of ontologies [20], [21], but with moderately small sizes of ontologies this condition can be met with a relative ease.

Fig.3 demonstrates a general overview of proposed integration mechanism for neurointerfaces into IoT ecosystem. Here bci:Device is denoting the neurointerface, bci:Context describes environment to be integrated with, BCI-O – main managing ontology, mediator – mediator itself, adapter – a system that implements an integration mechanism integration for a specific neurointerface into an IoT ecosystem.

The ontological device description (bci:Device) is being automatically united with the ontology describing infrastructure (bci:Context) in one common domain ontology, using which a system ("Generator") generates software for smart mediator.

To perform an integration of a neurointerface into existing IoT infrastructure (for example, a "smart home" ecosystem) it's necessary to execute following steps (through suggested algorithm description every mention of creation of ontologies assumes that this can be performed employing ontology learning tools for automated creating of ontologies):

1) Either select an existing ontological neurointerface description from the repository of the SciVi platform [22] or create a new one in the visual ontology editor ONTOLIS saving it to the SciVi repository.
2) Choose a preexisting or create new ontological descriptions of semantic filters for preprocessing input/intermediate/output signals based on the task requirements, similar to how this is implemented in our colleagues' work dedicated to the SciVi system for scientific visualization and visual analysis.
3) Either select an existing ontological description of the specific IoT ecosystem from the repository of the SciVi or create a new one consistent with BCI-O and IoT-O [23].
4) Use services of the proposed system for an automatic construction of adapters for neurodevices and an automatic generation of the smart mediator's firmware from obtained ontologies; this firmware will be responsible for integration of neurointerface in the infrastructure of IoT.

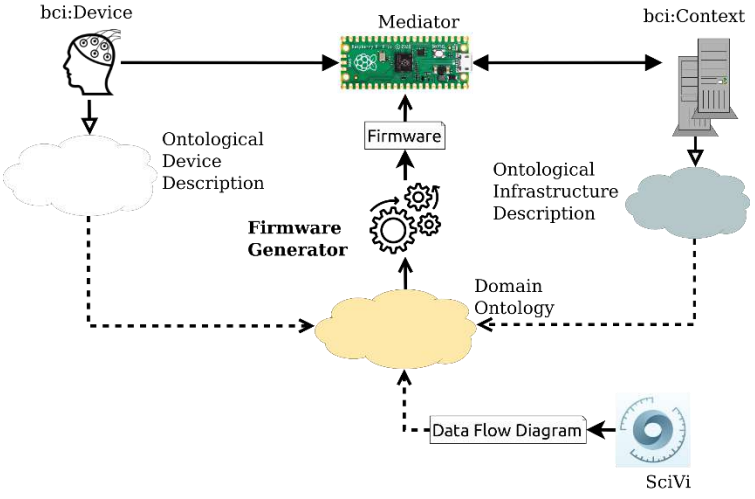


Fig. 3. Proposed integration mechanism

A mechanism for ontology-driven generation of firmware's source code for IoT ecosystem was previously developed by our colleagues as a part of a project aimed at creating and improving the

scientific visualization and visual analysis platform SciVi [22]. We reuse similar concepts when solving tasks related to integrating neurointerfaces into an already existing, or created from scratch ecosystem of IoT.

The proposed solution allows to unify the process of integration of neurointerfaces in a specific IoT ecosystem. Building an ontology describing a particular neurointerface must be performed only once, after which it is added to the corresponding repository of ontologies and can be repeatedly reused for automatic regeneration of mediator's firmware with the goal of integrating it with different environments.

Thus, unification of integration tools is achieved by reusing ontologies and the ability to adapt an ontology-driven system to solve specific problems through replacing ontologies without the need to make changes in previously developed source code for other components of the system [12-14].

6. Formal model of a system

In this paper we propose the following formal model of the system described in set-theoretic notation:

$$\Xi = \langle \Omega, \Delta, Ob, OF, Oi, \Gamma, M, E, \Lambda, OL, S \rangle, \quad (2)$$

where:

- Ω is the operator for generating source code for mediator, Ω: $O_D \rightarrow S$;
- Λ is the operator for building an ontology of module connections, Λ: $O_b \times O_F \times O_i \times D \rightarrow O_L$;
- O_D is the ontological description of the domain area;
- O_b is the ontological description of a specific neurointerface;
- $O_F = \bigcup_{i=1}^{k} O_{fi}$ the set of k ontological descriptions of modules that implement data transformation;
- O_i is an ontological description of the infrastructure, management of which is assumed to be implemented (module for controlling this infrastructure);
- O_L is an ontological description of the links between modules included in the mediator's firmware;
- Δ is the operator for building an ontology from parts, Δ: $O_b \times O_F \times O_i \times O_L \rightarrow O_D$;
- Γ is an interaction operator, Γ: $E \times M \rightarrow D$;
- M is a set of supported control elements;
- E is a set of supported user actions;
- D is a user-created formal description of the interconnection of modules in the form of DFD diagrams in SciVi toolbox;
- S is the source code for the mediator's software.

Let's note that Γ, M and E are out of scope of this paper and are based on reusing results obtained by our colleagues within the framework of the SciVi platform [12-14, 22].

A Δ operator responsible for building an integrated domain ontology is described at Algorithm 1.

7. Basic concepts and relationships of ontological descriptions

In order to maintain a compatibility with the SciVi system (in particular, for reusing the operator Γ, which is responsible for providing users with an easy-to-use interface for building a formal description of module interaction) in this work we largely reused a set of basic concepts and types of relationships that SciVi supports. Such modules and relationship types are highlighted in italics.

A Λ operator that builds an ontological representation of connections between modules is described at Algorithm 2. It reuses a Merge function from Algorithm 1.

Algorithm 1 Δ operator

```
procedure Merge(Om, Oi)
    for all n ∈ Nodes(Oi) do
        if n ∉ Om then
            AddNode(Om, n)
        end if
    end for
    for all r ∈ Relations(Oi) do
        if r ∉ Om then
            AddRelation(Om, r)
        end if
    end for
end procedure
OD ← ∅
Merge(OD, Ob)
for all Ofi ∈ OF do
    Merge(OD, Ofi)
end for
Merge(OD, Oi)
Merge(OD, OL)
```

Algorithm 2 Λ operator

```
Merge(OD, Ob)
for all Ofi ∈ OF do
    Merge(OD, Ofi)
end for
Merge(OD, Oi)
OL ← ∅
root_out ← FindNode(OT, 'Output')
for all block ∈ D do
    name ← Name(block)
    start ← FindNode(OT, name)
    outputs ← GetAdjacentNodes(OT, {start, root_out})
    for all dfd_out ∈ Outputs(block) do
        ont_out ← outputs[Name(dfd_out)]
        if ont_out ∉ L then
            AddNode(L, ont_out)
        end if
        for all dfd_conn ∈ Connections(dfd_out) do
            t_name ← Name(Target(dfd_conn))
            ont_in ← FindNode(OT, t_name)
            if ont in ∉ L then
                AddNode(L, ont_in)
            end if
            r ← Relation(ont_in, ont_out, 'use')
            AddRelation(L,r)
        end for
    end for
end for
```

The basic (root) concepts of the ontological model of the system include:

- *Root* – root node, required by SciVi for the proper operation of Γ operator.

- Module – a concept reflecting a specific software module potentially available for inclusion in the firmware.
- *Type* – data type.
- *Entry* – entry point into the module.
- *Input* – input data of the module.
- *Output* – output data of the module.
- *Array* – subtype of "array" data introduced to simplify the implementation of an inference engine.

Relationship types supported by the system:

- *is_a* – inheritance relation.
- *has* – relation of ownership.
- *use* – relation of employment. In the context of the system, it means "consumes" in the sense "the input parameter A of procedure P1 consumes the output parameter B of procedure P2."

The system allows for the existence of other basic concepts and types of relationships in ontologies, but does not try to interpret them. This architectural solution was also dictated by the desire to maintain compatibility between ontological descriptions with a representation understood by operator Γ. However, some relationship types, which are not strictly necessary, were excluded for maintaining readability of ontologies – for example, the "base type" relation used by operator Γ to display array element type to user during building of formal description D.

Examples of an ontological description of the module and link between modules using aforementioned concepts and relationship types are shown on Fig. 4 and Fig. 5, respectively. The Test1 module illustrated on Fig. 4 has a "test1_process" entry point and three parameters: a1, a2 (input) and a3 (output). Parameters a1 and a3 have an integer type `int`, while parameter a2 – the integer type `short`. The fragment of ontological description of link between module parameters on Fig. 5 reflects the following dependencies: parameter a2 gets its value from a parameter k2, parameter a1 – from k1, and a3 – from q3.

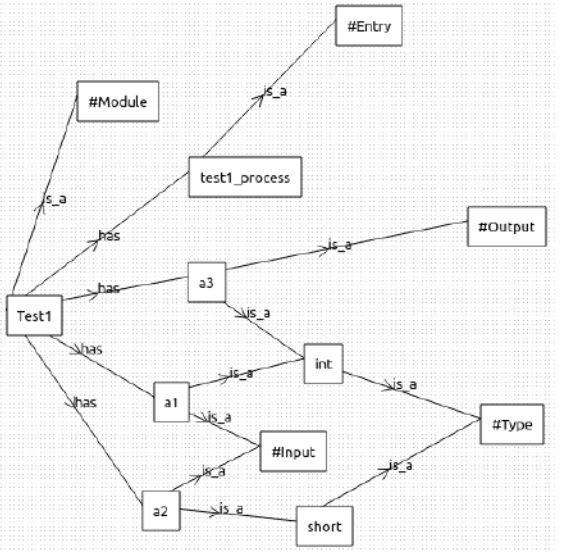


Fig. 4. Example of ontological description of module

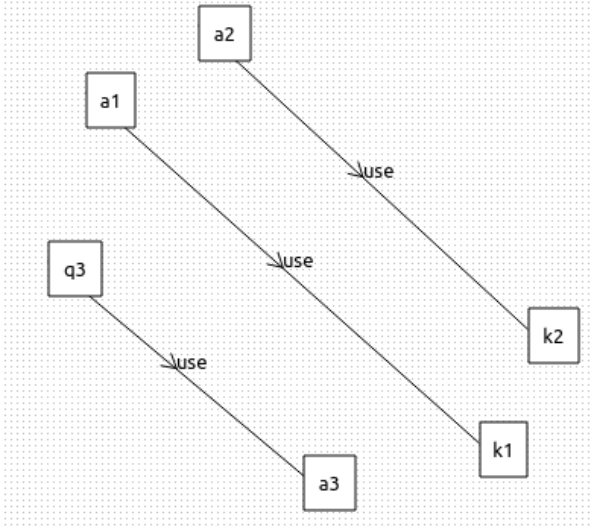


Fig. 5. Fragment of ontology describing link between modules

8. System development and implementation

Within the framework of current research, a system prototype was developed and implemented based on the proposed approach. The system allows its users to:

- create a formal description of the relationship between "smart mediator" modules employing the user-created ontological descriptions of these modules, then
- automatically build an overall domain ontology using merging strategy [12], [13] without performing the "alignment" procedure, describing the structure of a firmware for a smart mediator, and finally
- generate code that connects modules into a whole program component.

On Fig. 6, the architecture of the developed system is presented. Green color is used to highlight components implemented within the scope of this research. SciVi platform here acts as an ontology repository and is used by a user for creating a formal description of connection modules in the form of DFD diagram.

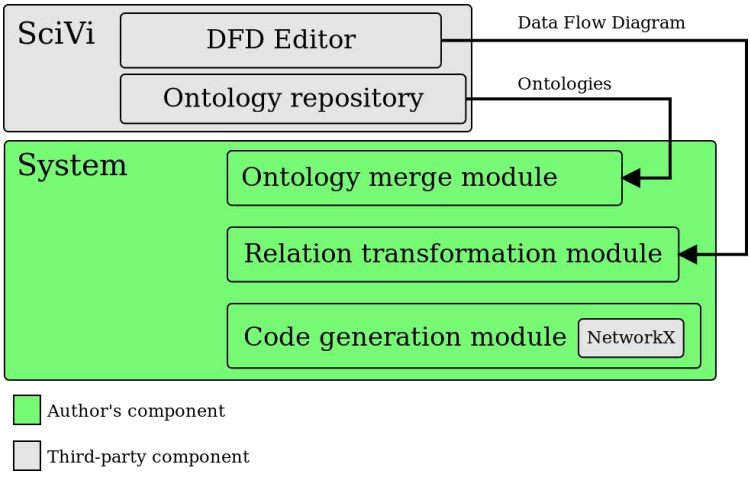


Fig. 6. Architecture of the developed system

General user-specialist workflow with system presented on Fig. 7. First (if necessary), all required program modules are created and their ontological descriptions are built. Then a DFD diagram describing connection these modules is created in SciVi environment. Based on this diagram and ontological descriptions of modules a general domain ontology is created, and after that a smart mediator code generator is executed.

System input data are a set of ontological descriptions of program modules available for inclusion in mediator's firmware and a formal description of connection these modules. As a result, the system generates a source code of a software module in C programming language, which after compilation and linkage with software modules chosen by user become a firmware for a "smart mediator" with goal to solve task of integrating a neurointerface into Internet of Things ecosystem.

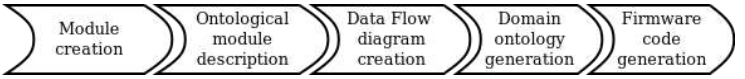


Fig. 7. System usage

Ontological descriptions of program modules are composed from a set of relations and basic concepts described in section VII. These ontological descriptions act as input data for the system, being combined into a whole general domain ontology that becomes the basis for further mediator's firmware code generation.

In addition, these ontological descriptions are used by SciVi platform to provide user with an interface of constructing formal description of program module connection.

Ontological descriptions aligned with program modules code are created (automatically or manually) and placed in SciVi repository from where they will be loaded during system's operation.

The system uses the JSON-based format of ontological descriptions – ONT. It's a proprietary format for representing ontologies supported by ONTOLIS [24], [25] and SciVi platform. Choosing it ensures interoperability between two systems.

Formal description of connection of modules is produced by a user in SciVi platform in form of a data flow diagram. Example of such description is presented on Fig. 8.

Pipeline presented above was designed for the task of brain activity analysis for processing recorded data and used in another work of the authors [7].

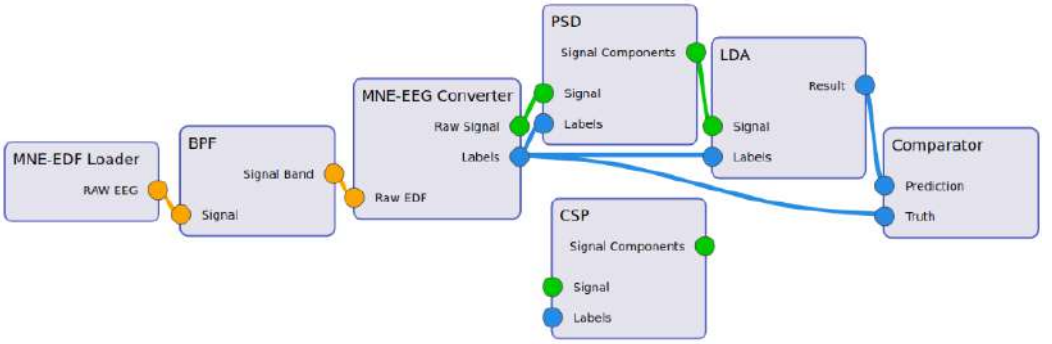


Fig. 8. DFD diagram example

After formal description is built it is exported from SciVi platform for further employment. This file uses the proprietary format based on JSON (JavaScript Object Notation) and should be integrated with ontological descriptions of modules mentioned above. This task is performed by a system's module that executes that transformation by accepting as input aforementioned file of proprietary format together with ontological descriptions of modules. It extracts information about parameters of modules and their connection, after which it creates ontology describing dependencies between data of modules which in turn is the result produced by this module.

The obtained ontology reflecting relations between parameters of modules gets combined with ontologies describing separate modules and general domain ontology is built describing internal structure of a smart mediator. This ontology represents different modules to be included into a firmware as well their data interconnections and in fact represents applied ontology for the task at hand.

On Fig. 9 example fragment of such ontology is presented.

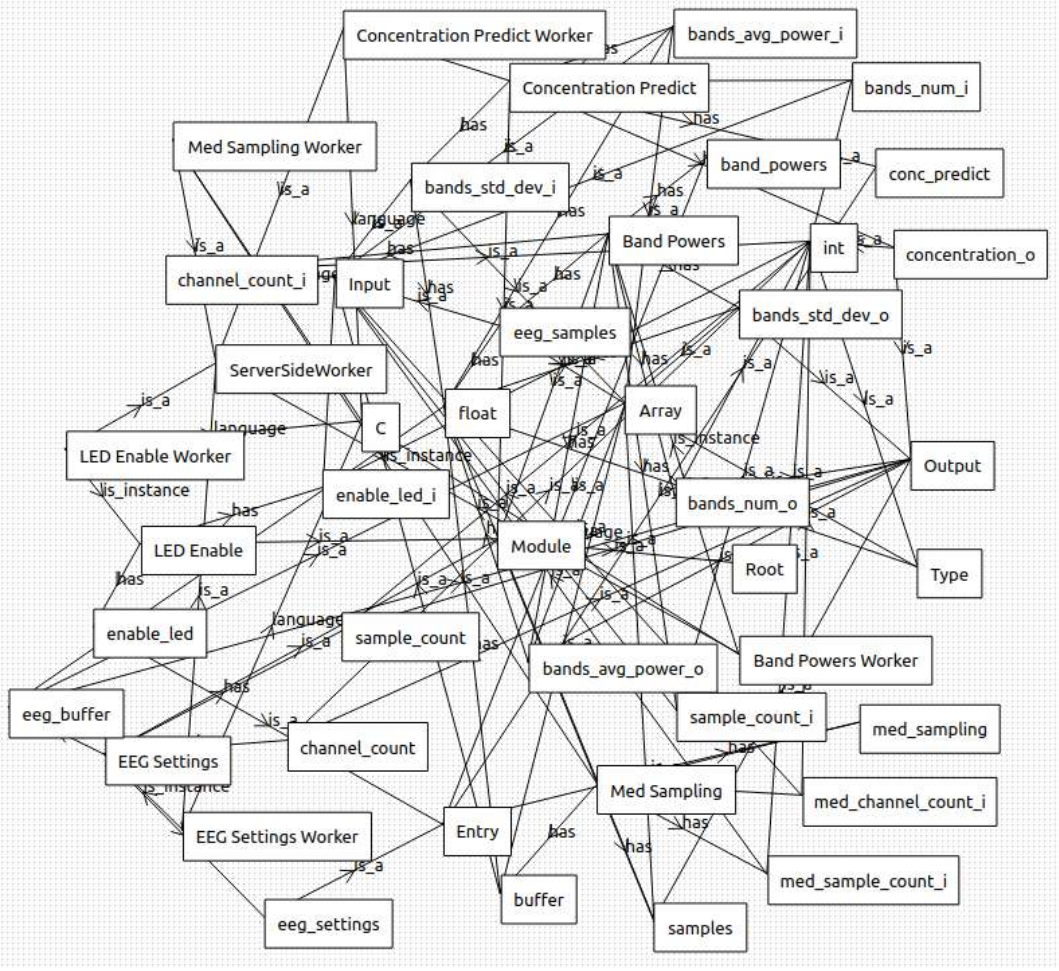


Fig. 9. Example of domain ontology

This ontology gets transferred to the input of the code generator. Generator uses information about modules and their interdependencies extracted from ontology and builds a direct graph describing modules interconnected by data transfers. This graph is processed using topological analysis methods is then transformed such that a following partial order is established: vertex V_i describing module M_i precedes vertex V_j describing module M_j only when there is no path from vertex V_j to vertex V_i. If for some module pair establishing such an order is not possible (it means there is a data cycle dependency between modules) generator stops processing and displays an error message. In case of a successful transformation generator proceeds to generate a source code calling modules' entry points in the order defined by dependencies graph taking into account types of parameters and their direction (input / output).

As a final result the system produces a file with a source code in C language. This file contains instructions which ensure interaction between ontologically described modules according to given ontology. Example of such generated code is provided in Fig. 10. We'd like to point out that while such a code is by no means can be considered complex by a seasoned system programmer, for a user with little to no background in programming writing such a glue can pose a significant challenge. Automated generation allows us to provide the less acquaint users with tools to solve their tasks in a more efficient way.

```
float* eeg_buffer;
int channel_count;
int sample_count;
float* samples;
int bands_num_o;
float* bands_avg_power_o;
float* bands_std_dev_o;
int concentration_o;

void eeg_settings(float** eeg_buffer, int* channel_count, int* sample_count);
void med_sampling(float* buffer, int med_sample_count_i, int med_channel_count_i,
                  float** samples);
void band_powers(float* eeg_samples, int channel_count_i, int sample_count_i,
                 int* bands_num_o, float** bands_avg_power_o, float** bands_std_dev_o);
void conc_predict(int bands_num_i, float* bands_std_dev_i, float* bands_avg_power_i,
                  int* concentration_o);
void enable_led(int enable_led_i);

eeg_settings(&eeg_buffer, &channel_count, &sample_count);
med_sampling(eeg_buffer, sample_count, channel_count, &samples);
band_powers(samples, channel_count, sample_count, &bands_num_o,
            &bands_avg_power_o, &bands_std_dev_o);
conc_predict(bands_num_o, bands_std_dev_o, bands_avg_power_o, &concentration_o);
enable_led(concentration_o);
```

Fig. 10. Example of generated code

Conclusion and future work

The paper is devoted to the urgent problem of automating the process of integrating different types of neurointerfaces into the IoT infrastructure and focuses on the development of methods and tools for unifying the ways of integrating neural interfaces with third-party systems on the principles of adaptability using ontology engineering methods.

The presented concept and formal model of the proposed solution are characterized by novelty, since a new ontology-driven method of firmware generation for the so-called "smart mediator" is proposed, which plays a major role in integrating a specific neural interface into arbitrary IoT infrastructure. The proposed approach makes it possible to reuse third-party ontological resources (BCI ontology) as a basis for building new ontologies of neurointerfaces and expands the existing functionality and ontological resources of the SciVi platform. The proposed approach has been tested on several real-world tasks and has proven its viability, as evidenced by the existing act on the successful implementation of the developed tools in the practical activity of the Educational and Scientific Laboratory of Sociocognitive and Computational Linguistics of PSU. The certificate of ROSPATENT № 2023612016 [26] on the state registration of the developed software has been received.

As a direction of further research, we aim to concentrate on improving usability of implemented tools and expanding their abilities; one such point is introducing a recurring dependency between program modules of the smart mediator.

References

[1]. Huang S., Tognoli E. Brainware: Synergizing software systems and neural inputs. In Companion Proceedings of the 36th International Conference on Software Engineering, ser. ICSE Companion 2014, Hyderabad, India: Association for Computing Machinery, 2014, pp. 444–447, ISBN: 9781450327688. DOI: 10.1145/2591062.2591131.

[2]. McCullagh P., Ware M., McRoberts A., Lightbody G., Mulvenna M., McAllister G., González J. L., Medina V.C. Towards standardized user and application interfaces for the brain computer interface. Universal Access in Human-Computer Interaction. Users Diversity, C. Stephanidis, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 573–582, ISBN: 978-3642-21663-3.

[3]. Huggins J., Guger C., Aarnoutse E., et al. Workshops of the seventh international brain-computer interface meeting: Not getting lost in translation. Brain-Computer Interfaces, pp. 1–31, Dec. 2019. DOI: 10.1080/2326263X.2019.1697163.

[4]. Allison B. The I of BCIs: Next generation interfaces for brain–computer interface systems that adapt to individual users. Human-Computer Interaction. Novel Interaction Methods and Techniques, J. A. Jacko, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 558–568, ISBN: 978-3-642-02577-8.

[5]. Лабутин И.А., Чуприна С.И. Концепция построения онтологически управляемых нейроинтерфейсов. Интеллектуальные системы в науке и технике, 2020, с. 105-111. / Labutin I.A., Chuprina S.I. Concept of ontology-driven neurointerface development, Intellectual systems in science and technology, 2020, pp. 105–111 (in Russian).

[6]. Lunev D., Poletykin S., Kudryavtsev D. Brain-computer interfaces: Technology overview and modern solutions. Modern Innovations, Systems and Technologies, vol. 2, no. 3, pp. 0117–0126, Jul. 2022, ISSN: 2782-2818. DOI: 10.47813/2782-2818-2022-2-3-01170126.

[7]. Ryabinin K., Chuprina S., Labutin I. Ontology-driven toolset for audio-visual stimuli representation in eeg-based bci research. In Proc. of the International Conference on Computer Graphics and Vision "Graphicon", CEUR, vol. 31, Keldysh Institute of Applied Mathematics, 2021, pp. 223–234. DOI: 10.20948/graphicon-2021-3027-223-234.

[8]. Gruber T.R. A translation approach to portable ontology specifications. Knowledge Acquisition, vol. 5, no. 2, pp. 199–220, 1993, ISSN: 1042-8143. DOI: https://doi.org/10.1006/knac.1993.1008.

[9]. Гаврилова Т., Хорошевский В. Базы знаний интеллектуальных систем: Учебник. Питер, 2000, 384 с. ISBN: 9785272000712. / Gavrilova T., Khoroshevskii V. Bazy znanii intellektual'nykh sistem: Uchebnik. Piter, 2000, 384 p. ISBN: 9785272000712 (in Russian).

[10]. Чуприна С.И. Адаптация технологий фабрик данных к разработке систем визуальной аналитики в области цифровой медицины. Труды 33 Международной конференции по компьютерной графике и машинному зрению «Графикон-2023», Институт проблем управления им. В.А. Трапезникова РАН, 2023, с. 405-416. / Chuprina S.I. To Adapt Data Fabric Technology to Visual Analytics Systems Development in the Field of Digital Medicine. In Proc. of 33th International Conference on Computer Graphics and Machine Vision "GrafiCon-2023", V.A. Trapeznikov Institute of Control Sciences of RAS, 2023, pp. 405–416. DOI: 10.20948/graphicon-2023-405-416 (in Russian).

[11]. Chuprina S.I., Ryabinin K.V., Koznov D.V., Matkin K.A. Ontology-driven visual analytics software development, Programming and Computer Software, vol. 48, no. 3, pp. 208–214, Jun. 2022, ISSN: 1608-3261. DOI: 10.1134/S0361768822030033.

[12]. Чуприна С.И., Рябинин К.В., Кознов Д.В., Маткин К.А. Онтологически управляемые средства автоматизации разработки приложений визуальной аналитики, Программирование, т. 3, 2022, с. 70-77. / Chuprina S.I., Ryabinin K.V., Koznov D.V., Matkin K.A. Ontologicheski upravlyaemye sredstva avtomatizatsii razrabotki prilozhenii vizual'noi analitiki, Programmirovanie, no. 3, 2022, pp. 70–77 (in Russian).

[13]. Chuprina S.I. Using data fabric architecture to create personalized visual analytics systems in the field of digital medicine, Scientific Visualization, vol. 15(5), 2023, pp. 50–63. DOI: 10.26583/sv.15.5.05.

[14]. Ryabinin K., Chuprina S., Belousov K. Ontology-driven automation of IOT-based human-machine interfaces development. In Computational Science – ICCS 2019, J. M. F. Rodrigues, P. J. S. Cardoso, J. Monteiro, et al., Eds., Cham: Springer International Publishing, 2019, pp. 110–124, ISBN: 978-3-030-22750-0.

[15]. Mendez S.J.R., Zao J.K. BCI ontology: A context-based sense and actuation model for brain-computer interactions. In Proc. of the 9th International Semantic Sensor Networks Workshop co-located with 17th International Semantic Web Conference, 2018, pp. 32-47.

[16]. Ryabinin K., Chuprina S. High-level toolset for comprehensive visual data analysis and model validation, Procedia Computer Science, vol. 108, pp. 2090– 2099, Dec. 2017. DOI: 10.1016/j.procs.2017.05.050.

[17]. Ryabinin K., Chuprina S., Kolesnik M. Calibration and monitoring of IOT devices by means of embedded scientific visualization. In Proc/ of International Conference on Conceptual Structures, 2018, pp. 655-668. DOI: 10.1007/978-3-319-93701-4_52.

[18]. Izumigawa C., Taylor B., Sato J. Automated ontology generation. In HCI International 2023 Posters, C. Stephanidis, M. Antona, S. Ntoa, and G. Salvendy, Eds., Cham: Springer Nature Switzerland, 2023, pp. 433–438, ISBN: 978-3-031-36004-6.

[19]. Elnagar S., Yoon V.Y., Thomas M.A. An automatic ontology generation framework with an organizational perspective. CoRR, vol. abs/2201.05910, 2022. arXiv: 2201.05910.

[20]. Sassi N., Jaziri W., Gargouri F. How to evolve ontology and maintain its coherence - a corrective operations-based approach. In Proc. of the First International Conference on Knowledge Engineering and Ontology Development, 2009, pp. 384–387.

[21]. Jaziri W., Sassi N., Gargouri F. Approach and tool to evolve ontology and maintain its coherence. International Journal of Metadata Semantics and Ontologies, vol. 5, pp. 151–166, May 2010. DOI: 10.1504/IJMSO. 2010.033284.

[22]. Рябинин К.В. Методы и средства разработки адаптивных мультиплатформенных систем визуализации научных экспериментов. Диссертация на соискание ученой степени кандидата физико-математических наук. ИПМ им. М.В. Келдыша, Москва, 2015, 208 с./ Ryabinin K. Metody i sredstva razrabotki adaptivnykh mul'tiplatformennykh sistem vizualizatsii nauchnykh eksperimentov. PhD Thesis, IPM im. M.V. Keldysha, Moskva, 2015, 208 p. (in Russian).

[23]. Seydoux N., Drira K., Hernandez N., Monteil T. IOT-O, a core-domain IOT ontology to represent connected devices networks. In Knowledge Engineering and Knowledge Management, E. Blomqvist, P. Ciancarini, F. Poggi, and F. Vitali, Eds., Cham: Springer International Publishing, 2016, pp. 561–576, ISBN: 9783-319-49004-5.

[24]. Чуприна С.И., Зиненко Д.В. Онтолис: Адаптируемый визуальный редактор онтологий. Вестник Пермского Университета. Серия: Математика. Механика. Информатика, том 3 (22), 2013, с. 106-110. / Chuprina S.I., Zinenko D.V. Adaptable Visual Ontological Editor ONTOLIS. Vestnik Permskogo universiteta. Seriya: Matematika. Mekhanika. Informatika, vol. 3 (22), 2013, pp. 106–110 (in Russian).

[25]. Chuprina S., Nasraoui O. Using ontology-based adaptable scientific visualization and cognitive graphics tools to transform traditional information systems into intelligent systems. Scientific Visualization, vol. 8 (1), pp. 23–44, Jan. 2016.

[26]. Лабутин И.А. Система генерации программного обеспечения для устройств интернета вещей на базе онтологического описания инфраструктуры. 17.01.2023, РОСПАТЕНТ №2023612016 / Labutin I.A. Sistema generatsii programmnogo obespecheniya dlya ustroistv interneta veshchei na baze ontologicheskogo opisaniya infrastruktury. 17.06.2023. ROSPATENT №2023612016 (in Russian).

Информация об авторах / Information about authors

Светлана Игоревна ЧУПРИНА – кандидат физико-математических наук, профессор кафедры математического обеспечения вычислительных систем ПГНИУ, почетный работник высшего профессионального образования РФ, член-корреспондент Международной академии информатизации. Сфера научных интересов: базы данных и базы знаний, экспертные системы в составе СППР, интернет вещей, семантический веб, методы и средства онтологического инжиниринга для построения интеллектуальных систем, научная визуализация, машинное обучение и генеративный ИИ в задачах автоматической обработки текстов.

Svetlana Igorevna CHUPRINA – Cand. Sci. (Phys.-Math.), Prof. of Computer Science Dept. at Perm State University, Honorary Worker of Higher Professional Education of the Russian Federation, Corresponding Member of the International Academy of Informatization. Research interests: databases and knowledge bases; expert systems as a part of the DSS, the Internet of Things, semantic

web, methods and tools of ontological engineering for building intelligent systems, scientific visualization, machine learning and generative AI in NLP.

Иван Александрович ЛАБУТИН является аспирантом кафедры математического обеспечения вычислительных систем ПГНИУ. Его научные интересы включают сферу интернета вещей и вопросы интеграции нейроинтерфейсов в экосистему интернета вещей, обработку естественного языка, системное программирование.

Ivan Aleksandrovich LABUTIN is a postgraduate student of the Computer Science Department at Perm State University. His research interests include IoT and approaches for integrating neurointerfaces with IoT infrastructure, natural language processing, system programming.

DOI: 10.15514/ISPRAS-2024-36(2)-9

Набор методических тестовых программ для численного исследования параметров высокопроизводительных вычислительных систем

А.О. Игнатьев, ORCID: 0000-0003-4902-2123 <*a.o.ignatyev@vniitf.ru*>
С.Ю. Мокшин, ORCID: 0000-0002-7454-6597 <*s.yu.mokshin@vniitf.ru*>
А.В. Ершов, ORCID: 0009-0007-4439-1516 <*a.v.ershov@vniitf.ru*>
А.В. Карпеев, ORCID: 0009-0007-2203-9423 <*a.v.karpeev@vniitf.ru*>
Р.Ф. Мухамадиев, ORCID: 0009-0006-2059-2113 <*r.f.mukhamadiev@vniitf.ru*>
Е.М. Романова, ORCID: 0009-0006-1478-9059 <*e.m.romanova@vniitf.ru*>
Д.А. Ушаков, ORCID: 0009-0001-6458-2631 <*d.a.ushakov@vniitf.ru*>
В.О. Анисов, ORCID: 0009-0004-5620-099X <*v.o.anisov@vniitf.ru*>

Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина,
456770, Россия, г. Снежинск, Челябинская область, ул. Васильева, 13.

Аннотация. Неотъемлемой частью процесса создания высокопроизводительных вычислительных систем, предназначенных для решения задач численного моделирования различных физических процессов, является проверка их соответствия на заявленные при их проектировании характеристики. При этом существует проблема оценки производительности вычислительных систем на синтетических тестах, значительно уступающих по математической сложности реальным прикладным задачам. В статье рассматривается разработанный авторами набор тестовых программ, позволяющий более точно оценивать реальную производительность вычислительных систем.

Ключевые слова: вычислительная система; высокопроизводительные вычисления; математическое моделирование; тестирование вычислительных систем.

Для цитирования: Игнатьев А.О., Мокшин С.Ю., Ершов А.В., Карпеев А.В., Мухамадиев Р.Ф., Романова Е.М., Ушаков Д.А., Анисов В.О. Набор методических тестовых программ для численного моделирования параметров высокопроизводительных вычислительных систем. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 109–126. DOI: 10.15514/ISPRAS–2024–36(2)–9.

A Set of Methodological Test Programs for the Numerical Research of High-Performance Computing Systems Parameters

A.O. Ignatyev, ORCID: 0000-0003-4902-2123 <*a.o.ignatyev@vniitf.ru*>
S.Yu. Mokshin, ORCID: 0000-0002-7454-6597 <*s.yu.mokshin@vniitf.ru*>
A.V. Ershov, ORCID: 0009-0007-4439-1516 <*a.v.ershov@vniitf.ru*>
A.V. Karpeev, ORCID: 0009-0007-2203-9423 <*a.v.karpeev@vniitf.ru*>
R.F. Mukhamadiev, ORCID: 0009-0006-2059-2113 <*r.f.mukhamadiev@vniitf.ru*>
E.M. Romanova, ORCID: 0009-0006-1478-9059 <*e.m.romanova@vniitf.ru*>
D.A. Ushakov, ORCID: 0009-0001-6458-2631 <*d.a.ushakov@vniitf.ru*>
V.O. Anisov, ORCID: 0009-0004-5620-099X <*v.o.anisov@vniitf.ru*>

Russian Federal Nuclear Center –
Zababakhin All-Russian Research Institute of Technical Physics,
456770, Russia, Snezhinsk, Chelyabinsk region, Vasilieva street, 13.

Abstract. An integral part of the process of creating high-performance computing systems designed to solve problems of numerical modeling of various physical processes is to check their compliance with the characteristics stated during their design. At the same time, there is a problem of evaluating the performance of computing systems on synthetic tests, which are significantly primitive in mathematical complexity to real applied problems. The article considers a set of test programs developed by the authors, which allows to more accurately assess the real performance of computing systems.

Keywords: computing system; high-performance computing simulation; mathematical modeling; high-performance computing system testing.

For citation: Ignatyev A.O., Mokshin S.Yu., Ershov A.V., Karpeev A.V., Mukhamadiev R.F., Romanova E.M., Ushakov D.A., Anisov V.O. A set of methodological test programs for the numerical research of high-performance computing systems parameters. Trudy ISP RAN/Proc. ISP RAS, vol. 36, issue 2, 2024. pp. 109-126 (in Russian). DOI: 10.15514/ISPRAS-2024-36(2)-9.

1.Введение

При решении задач численного моделирования физических процессов требуется огромный объем вычислений, обеспечиваемых высокопроизводительными вычислительными системами (ВВС). Современные ВВС, предназначенные для решения задач численного моделирования, представляют собой множество вычислительных узлов, систему хранения с параллельным доступом и сервисные подсистемы, объединяемых высокопроизводительной коммуникационной средой [1]. При проектировании таких ВВС закладываются ожидаемые характеристики по производительности вычислений, основанные на физических показателях используемых элементов и используемой архитектуре ВВС. Однако, как показывает практика, переход от одной ВВС к другой с большей заявленной производительностью не гарантирует кратного сокращения сроков решаемых задач. Даже традиционно применяемый в рейтинге Супер-ЭВМ TOP-500 [2] тест High Performance Linpack (HPL) [3] уже не подтверждает заявленную производительность ВВС, так как его математический аппарат скорее типичен для вычислительных систем начала нашего века. Поэтому во всем мире, помимо предложенного Д. Донгарра нового теста HPCG [4], использующего для решения систем линейных алгебраических уравнений метод сопряженных градиентов, разрабатываются специализированные пакеты тестовых программ, реализующие различные численные алгоритмы и позволяющие оценить производительность ВВС применительно к данным алгоритмам.

Одним из первых подобных пакетов можно считать NAS Parallel Benchmark (NPB) [5], содержащий на момент его создания 8 параллельных программ, представляющих различные численные алгоритмы и использующих MPI [6] для распараллеливания. Этот пакет

продолжает развиваться, расширяется набор программ, добавляются новые алгоритмы и новые способы распараллеливания (OpenMP [7], CUDA [8]). Следующим шагом можно считать набор тестовых программ министерства энергетики США, используемых при создании Супер-ЭВМ ASC Purple и следующих за ней Супер-ЭВМ [9], содержащий упрощенные версии производственных программ. Существуют и более современные версии подобных тестовых программ, например, HPC (High Performance Computing) benchmarks [10].

В России, на предприятиях государственной корпорации «Росатом», также ведутся работы по созданию пакетов тестовых программ, максимально приближенных к используемым на этих предприятиях программам численного моделирования различных физических процессов. В результате разработан комплекс тестов, применяемых при создании и проведении приемочных испытаниях различных ВВС, включающих в себя наряду с рядом тестов из NPB (LU, SP, BT) и ASCI Purple (sPPM, IOR) методические тестовые программы, разработанные в РФЯЦ-ВНИИЭФ и РФЯЦ-ВНИИТФ [11].

В данной статье приводится описание набора методических тестовых программ, разработанных в РФЯЦ-ВНИИТФ и входящих в общий комплекс тестов государственной корпорации «Росатом».

2. Общая характеристика методических тестовых программ

Набор методических тестовых программ представляет широкий класс вычислительных алгоритмов, используемых при численном моделировании физических процессов и, в сочетании с другими тестовыми программами, позволяющих прогнозировать поведение реальных программ численного моделирования.

Требования к методическим тестовым программам узаконены в российском стандарте [12], принятом в 2019 году.

Методические тестовые программы удовлетворяют следующим условиям:

- используются языки программирования Фортран и Си (Си++);
- используется многоуровневое распараллеливание: OpenMP внутри вычислительного узла и MPI между узлами;
- программы готовят необходимые начальные данные для проведения расчета и выполняют расчет тестовой задачи;
- масштабирование задачи в программах производится как методом деления (единая вычислительная нагрузка на все потоки выполнения в пределах одного вычислительного узла), так и методом умножения (вычислительная нагрузка увеличивается кратно увеличению количества процессов при использовании множества вычислительных узлов);
- программы верифицируют полученные результаты;
- программы определяют полученную производительность в операциях или других единицах.

3. Описание методических тестовых программ

3.1 Тест SLON

Параллельная программа SLON решает двумерное стационарное уравнение переноса для сферических систем в односкоростном кинетическом приближении, записанное в цилиндрической системе координат и вычисляет критический параметр λ. Программа

предназначена для проведения исследований производительности и эффективности распараллеливания на многопроцессорных ЭВМ с распределенной и общей памятью.

3.1.1 Математическая модель

Математическая модель программы представляет собой решение однородной краевой задачи:

$$div(\vec{\Omega}N_g) + (\alpha_g + \frac{\lambda}{\upsilon_g})N_g = \frac{1}{2\pi}\sum_{g'=1}^{G}\int_{\Omega}\beta_{g'g}\,N_g d\Omega$$

$$N_g = N_g(r,z,\mu,\psi)\big|_{\Gamma} = 0, \text{ при } \overline{\Omega}\overline{n} < 0.$$

где:
r, z – цилиндрические координаты точки, в которой решается задача;
Г – граница системы, в которой решается задача;
$N_g(r,z,\mu,\psi)$ – число нейтронов, летящих в направлении вектора единичной длины $\overline{\Omega}(\mu,\psi)$;
υ_g – скорость нейтронов группы g относительно вещества;
g – индекс группы;
G – количество рассматриваемых групп, число групп задается как параметр распараллеливания при счете задачи, скорости в группах одинаковые;
$\alpha_g = \alpha_g(r,z)$ – коэффициент поглощения нейтронов;
$\beta_{g'g} = \beta_{g'g}(r,z)$ – коэффициент рассеяния нейтронов из группы g' в группу g;
$\overline{n}$ – нормаль к стороне ячейки;
ψ– угол между $\overline{\Omega'}$ и осью $\vec{r}$;
$\overline{\Omega'} = \xi\vec{n}_r + \mu\vec{n}_z$– проекция $\overline{\Omega}$ на плоскость (r,z);
$\mu = cos\,\theta\,, \xi = cos\,\psi\sqrt{1-\mu^2}, \eta = sin\,\psi\sqrt{1-\mu^2}$;
θ – угол между $\overline{\Omega}$ и осью $\vec{z}$.

Для указанной задачи существует положительная собственная функция N^*, соответствующая характеристическому числу λ^*, которое по модулю больше других характеристических чисел λ_k. Решение однородной нестационарной задачи (без источников и с нулевыми граничными условиями) при $t \to \infty$ будет пропорционально $N_g^* e^{\lambda t}$, т. е. λ^* представляет собой временную постоянную размножения нейтронов [13]. Уравнение переноса решается Sn-методом по DDST-схеме [14] на четырехугольных сетках. На внутренних итерациях по интегралу столкновений используется метод Келлога [15]. На внешних итерациях используется метод Ньютона-Рафсона (модификация метода Ньютона [16]).

Программа написана на языке FORTRAN и содержит 4 уровня распараллеливания по параметрам фазового пространства задачи, реализованных на распределенной памяти средствами MPI и на общей памяти средствами OpenMP. Рассмотрим основные особенности каждого уровня распараллеливания и используемые при этом декомпозиции.

3.1.2 Декомпозиция по энергетическим группам

Многогрупповой режим моделируется с помощью размножения задачи по процессам MPI, без изменения скорости полета нейтронов. Такой подход сохраняет все арифметические операции, необходимые для реализации разностной схемы и позволяет оценить накладные расходы на межпроцессорные обмены данными. При этом естественным образом используется метод увеличения задачи, полностью сбалансированный по процессам MPI. Система нейтронно-ядерных констант в программе SLON не используется, решения на каждом процессе дублируются.

3.1.3 Геометрическая декомпозиция по областям системы

Уровень реализован средствами MPI, используется метод увеличения, реализованный с помощью увеличения числа точек в системе. В этом случае подобласть системы в одном процессе MPI будем называть блоком. Для совпадения решения такой задачи в параллельном режиме с решением в последовательном режиме требуется больше итераций. Метод обеспечивает примерно одинаковое число арифметических операций в каждом счетном блоке, что позволяет вычислять эффективность распараллеливания достаточно точно.

3.1.4 Распараллеливание по направлениям в пространстве полета нейтронов

Уровень реализован методом дробления, средствами OpenMP на общей памяти вычислительного узла ВВС. Позволяет ввести практически любую квадратуру для увеличения арифметической нагрузки на узел [17].

3.1.5 Мелкозернистое распараллеливание

Уровень реализован за счет дополнительной геометрической декомпозиции методом дробления блока средствами OpenMP для каждого MPI процесса. Блок системы на каждом MPI процессе разбивается на дополнительное число счетных областей с сохранением числа точек в блоке. Обмен граничными условиями между соседними областями производится во время счета итерации. Данный уровень распараллеливания предполагает решение на одном OpenMP потоке уравнения переноса для одной счетной области и части направлений полета нейтронов, что позволяет менять конфигурацию счетного сегмента фазового пространства и исследовать возможность наилучшего попадания задачи в быструю память узла.

3.1.6 Получение отчетных показателей

В качестве отчетных показателей в тестовой программе SLON используются следующие измеряемые параметры:

- эффективность распараллеливания;
- производительность задачи в вещественных и целых операциях в секунду.

При этом время исполнения программы делится на 2 этапа:

- 1-й этап – подготовка начальных данных и упорядочивание системы. Время исполнения этого этапа не учитывается при измерении эффективности, поскольку эффективность распараллеливания на этом этапе составляет 100% (обменов между процессами нет) и, при небольшом числе итераций на 2-м этапе счета, может неоправданно завысить эффективность распараллеливания задачи.
- 2-й этап – счет итераций. Времена выполнения 2-го этапа на одном вычислительном узле и на множестве узлов используются для вычисления эффективности распараллеливания. Обычно задается число итераций = 100, что обеспечивает сходимость четырех значащих цифр параметра λ. В случае необходимости число итераций можно увеличить до сходимости решения.

3.2 Тест GROM

Параллельная программа GROM численно решает двумерное нестационарное квазилинейное уравнение теплопроводности для плоского случая или случая с осевой симметрией для задачи [18-19], которая имеет аналитическое решение в виде бегущей волны. Программа GROM позволяет проводить расчеты на вычислительных системах, как в однопроцессорном, так и в параллельном режиме. Программа предназначена для проведения исследований

производительности и эффективности распараллеливания на ВВС с распределенной и общей памятью.

3.2.1 Математическая модель

В двумерной области $r \in [0; r_0], z \in [0; z_0], t \in [t^0; t^n]$ рассматривается краевая задача для двумерного уравнения теплопроводности без учета независимого источника:

$$\frac{\partial \varepsilon}{\partial t} = \frac{1}{\rho R}\left[\frac{\partial}{\partial r}\left(R \cdot \chi \frac{\partial T}{\partial r}\right) + \frac{\partial}{\partial z}\left(R \cdot \chi \frac{\partial T}{\partial z}\right)\right]$$

где:
r, z – декартовы координаты;
$R=1$ – плоский случай;
$R=r$ – осевая симметрия (z – ось симметрии);
$T(t,r,z)$ – температура;
ρ – плотность;
$\varepsilon(\rho,T)$ – внутренняя энергия;
$\chi(\rho,T)$ – коэффициент теплопроводности.
В рассматриваемой задаче $\rho = 1, \varepsilon = T, \chi = \chi_0 T^{\alpha}$, где $\chi_0 > 0, \alpha > 0$,
и рассматривается с начальными условиями:

$$T(t^0, r, z) = T^0(r, z)$$

и граничными условиями (теплоизолирующая стенка):

$$T(t, r, z = 0) = \left(\frac{\alpha c^2}{\chi_0} t\right)^{1/\alpha}$$

где c – скорость распространения тепловой волны.
Задача имеет автомодельное решение, представляющее собой бегущую волну [18]:

$$T(t, r, z) = \begin{cases} \left(\frac{\alpha c}{\chi_0}(ct - z\right)^{1/\alpha}, z < ct, \\ 0, z \geq ct \end{cases}$$

Уравнение $\frac{\partial \varepsilon}{\partial t}$ аппроксимируется неявной, безусловно устойчивой конечно-разностной схемой «РОМБ» [18,19,20]. Для решения получаемой системы разностных уравнений применяется итерационный метод со стабилизирующей поправкой (ИМСП) [19], который сводит многомерную разностную задачу к совокупности квазиодномерных подзадач, решаемых прогонками по каналам. Итерации метода ИМСП совмещаются с итерациями по нелинейности χ. По нелинейности χ применяется метод простой итерации (μ-итерации). Этот метод позволяет вести счет с крупным шагом по времени, при этом схема обладает свойством полной аппроксимации. Счет на одной итерации μ разбивается на два этапа. На первом этапе счет ведется вдоль каналов $j + 1/2 = const$, на втором – вдоль каналов $i + 1/2 = const$. Причем вторая система может быть решена только после решения первой.

3.2.2 Программная реализация

В программе GROM счетная область разделяется на прямоугольные в топологическом плане подобласти, каждая подобласть обрабатывается отдельным MPI процессом. Поскольку шаблон методики «РОМБ» строится в рамках одной ячейки, используются непересекающиеся подобласти. Обмены данными происходят лишь между «соседними» MPI процессами. В управляющем файле программы GROM пользователь задает конфигурацию

разбиения Numr×Numz счетной области, где Numr и Numz – количество MPI процессов по оси r и z соответственно. При этом данные равномерно распределяются между MPI процессами. Схематично декомпозицию счетной области можно представить так, как показано на рис. 1.

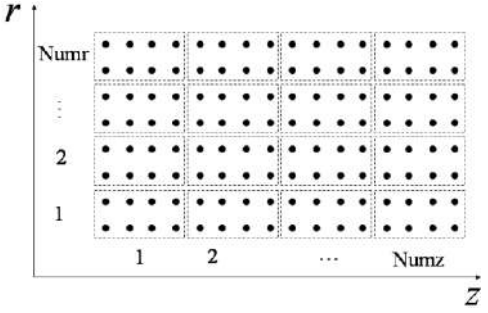


Рис. 1. Декомпозиция счетной области
Fig. 1. Calculation area decomposition

Программа GROM распараллелена по двум уровням:

- Первый уровень – геометрическая декомпозиция по пространственным переменным. Для этого система разбивается на подобласти так, как указано выше. Распараллеливание по подобластям производится на распределенной памяти средствами MPI. На каждом этапе итерации по нелинейности коэффициента теплопроводности происходит обмен массивами граничных условий между соседними подобластями. Число подобластей является задаваемым параметром. Количество подобластей должно быть кратно числу задействованных в расчете узлов ВВС.
- Второй уровень распараллеливания – совокупность каналов $j + 1/2 = const$ или $i + 1/2 = const$ в зависимости от этапа. Данный уровень распараллеливания реализован на общей памяти при помощи библиотеки OpenMP. Число потоков является входным параметром и не должно превосходить числа ядер одного узла, тестируемого ВВС.

Векторизация вычислений – неявная, средствами используемого компилятора.

3.2.3 Получение отчетных показателей

Программа GROM предназначена для тестирования различных ВВС. При тестировании задаются параметры задачи, количество узлов, количество MPI процессов и количество потоков на вычислительном узле в управляющем файле конфигурации программы. При этом измеряемыми параметрами являются:

- эффективность распараллеливания;
- производительность ЭВМ в операциях в секунду над вещественными числами с двойной точностью.

3.3 Тест MHT

Тестовая программа MHT (Monte-Carlo High Performance Computing Test) предназначена для оценки производительности ВВС. Программа MHT использует двухуровневую схему распараллеливания: между узлами с помощью MPI, внутри узла – на общей памяти с помощью потоков стандартной библиотеки C++11. Геометрия системы представляет собой трехмерную сетку из произвольных шестигранников. Основным результатом работы теста

является условная производительность, представляющая собой число поколений в секунду (скорость счета).

3.3.1 Математическая модель

Программа МНТ имитирует расчет на произвольных гексагональных сетках с использованием метода максимального сечения. Тестовая задача: расчет Keff-методом поколений с фиксированным количеством точек деления в поколении. Модель среды: приближение свободных покоящихся атомов. Константы: спектральные, на основе ENDF/B-5. Свободный пробег моделируется на прямоугольной сетке виртуальных областей 20x20x20.

Зависимость максимального сечения от энергии задается в виде таблиц с логарифмическим шагом по энергии. Число узлов 150. В точке соударения, производится поиск текущей ячейки, разыгрываются тип и параметры взаимодействия. В процессе поиска ячейки сетки используется ускоряющая структура – kD-дерево, равномерно разбивающая пространство (рис. 2, 3).

В ее узлах расположены по 8 ячеек, воспроизводящих нагрузку на операцию проверки принадлежности.

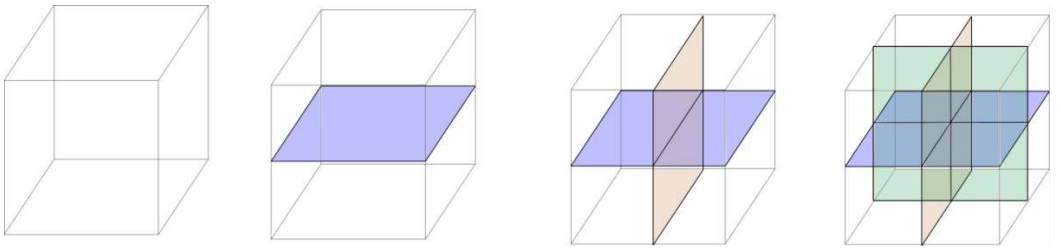


Рис. 2. Разбиение пространства
Fig. 2. Space partitioning

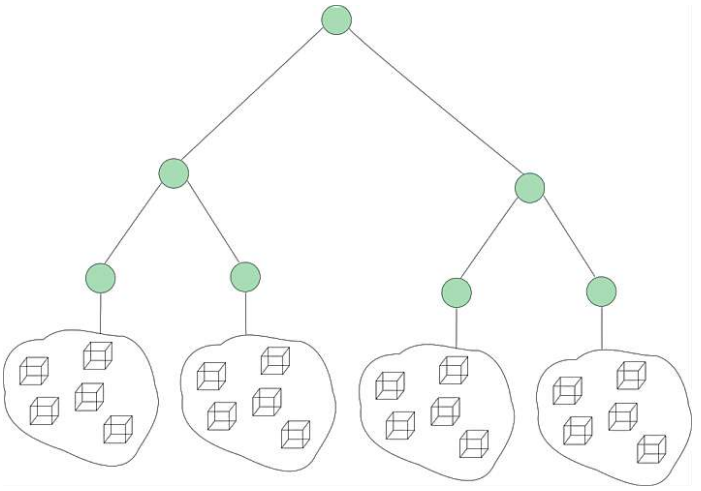


Рис. 3. kD-дерево
Fig. 3. kD-tree

Для неравномерных сеток (например, Лагранжево-Эйлеровых методик) к узлам дерева может быть привязано большое количество ячеек из-за того, что их невозможно разделить с помощью простых ограничивающих тел (рис. 4). В этом случае может существенно вырасти

время, приходящееся на работу геометрического модуля. В текущей версии программы количество ячеек, привязанных к узлу, фиксировано и равно 8. Имитация нагрузки, возникающей из-за неравномерности сеток, реализована путем искусственного увеличения количества выполняемых операций.

Система представляет собой критическую сборку типа Godiva [21], окруженную слоем воды (рис. 5). Задание системы производится с помощью указания материалов в ячейках сетки.

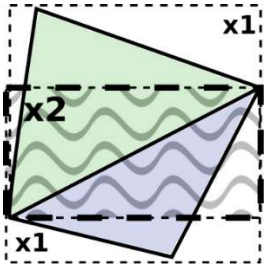


Рис. 4. Проблема разделимости ячеек
Fig. 4. Cell separation problem

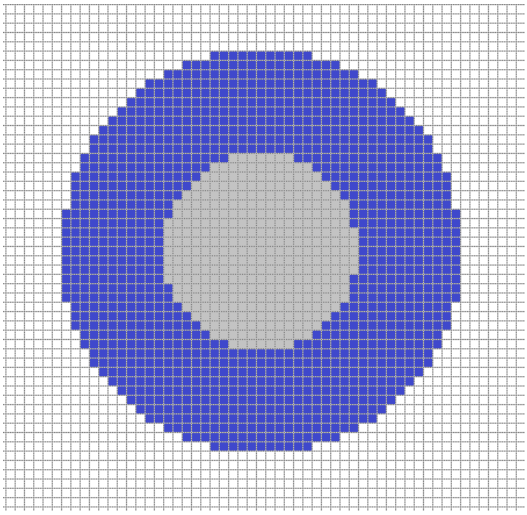


Рис. 5. Задание системы
Fig. 5. System definition

3.3.2 Программная реализация

Программа МНТ написана на языке программирования C++, при этом используется следующая схема распараллеливания:

- обмены между процессами средствами библиотеки MPI;
- разбиение алгоритмов на потоки средствами стандартной библиотеки языка.

3.3.3 Получение отчетных показателей

Измеряемыми параметрами являются:

- Generations/sec – количество рассчитываемых поколений за одну секунду;
- Generations processed – суммарное количество промоделированных поколений;

- Mean calculation time – среднее время счета активных поколений на узле;
- Skipping finished – время, затрачиваемое на отброс первых 30 поколений на каждом вычислительном потоке;
- RND time – время инициализации расчета;
- Calc time – время работы счетной части (включает время Skipping finished);
- Memory usage – объём памяти, занимаемый геометрией;
- Cells count – количество ячеек.

Основным оценочным рассчитываемым параметром является количество рассчитываемых поколений в секунду Generations/sec – условная производительность.

3.4 Тест Machete

Тестовая параллельная программа моделирует движение сплошной среды, решая для этого систему уравнений идеальной газовой динамики в Эйлеровых координатах на Декартовой кубической сетке. Программа позволяет проводить расчеты на ВВС, как в последовательном режиме, так и с использованием произвольных комбинаций следующих подходов распараллеливания: векторизация, OpenMP, MPI. Программа предназначена для проведения исследований производительности и эффективности распараллеливания заложенных в нее численных алгоритмов на современных вычислительных системах.

3.4.1 Математическая модель

Для моделирования движения среды используется система уравнений идеальной газовой динамики в эйлеровых переменных:

$$\frac{\partial \rho}{\partial t} + div(\rho U) = 0,$$
$$\frac{\partial \rho U}{\partial t} + div(\rho U \cdot U) + grad(P) = 0,$$
$$\frac{\partial \rho E}{\partial t} + div(\rho E U) + grad(PU) = 0.$$

где:

ρ – плотность;

$U=(u,v,w)$ – вектор скорость;

ε – удельная внутренняя энергия;

$E = \varepsilon + 0.5U^2$ – удельная полная энергия;

P – давление вещества, которое связано с плотностью и энергией соотношением $P = P(\rho,\varepsilon)$.

В качестве тестовой выбрана задача «о точечном взрыве». В данной задаче моделируется сферически-симметричное течение, возникающее в результате точечного мгновенного энерговыделения в изначально холодном однородном газе без противодавления. Аналитическое решение для данной задачи получено Л.И. Седовым [22].

Для аппроксимации дифференциальных уравнений область интегрирования задачи разбивается равномерной кубической сеткой на Nx×Ny×Nz ячеек. Количество ячеек по направлению (линейный размер сетки), временной шаг и количество шагов в расчете являются параметрами задачи, которые варьируются в зависимости от исследуемой характеристики и определяются через входные параметры программы. В начальный момент времени распределение величин в ячейках задано. Разностная схема для перехода на следующий временной слой построена на основе FLIC-метода [22, 23].

3.4.2 Программная реализация

Программа MACHETE написана на языке программирования C++. В программе MACHETE реализована возможность использовать три уровня распараллеливания:

- на распределенной памяти средствами MPI;
- на общей памяти средствами OpenMP;
- векторизация.

При организации распараллеливания на распределенной памяти средствами MPI в программе MACHETE использован принцип пространственной декомпозиции счетной области на подобласти одинакового размера. Количество разбиений по каждому из направлений $N_p^{(x)}, N_p^{(y)}, N_p^{(z)}$ является параметром задачи и определяется в конфигурационном файле программы через размеры счетной области в одном MPI-процессе (параметр «box_size»). При этом общее число MPI-процессов в задаче $N_p = N_p^{(x)} \times N_p^{(y)} \times N_p^{(z)}$. Данный подход позволяет получать разбиения области и по одному, и по двум, и по трем направлениям. Такая гибкость дает возможность находить подходящее разбиение для различных комбинаций гибридного распараллеливания с участием MPI. В одном случае может быть оптимальным минимизировать объем пересылаемых сообщений, в другом – поддержать достаточное количество листов для OpenMP, в третьем – максимально использовать последовательность укладки данных для использования предвыборки.

Для согласований решений между процессами используется технология виртуальных ячеек. С целью минимизации количества MPI-вызовов внутри одного временного шага размер виртуальной области имеет ширину в три ячейки. Таким образом, если на процессе рассчитывается «полезная» область размером $n_x \times n_y \times n_z$ ячеек, то размер посылаемых и принимаемых на каждом шаге данных определяется выражением

$$V_{send} = V_{recv} = Nb_{cell} \cdot \left[(n_x + 6) + \left(n_y + 6\right) + (n_z + 6) - n_x \cdot n_y \cdot n_z\right]$$

где:

$NBcell = 56$ байт – объём данных, пересылаемых из одной ячейки.

Возможность распараллеливания на общей памяти в программе MACHETE реализована средствами библиотеки OpenMP. Используется независимый обход внешних циклов 3-го координатного направления. Количество используемых в расчете потоков является входным параметром и определяется через входной параметр задачи. Желательно, чтобы эта величина не превосходила количества ячеек в подобласти по z-направлению и допустимого числа потоков на одном вычислительном узле.

Все основные счетные циклы реализованного в программе MACHETE алгоритма допускают автоматическую векторизацию. Для эффективной реализации такой возможности ячеечные величины упакованы в структуры с последовательным хранением данных относительно внутреннего индекса. В программе MACHETE использованы одномерные N-компонентные массивы, которые в дополнение к автоматической векторизации позволяют минимизировать количество операций на определение адресов переменных для хранения данных рассматриваемой точки разностного шаблона.

3.4.3 Получение отчетных показателей

Тестовая программа MACHETE предназначена для тестирования различных ВВС. При тестировании варьируются параметры задачи, количество MPI-процессов и количество OpenMP потоков на них. Значения этих величин задаются в текстовом файле, имя которого подается в качестве аргумента основной программы.

Измеряемые параметры и методика их получения:

- производительность ВВС (Флопс).
- эффективность распараллеливания.

Количество арифметических операций над вещественными числами, выполняемых программой за один временной шаг, является функцией от линейного размера задачи. Отсюда, с учетом полного времени выполнения программы и количества выполненных временных шагов, определяется производительность перемножением этих величин.

3.5 Тест HCTest

Тестовая параллельная программа HCTEST моделирует процесс нелинейной (лучистой) теплопроводности в цилиндрической системе координат на регулярной прямоугольной или нерегулярной треугольной сетке. Программа позволяет проводить расчеты на ВВС, как в последовательном режиме, так и с использованием произвольных комбинаций следующих подходов распараллеливания: векторизация (средствами компилятора), OpenMP, MPI, CUDA. При этом используется собственная равномерная рассадка по NUMA узлам в рамках вычислительного узла.

3.5.1 Математическая модель

Для моделирования нелинейной теплопроводности используется система уравнений теплопроводности газа в цилиндрических координатах:

$$\rho\frac{d\varepsilon}{dt} + div\ \Phi = \rho Q,$$
$$\Phi + D\ grad\ T = 0,$$
$$\alpha(\mathrm{r},t)T - \beta(\mathrm{r},t)\Phi\mathrm{n} = \mu(\mathrm{r},t), \mathrm{r} \in \Sigma.$$

где:

$\varepsilon(\rho,T)$ – удельная внутренняя энергия;

$\rho(r,T)$ – плотность;

$T(r,t)$ – температура;

$\Phi(T)$ – тепловой поток;

$D(T)$ – коэффициент теплопроводности;

$\alpha(r,t), \beta(r,t), \mu(r,t)$ – тройка величин, задающих граничные условия;

Σ – граница области;

Φn – тепловой поток, умноженный на внешнюю нормаль к границе.

В качестве тестовой задачи выбрана автомодельная задача «о мгновенном точечном источнике». В данной задаче моделируется течение тепла, возникающее от точечного мгновенного теплового источника [24]. Расчетная область от 0 до 3 по оси X и от 0 до 1.5 по оси Y, точечный источник помещен в точку (1.5, 0). Коэффициент теплопроводности рассчитывается по формуле $D(T) = T^2$. В качестве начального профиля используется профиль в момент времени t = 0.05, который изображен на рис. 6.

Для аппроксимации дифференциальных уравнений область интегрирования задачи (расчетная область), покрывается регулярной прямоугольной сеткой или нерегулярной треугольной сеткой (определяется входным параметром). Множество всех ячеек обозначим за Ω, а величины ячейки $i_n \in \Omega$ будем записывать в виде T_{i_n}. Заметим, что нерегулярная сетка больших размеров, более двух миллионов ячеек, строится значительное время, поэтому необходимо её строить особым образом. Нерегулярная сетка получается в результате укладки необходимого числа одного и того же масштабированного квадратного блока (рис. 7), на котором заранее построена треугольная сетка специального вида в области [0,0]×[1,1]. Специальный вид заключается в том, что узлы, лежащие на границе, расположены регулярно,

то есть с одинаковым шагом вдоль. Это делается для того, чтобы во время укладки блоков узлы, лежащие на границе, совпадали.

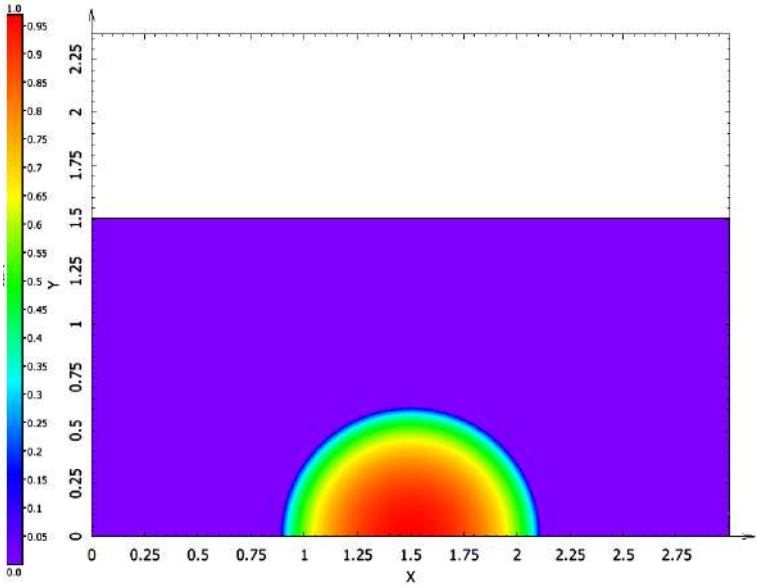
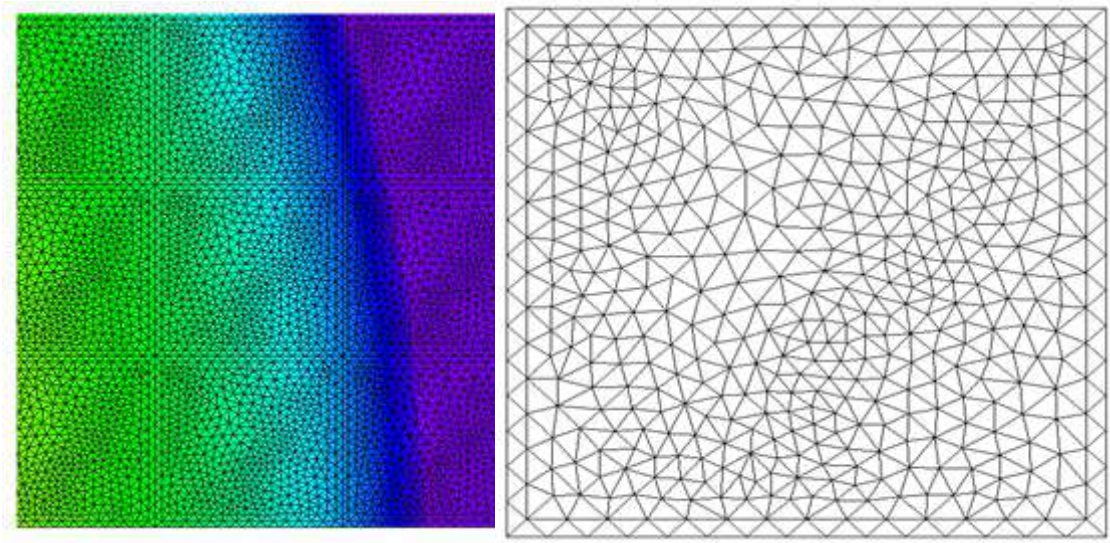


Рис. 6. Начальный профиль температуры
Fig. 6. Initial temperature profile



а) б)

Рис. 7. Пример построения нерегулярной треугольной сетки (а), полученной в результате укладки набора одинаковых блоков (б)
Fig. 7. An example of irregular triangular mesh building (a), laying set of identical blocks (б)

Метод решения можно записать в виде следующего алгоритма.

В начальный момент времени распределение величин в ячейках задано. Для перехода на следующий временной слой используется итерационный процесс по нелинейности внутренней энергии и коэффициентов теплопроводности: положим $T_{i_n}^{\nu} = T_{i_n}^{n}$, найдем $T_{i_n}^{\nu+1}$, для этого запишем уравнение итоговой схемы для каждой ячейки, получив систему линейных алгебраических уравнений (СЛАУ), имеющую вид $Ax = b$. Матрица A и вектор B рассчитываются на основе выбранной схемы. Во время итерационного процесса неизвестный вектор x – это "промежуточный" профиль температуры между временными слоями, а по окончанию, x – это профиль следующего временного слоя. В программе на выбор используются два условия остановки решателя:

$$\|Ax_n - B\|_{L2} \le tol \cdot \|B\|_{L2}$$

$$\|Ax_n - B\|_{L2} \le tol \cdot \|Ax_0 - B\|_{L2}$$

где tol – точность решателя СЛАУ.

Среднее время одной итерации решателя зависит от количества проделанных итераций. Число выполненных итераций ограничено снизу числом 40. Данное число было подобрано для тестовой задачи при разном размере сетки, временном шаге и других параметрах, а также при вычислениях на различном оборудовании.

В результате решения СЛАУ будут получены температуры в центрах ячеек на новой итерации $T_{i_n}^{\nu+1}$. Далее проверяется условие остановки итерационного процесса:

$$\left|T_{i_n}^{\nu+1} - T_{i_n}^{\nu}\right| - \varepsilon\left(1 + T_{i_n}^{\nu}\right) \le 0, i_n \in \Omega, \varepsilon \to 0$$

Если указанное условие не выполнено, тогда повторяем процедуру нахождения температуры на следующей итерации, положив $T_{i_n}^{\nu}$=$T_{i_n}^{\nu+1}$. В противном случае, если это условие выполнено, тогда за температуру на следующем временном слое $T_{i_n}^{n+1}$ принимается $T_{i_n}^{\nu+1}$.

3.5.2 Программная реализация

Для реализации на универсальном процессоре был использован язык программирования C++ с применением технологий распараллеливания OpenMP, MPI и векторизацией средствами компилятора. Для реализации на графических ускорителях используется технология NVidia CUDA.

В программе HCTest используется двухуровневое распараллеливание. Нижний уровень OpenMP и CUDA для независимого обхода внутри циклов, средствами OpenMP и CUDA NVidia соответственно. Верхний уровень MPI – используется геометрическая декомпозиция расчетной сетки.

Возможность распараллеливания на общей памяти в программе HCTEST реализована средствами библиотеки OpenMP и библиотеки CUDA. Используется независимый обход элементов внутри циклов. Далее представлена табл. 1, которая наглядно показывает четыре режима счета и используемые средства распараллеливания на нижнем уровне.

Табл. 1. Нижний уровень распараллеливания алгоритмов для режимов счета
Table 1. The lower parallelization level of algorithms for calculation modes

Алгоритмы / Режим счета	Построение сетки	Моделирование процесса теплопроводности, кроме решения СЛАУ	Решение СЛАУ
CPU	OpenMP	OpenMP	OpenMP
CPUGPU	OpenMP	OpenMP	CUDA
GPU	OpenMP	CUDA	CUDA
HYBRID	OpenMP	OpenMP / CUDA	OpenMP / CUDA

Большинство основных счетных циклов реализованного в программе HCTEST алгоритма допускают автоматическую векторизацию. Для эффективной реализации такой возможности ячеечные величины упакованы в структуры с последовательным хранением данных относительно внутреннего индекса.

3.5.3 Получение отчетных показателей

Тестовая программа HCTEST предназначена для тестирования различных ВВС на основе, как универсальных узлов, так и гибридных узлов с графическими ускорителями (поддерживающими CUDA). В ходе тестирования могут быть получены следующие параметры:

- верификация полученного решения на основе известного точного решения;
- производительность ВВС (Флопс);
- эффективность распараллеливания.

4. Использование методических тестовых программ при проведении испытаний ВВС

При тестировании ВВС с использованием методических прикладных тестов на первом этапе определяется эффективность распараллеливания внутри одного вычислительного узла методом деления (при этом запускается один процесс MPI и множество потоков выполнения OpenMP).

Ускорение при распараллеливании рассчитывается по формуле:

$$Ksp_n = \frac{T_1}{T_n}$$

где:

T_1 – время выполнения расчета тестовой задачи на одном потоке OpenMP;

T_n – время выполнения расчета тестовой задачи на n потоках OpenMP.

Эффективность распараллеливания рассчитывается по формуле (в процентах):

$$E_n = \frac{Ksp_n}{n} * 100$$

На втором этапе производится серия параллельных расчетов на множестве вычислительных узлов методом увеличения. Ускорение при увеличении числа задействованных узлов в случае, если программа умеет определять производительность, рассчитывается по формуле:

$$Ksp_n = \frac{V_n}{V_1}$$

Если программа не определяет производительность, то ускорение рассчитывается по формуле:

$$Ksp_n = \frac{T_1 \times n}{T_n}$$

где:

V_n – производительность при использовании n узлов;

V_1 – производительность при использовании одного узла.

T_1 – производительность при использовании одного узла;

T_n – время выполнения расчета тестовой задачи на n узлах.

5. Заключение

В статье рассмотрены методические тестовые программы, разработанные в РФЯЦ-ВНИИТФ. Вместе с тестовыми программами из состава тестов NPB, ASCI Purple, а также методическими тестовыми программами, разработанными в РФЯЦ-ВНИИЭФ, они составляют комплекс тестовых программ, используемый как в РФЯЦ-ВНИИТФ, так и на других предприятиях госкорпорации «Росатом». Этот комплекс позволяет оценить производительность и реальную эффективность использования ВВС, подобрать оптимальный режим выполнения расчетных программ (число вычислительных узлов и используемых на узлах процессов MPI и потоков OpenMP) в зависимости от размера решаемой задачи, а также выявить возникающие в ходе наладки и эксплуатации ВВС проблемы, влияющие на производительность отдельных компонент и подсистем ВВС.

Авторы выражают надежду, что изложенная в статье информация о составе и функциональных возможностях методических тестовых программ окажется полезной другим специалистам, занимающимся разработкой ВВС.

Список литературы/References

[1]. Игнатьев А.О., Мокшин С.Ю. Типовая архитектура высокопроизводительной вычислительной системы для решения задач численного моделирования, Препринт РФЯЦ-ВНИИТФ № 265, Снежинск, 2020 г., 21 с.
[2]. TOP-500. Available at: https://www.top500.org/ accessed 07.05.2024.
[3]. HPC Linpack. Available at: https://netlib.sandia.gov/benchmark/hpl/, accessed 07.05.2024.
[4]. HPCG. Available at: https://www.hpcg-benchmark.org/, accessed 07.05.2024.
[5]. NAS Parallel Benchmarks. Available at: https://www.nas.nasa.gov/software/npb.html, accessed 07.05.2024.
[6]. Gropp W., Lusk E., Skjellum A. Using MPI: Portable parallel programming with the message-passing interface. – Second edition. – The MIT Press, 1999
[7]. OpenMP. Available at: https://www.openmp.org/, accessed 07.05.2024.
[8]. CUDA Toolkit. Available at: https://developer.nvidia.com/cuda-toolkit/, accessed 07.05.2024.
[9]. ASC Purple. Available at: https://asc.llnl.gov/computers/historic-decommissioned-machines/purple, accessed 07.05.2024.
[10]. High Performance Computing benchmarks. Available at: https://openbenchmarking.org/suite/pts/hpc, accessed 07.05.2024.
[11]. Алексеев А.В., Беляев С.П., Бочков А.И. и др. Методические прикладные тесты РФЯЦ-ВНИИЭФ для численного исследования параметров высокопроизводительных вычислительных систем. ВАНТ, сер. Математическое моделирование физических процессов. 2020. Вып. 2., с. 86-100.
[12]. ГОСТ Р 57700.18-2019. Высокопроизводительные вычислительные системы. Требования к тестовым программам приемочных испытаний.
[13]. Carlson B.G. A method of characteristics and other improvements in solutions methods for the transport equation. Nuclear Science and Engineering. 1976, №3, p. 408-425.
[14]. Басс Л.П., Волощенко А.М., Гермогенова Т.А. Методы дискретных ординат в задачах о переносе излучения. Сборник, ИПМ им. М.В. Келдыша АН СССР, 1986.
[15]. Владимиров В.С. Математические задачи односкоростной теории переноса частиц. Труды МИАН СССР, 1961.
[16]. Корн Г., Корн Т. Справочник по математике для научных работников и инженеров – М.: Наука, 1970.
[17]. Лебедев С.Н., Писарев В.Н., Романова Е.М. и др. Параллельные вычисления при решении стационарного уравнения переноса. Международный семинар “Супервычисления и математическое моделирование”, Тезисы, Саров, 2000.
[18]. Самарский А.А. Теория разностных схем. – 3-е изд., испр. – М.: Наука, 1989. – 616 с.
[19]. Гаджиев А.Д., Писарев В.Н., Рыкованов В.В., Шестаков А.А. Методика и программа для решения двумерного уравнения теплопроводности. ВАНТ. Методики и программы численного решения задач математической физики, 1985, вып. 1, с. 53-65.

[20]. Писарев В.Н. О параметрическом семействе схем «РОМБ» для нелинейного уравнения теплопроводности. ВАНТ. Методики и программы численного решения задач математической физики, 1986, вып. 3, с. 44-52.
[21]. Модестов Д.Г. Оценка временной постоянной при учете запаздывающих нейтронов. ВАНТ, 2022, вып.1, с 17-26.
[22]. Седов Л.И. Методы подобия и размерности в механике. – 8-ое, переработанное изд. – М.: Наука, 1977.
[23]. Gentry R.A., Martin R.E., Daly B.J. An eulerian differencing method for unsteady compressible flow problems // Journal of Computational Physics. – 1966. – vol. 1.
[24]. Зельдович Я.Б., Райзер Ю.П. Физика ударных волн и высокотемпературных гидродинамических явлений. М. Наука, 1966г, 688стр.

Информация об авторах / Information about authors

Алексей Олегович ИГНАТЬЕВ – научный сотрудник Федерального государственного унитарного предприятия «Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина» с 1998 года. Сфера научных интересов: проектирование вычислительных систем, разработка параллельных программ численного моделирования, разработка операционных систем, методы и средства защиты информации.

Alexey Olegovich IGNATYEV – The scientific assistant of Russian Federal Nuclear Center E. I. Zababakhin «All-Russian Scientific Research Institute of Technical Physics» since 1998. Research interests: design of supercomputer systems, parallel numerical simulation programs development, operating systems development, methods and means of information security.

Сергей Юрьевич МОКШИН – научный сотрудник Федерального государственного унитарного предприятия «Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина» с 2016 года. Сфера научных интересов: проектирование вычислительных систем, разработка функциональных подсистем для высокопроизводительных вычислительных систем, разработка операционных систем, методы и средства защиты информации.

Sergey Yurievich MOKSHIN – The scientific assistant of Russian Federal Nuclear Center E. I. Zababakhin «All-Russian Scientific Research Institute of Technical Physics» since 2016. Research interests: design of supercomputer systems, development of functional subsystems for high performance supercomputing systems, operating systems development, methods and means for protecting information.

Александр Викторович ЕРШОВ – научный сотрудник Федерального государственного унитарного предприятия «Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина» с 2020 года. Сфера научных интересов: разностные методы решения интегро-дифференциальных уравнений, разработка параллельных программ численного моделирования.

Alexander Victorovich ERSHOV — The scientific assistant of Russian Federal Nuclear Center E. I. Zababakhin «All-Russian Scientific Research Institute of Technical Physics» since 2020. Research interests: Difference methods of solution integrate-differential equations, parallel numerical simulation programs development.

Артем Владимирович КАРПЕЕВ – научный сотрудник Федерального государственного унитарного предприятия «Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина» с 2015 года. Сфера научных интересов: численные методы решения уравнения механики

сплошной среды, параллельные технологии для современных высокопроизводительных вычислительных систем.

Artem Vladimirovich KARPEEV – The scientific assistant of Russian Federal Nuclear Center E. I. Zababakhin «All-Russian Scientific Research Institute of Technical Physics» since 2015. Research interests: numerical methods for solving the equation of continuum mechanics, parallel technologies for modern high-performance computing systems.

Рим Фанавиевич МУХАМАДИЕВ – научный сотрудник Федерального государственного унитарного предприятия «Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина» с 2012 г. Сфера научных интересов: разработка параллельных программ численного моделирования

Rim Fanavievich MUKHAMADIEV – The scientific assistant of Russian Federal Nuclear Center E. I. Zababakhin «All-Russian Scientific Research Institute of Technical Physics» since 2012. Research interests: development of parallel numerical simulation programs.

Елена Михайловна РОМАНОВА – научный сотрудник Федерального государственного унитарного предприятия «Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина» с 2015 года. Сфера научных интересов: уравнения переноса частиц, разработка алгоритмов решения уравнения переноса частиц для современных вычислительных платформ.

Elena Mikhailovna ROMANOVA – The scientific assistant of Russian Federal Nuclear Center E. I. Zababakhin «All-Russian Scientific Research Institute of Technical Physics» since 2015. Research interests: particle transport equations, development of algorithms for solving the particle transport equation for modern computing platforms.

Денис Александрович УШАКОВ – научный сотрудник Федерального государственного унитарного предприятия «Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина» с 2015 года. Сфера научных интересов: уравнения диффузии, разработка алгоритмов решения уравнения диффузии для современных вычислительных платформ.

Denis Alexandrovich USHAKOV – The scientific assistant of Russian Federal Nuclear Center E. I. Zababakhin «All-Russian Scientific Research Institute of Technical Physics» since 2015. Research interests: diffusion equations, development of algorithms for solving the diffusion equation for modern computing platforms.

Вадим Олегович АНИСОВ – научный сотрудник Федерального государственного унитарного предприятия «Российский Федеральный Ядерный Центр – Всероссийский научно-исследовательский институт технической физики имени академика Е.И. Забабахина» с 2018 года. Сфера научных интересов: уравнения диффузии, разработка алгоритмов решения уравнения диффузии для современных вычислительных платформ.

Vadim Olegovich ANISOV – The scientific assistant of Russian Federal Nuclear Center E. I. Zababakhin «All-Russian Scientific Research Institute of Technical Physics» since 2018. Research interests: diffusion equations, development of algorithms for solving the diffusion equation for modern computing platforms.

DOI: 10.15514/ISPRAS-2024-36(2)-10

Designing Data Visualization System Based on Language-Oriented Approach

[1] *A. D. Dzheiranian*, ORCID: 0009-0000-8916-2855 <*addzheyranyan@edu.hse.ru*>
[2] *I. D. Ermakov*, ORCID: 0000-0003-2897-7158 <*john.ermakov27@gmail.com*>
[1] *K.A. Proskuryakov*, ORCID: 0009-0001-1678-5653 <*k.proskuryakov22@gmail.com*>
[1] *L. N. Lyadova*, ORCID: 0000-0001-5643-747X <*LNLyadova@gmail.com*>

[1] *HSE University,*
38, Studencheskaia St., Perm, 614070, Russia.
[2] *Perm State University,*
15, Bukirev St., Perm, 614990, Russia.

Abstract. The data visualization method based on a language-oriented approach is proposed. An analysis of data visualization tools and their customizability for subject areas based on user needs has been carried out. It is noted that these tools require highly qualified users to customize the data visualization format (users must have programming skills). It is proposed to customize visualization tools to the needs of users and the specifics of the user's tasks being solved by creating domain-specific languages (DSL). A system architecture based on the use of multifaceted ontology is described. The ontology includes descriptions of languages and domains, as well as rules for generating new languages and transforming constructed models. Languages are designed to describe different classes of diagrams. This system includes tools for automatic new DSL generation via mapping domain ontology onto the base language metamodel according to user-specified rules. Different types of diagrams have been classified and the main components of each type diagrams have been identified, which provides the basis for creating an ontology of data visualization languages. A base language is proposed for creating diagrams. The language customizability for specific domains is demonstrated. An example of the created data visualization models is shown.

Keywords: data visualization; domain-specific modeling; domain-specific language; metamodeling; formal grammar; multifaceted ontology; model transformation.

For citation: Dzheiranian A. D., Ermakov I. D., Proskuryakov K. A., Lyadova L. N. Designing Data Visualization System Based on Language-Oriented Approach. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 127-140. DOI: 10.15514/ISPRAS-2024-36(2)-10.

Проектирование системы визуализации данных, основанной на языково-ориентированном подходе

[1] *А. Д. Джейранян*, ORCID: 0009-0000-8916-2855 <*ADDzheyranyan@edu.hse.ru*>
[2] *И. Д. Ермаков*, ORCID: 0000-0003-2897-7158 <*John.Ermakov27@gmail.com*>
[1] *К. А. Проскуряков*, ORCID: 0009-0001-1678-5653 <*K.Proskuryakov22@gmail.com*>
[1] *Л. Н. Лядова*, ORCID: 0000-0001-5643-747X <*LNLyadova@gmail.com*>

[1] *Национальный исследовательский университет «Высшая школа экономики», Россия, 614070, г. Пермь, ул. Студенческая, д. 38.*
[2] *Пермский государственный национальный исследовательский университет, Россия, 614990, г. Пермь, ул. Букирева, д. 15.*

Аннотация. Предлагается метод визуализации данных, основанный на языково-ориентированном подходе. Проведен анализ инструментов визуализации данных и возможности их настройки на предметные области исходя из потребностей пользователей. Отмечено, что эти инструменты требуют от пользователей высокой квалификации для настройки формата визуализации данных (пользователи должны иметь навыки программирования). Предлагается настраивать средства визуализации под нужды пользователей и специфику решаемых пользователями задач путем создания предметно-ориентированных языков (DSL). Описывается архитектура системы, основанной на использовании многоаспектной онтологии. Онтология включает описания языков и предметных областей, а также правила генерации новых языков и трансформации построенных моделей. Языки предназначены для описания различных классов диаграмм. Эта система включает в себя инструменты для автоматического создания новых DSL посредством отображения онтологии предметной области на метамодель базового языка по заданным пользователем правилам. Выполнена классификация различных типов диаграмм и выявлены основные компоненты диаграмм каждого типа, что дает основу для создания онтологии языков визуализации данных. Предлагается базовый язык для создания диаграмм. Демонстрируется возможность настройки языка для конкретных предметных областей. Приведен пример созданных моделей визуализации данных.

Ключевые слова: визуализация данных; предметно-ориентированное моделирование; предметно-ориентированный язык; метамоделирование; формальная грамматика; многоаспектная онтология; трансформация модели.

Для цитирования: Джейранян А. Д., Ермаков И. Д., Проскуряков К. А., Лядова Л. Н. Проектирование системы визуализации данных, основанной на языково-ориентированном подходе. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 127–140 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–10.

1. Introduction

Visualization tools have gained widespread use in various industries, business functions, and IT disciplines, both in the private and public sectors. They are actively used in such areas as energy industry, cartography, structural health monitoring, discrete mathematics, nutrition, biology, finance, social networks, and many others. In this context, data visualization serves as a method of data analysis.

The needs of end users (data analysts) include the ‘ to create custom types of diagrams for specific tasks and domains, as basic types of diagrams with basic geometries can limit information transmission [1] and lead to ineffective visualization, which, in turn, can lead to mistakes in the decision-making [2]. Often such customization requires the use of a programming language [3]. The lack of deep programming knowledge among users leads to the need to create low-code platforms.

The available data visualization tools systems can be categorized into following groups:

1) spreadsheets (e.g., Excel, Google Sheets),
2) analytics platforms (e.g., Microsoft Power BI, Tableau),
3) diagram editors (e.g., Miro, ChartBlocks).

The standard tools of the first two groups are limited to the basic diagram types and visual effects customization options. The tools of the third group allow to create custom visualizations, edit the locations of elements, but they do not provide settings for domains.

In contrast, the use of general-purpose programming languages (e.g., Python libraries for data visualization: Matplotlib, Seaborn, Plotly, etc.) contributes to the creation of the expressive visualizations to solve specific tasks, but it requires deep programming knowledge from the diagram developer. Also, the created solution cannot be reused for other visualizations, and it is a "black box" where it is not clear how the visualization is configured [3].

Thus, the end users face two challenges:

1) How to automate visualizations development to reduce level of requirements for user programming knowledge?
2) How to ensure the customization of visualizations for specific domains?

To enhance visualization tools, a language-oriented approach is proposed. Initially, the authors proposed the concept of automating domain-specific languages (DSL) creation based on using multifaceted ontology [4]-[5]. To implement the approach, the following tasks must be solved:

1) To determine specific user requirements for customizing data visualizations for user's domains and tasks being solved. To identify the problems that users face in order to remove these limitations through the development of DSL.
2) To analyze and classify data visualization diagrams to identify the foundation for developing the data visualization languages and formalize the results for ontology development.
3) To determine the general structure of the data visualization system.
4) To develop the ontology of data visualization languages based on the completed classification of charts.
5) To describe the base language meta-model for the development of new data visualization languages.
6) To give an example of data visualization model customization.
7) To describe a code generation approach for data visualization based on created models.

2. General Requirements and Design Challenges

The specificity of user requirements lies in the need to identify new types of diagrams. This result can be achieved by interactivity, combining different types of diagrams (e.g., a combination of a histogram and a line graph) or different elements/shapes within a single diagram, etc.

Users require the creation of visualizations adapted to specialized tasks and domains too. There is also a need for modeling previously created diagrams and implementing visualization specifications into the system.

These specifications define how users can specify their requirements for creating visualizations [6]. Statistical visualizations may not be effective for comparing concepts within large data sets. Therefore, there is a demand for the rapid and straightforward creation of interactive visualizations [3]. These methods should be capable of working with various types of data and data sources [6].

Modern researchers [2], [7] highlight the limited degree of flexibility in manipulating diagram elements and the lack of focus on the real needs of the user in existing data visualization tools. Another challenge in information visualization is the loss of data transmission efficiency when an inappropriate visualization method is chosen.

3. Related Works

3.1 Classification of Visualization Methods

The diversity of visualization methods is quite large and continues to expand, which is confirmed by the constant emergence of new specialized types of diagrams (tree-like, chord, network, etc.).

Determining the most appropriate type of visualization is not an exact science and involves a multitude of approaches, but it is based on the key question: what idea you want to convey with the diagram [8], [9]. To determine which specific visualization methods are sufficient to implement most visualization ideas, it is necessary to settle on a specific taxonomy for classifying data visualization methods from the existing range.

It was decided to focus on the five-category structure proposed by Andy Kirk [9]:

1) comparing categories,
2) assessing hierarchies and part-to-whole relationships,
3) showing changes over time,
4) plotting connections and relationships,
5) mapping geospatial data.

It is followed by a review of each of these categories, highlighting specific diagrams, their unique characteristics, and elements.

The main classification criteria have been identified. These criteria are listed below:

1) The greatest popularity of visualization methods.
2) The inclusion of methods for visualizing abstract data that is characterized by multiple dimensions and the absence of explicit spatial references in addition to standard visualization methods (line graphs, histograms, pie charts, bar charts, and scatter plots).
3) The ability to present any type of visualization in an interactive form, enhancing their utility for large-scale datasets.

As a result, a classification consisting of sixteen visualization methods depending on the purpose of creating the visualization was developed (Fig. 1). This classification is the basis for the development of the data visualization languages ontology and languages metamodels.

3.2 Data Visualization Tools and Customization Options

There are few publications on creating DSLs for data visualization. Based on the review, most DSLs focus on a small set of standard charts (pie charts, histograms, etc.) or visualizations of specific data types (e.g., geospatial). They differ in levels of abstraction, contexts of use, and implementation capabilities.

The article [10] describes the process of developing a DSL for constructing and transforming data visualization techniques. The DSL is built into the Haskell programming language. The authors provide several levels of abstraction: at the lowest level, the user can create an element consisting of a specific primitive shape and a set of visual parameters. It is important to note that the basic language constructs are limited to the histogram and pie chart. However, it is allowed to arrange their elements in different ways to create more complex examples.

Article [11] presents a variational visualization model implemented through a DSL built into the PureScript programming language. This DSL allows to create variation visualizations and their combinations, such as overlaying alternative histograms. Methods for representing variation and adding variation to visualizations via DSL are discussed in the article also. This includes creating, manipulating, navigating, and rendering variational visualizations.

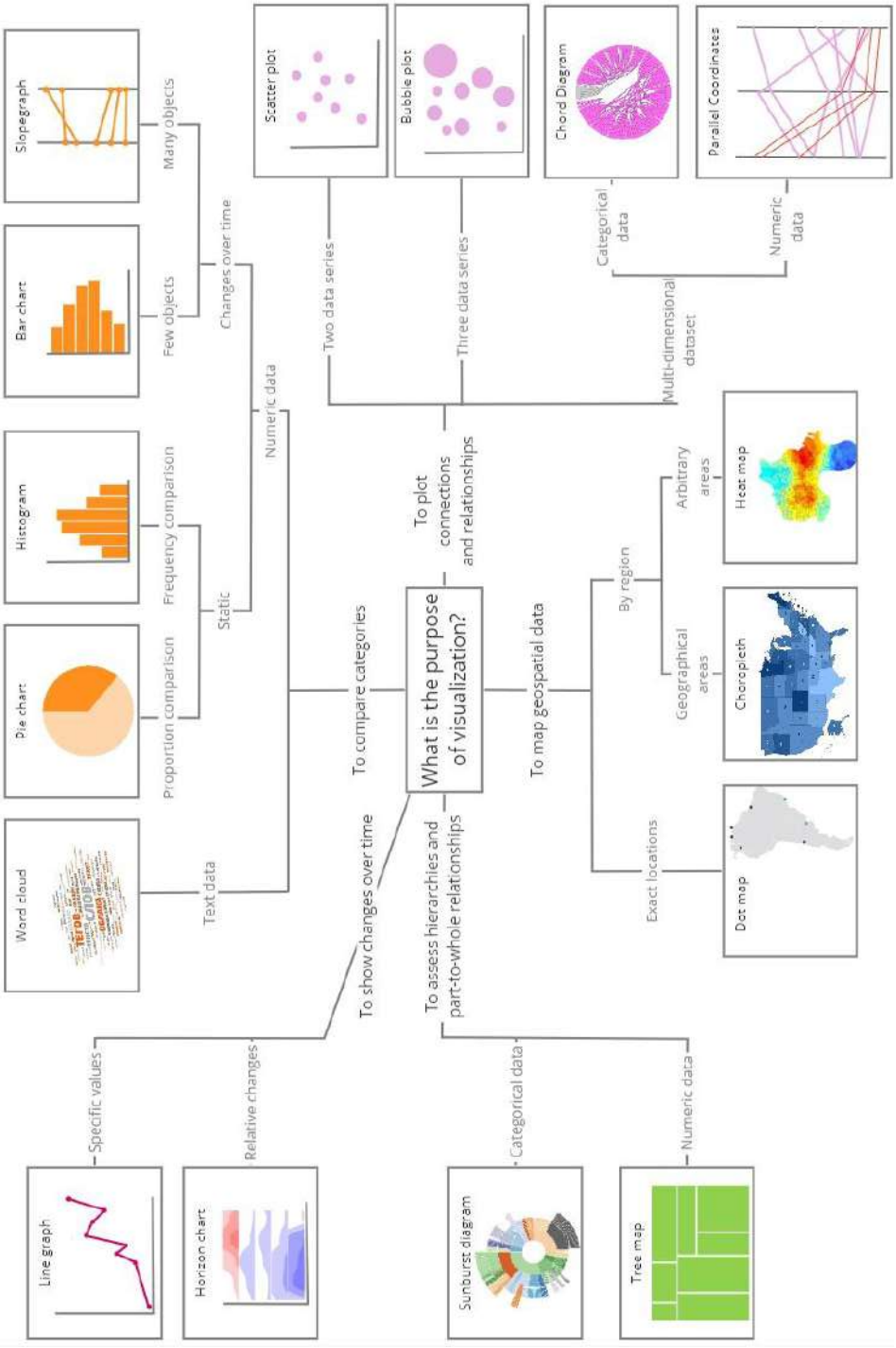


Fig. 1. Classification of visualization methods by purpose

The researchers in study [12] introduce a DSL that is focused on data geovisualizations. They utilize a compiler to facilitate the automatic generation of visualizations and the pre-processing of data. Their system leverages the power of multi-core parallelism to expedite the data pre-processing.

The considered tools provide the ability to develop new types of diagrams. But customization for specific domains was not found in them. Thus, a language-oriented approach for implementing a data visualization tool can become the main one for developing a data visualization system.

3.3 Ontology-Driven Approach to Implementing DSL Toolkits

In papers [13], [14] has been suggested to use ontologies as part of the architecture for the analytical platform. In this case, a multifaceted ontology is used, which allows to avoid data duplication, ensure changes traceability of ontologies, and automatically interpret data and the results of data analysis to provide them to different groups of users according to terminology that they are familiar with.

Use of ontologies is also considered by researchers within the domain-specific modeling (DSM) approach [15]. Domain-specific modeling is the part of model-driven engineering approach. It allows for the reduction of complexity in software system development by using DSL's. Language toolkits (DSM platforms) are used to implement this approach. They enable the generation of all essential components for working with the language (graphic editor, interpreter, etc.) according to the described metamodel.

One of the ideas for implementing the DSM approach is the usage of knowledge, specifically – the ontologies [4-5, 13]. Due to this, it is possible to automate the process of creating domain-specific languages. This approach is taken as the basis for the development of a data visualization system.

4. Generalized Structure of the Data Visualization Tools Based on the DSM Approach

The Fig. 2 shows the structure of the data visualization platform based on the DSM approach. The core of the system is a multifaceted ontology. This is an ontology that includes many other ontologies. They can be divided into three groups:

1) *Ontology of languages*. This is an ontology, in which metamodels of visual and textual languages are stored in accordance with a certain classification (by task or methodology). To describe metamodels, the HPGPR [15] metalanguage is used – an extended version of MetaCase's GOPPRR language. Models are also presented in the ontology of languages as instances of a model class.
2) *Information ontology*. This group may include several types of ontologies. First, the ontologies of data sources – they include information about data types, data storage formats, relationships, attributes, etc. Secondly, the domain ontologies. These ontologies contain a description of a specific subject area – the basic concepts (objects) and the relationships between them.
3) *Applied ontologies*. These are ontologies, that are used during metamodel generation (projection rules ontology) and model transformation (transformation rules ontology).

The platform is also partitioned into logical blocks. Let's take a closer look.

First of all, this is a block of *Functional Modules*. These are modules responsible for the basic functionality of the platform – model creation and editing, model transformations, code generation and others.

The *Language Generation Module* is used to automate the creation of meta-models using ontologies. Development of this module is based on the idea of reusing previously created languages with their reconfiguration to new subject areas through domain ontologies.

The Models Management Module is used to manage models in the platform structure: import and export models from ontologies and convert into a view applicable within the platform. Both other modules use this module to access models and languages (meta-models).

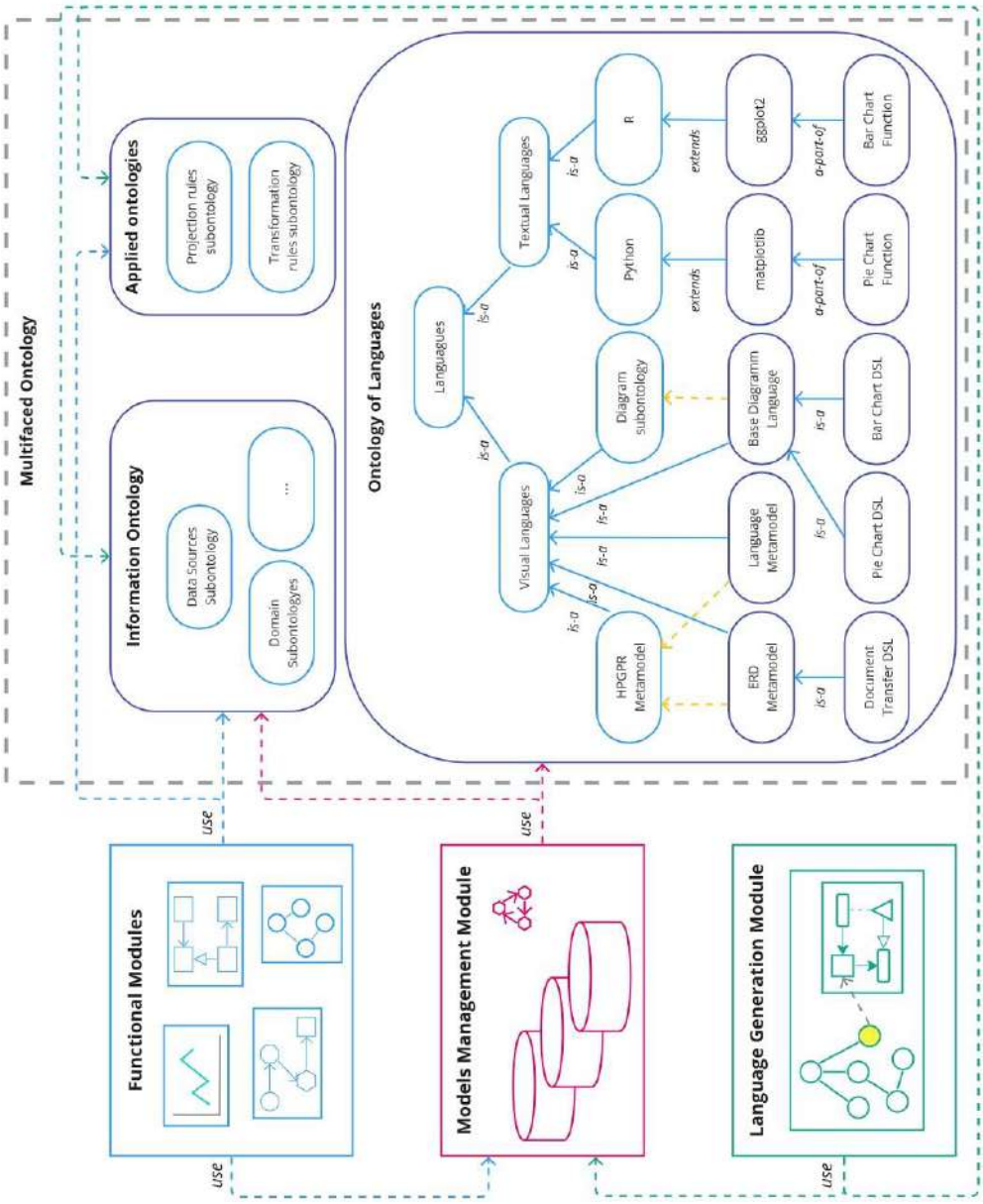


Fig. 2. Generalized structure of the data visualization tools

5. Developing Languages for Creating Data Visualization Models

To create a new language the user needs to complete several preliminary steps.

First, an ontology of data visualization should be developed. At the same time, metamodels of basic languages for creating diagrams should be developed. The basis of the ontology should consist of diagram classification and a special language that enables the creation of these languages based on it.

To customize created languages to the user's domain and tasks, it is proposed to use the mentioned approach for automizing language creation via mapping the concepts of the domain onto the metamodel of the base language [13].

5.1 Data Visualization Ontology

The process of building an ontology of data visualization languages includes the following steps:

1) Formalization of an abstract diagram, which will include properties common to all types of diagrams (title, legend, width, height, etc.).
2) Distinguishing types of diagrams into separate classes based on the developed classification of diagrams (Fig. 1).
3) Adding unique elements of charts to classes.
4) Defining relationships between the classes.

As a result, each type of diagram will have its own formalized model, for which its own DSL will be generated later.

Fig. 3 shows an abstract diagram class and its descendants in the ontology. But it is not enough to create only subclasses of diagrams. Each of the charts consists of separate elements that have their own properties. Such elements are separated into subclasses of the "*Diagram_Element*" class (Fig. 4). After defining the main classes and their unique elements, it is necessary to define the hierarchical and part-to-whole relationships between the classes. Relationships "as-is" are automatically created between the parent class and the child class. Then, to display the "part-of" relationships between a specific diagram and its elements, special connections were created. The list of relationships is shown in Fig. 5.

5.2 Base Language Metamodel and Example

The classification of diagrams and the identification of its main components allowed the development of a new DSL for creating diagrams. The meta-model of this language is shown in Fig. 6. Created language allows building data flows from sources to a diagram with the ability to filter data. User can attach various events to the diagram components, such as mouse hover or mouse click.

Created metamodel describes an abstract language. This language is the basis for the development of new languages for visualizing data in a specific domain area. It needs to be modified for a specific type of diagrams in order to use. Designed new language meta-model can be extended and customized by user with mapping domain model onto components of diagrams in basic language meta-model.

5.3 Language Customization Example

For example, let's take the customer evaluation of service. As a basic language, we take a bar chart. Now we need to create an ontology in which we describe emoji's – add vertices for each emotion and set an image for them. During the language generation, we will need to specify the correspondence between the concepts of the domain ontology and the concepts of the diagram language. As a result, we get a language that allows to create diagrams, as in Fig. 7.

Such language uses domain terms, and it is easy to use for end users.

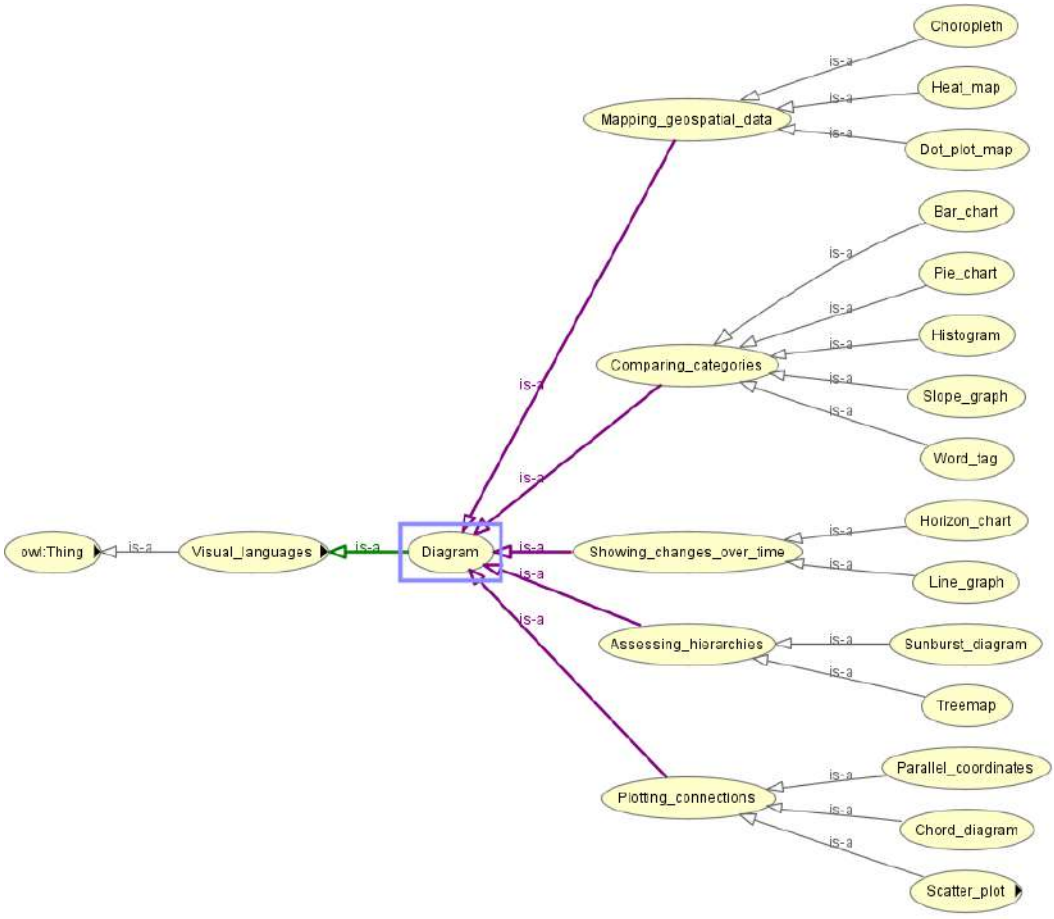


Fig. 3. Fragments of the multifaceted ontology: hierarchy of diagram classes in the ontology

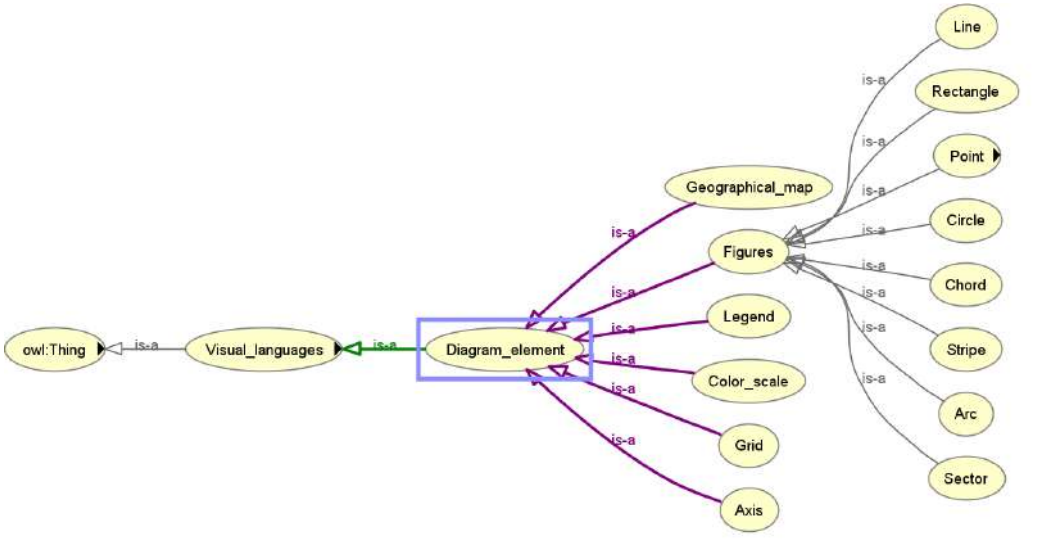


Fig. 4. Fragments of the multifaceted ontology: class hierarchy of diagram elements in the ontology

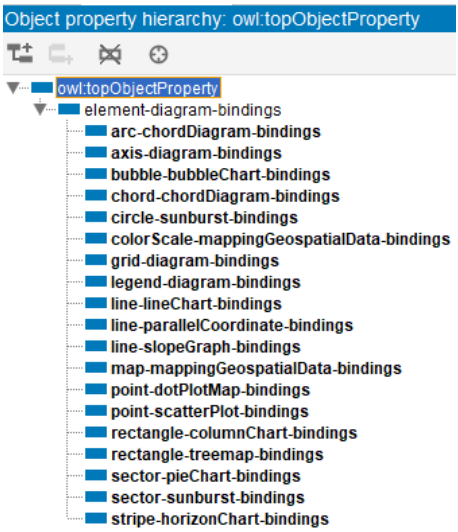


Fig. 5. Fragments of the multifaceted ontology: relation types

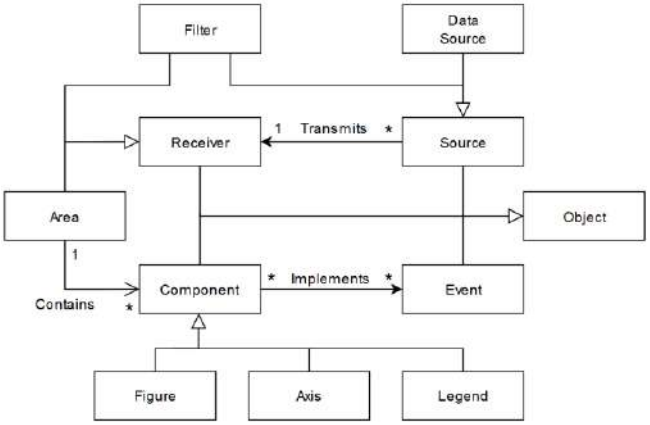


Fig. 6. Metamodel of the base language for diagram creation

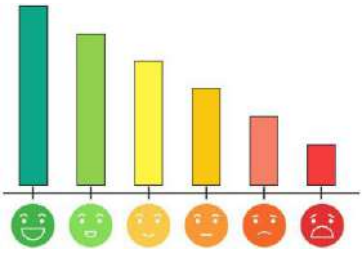


Fig. 7. Diagram customization example

6. Generating Code for Data Visualization Based on Created Models

Using DSL, visualization models are built, but code generation (also called "*Model-To-Text*" transformation) is required for data visualization. The vast majority of modern DSM platforms use a template-based approach to code generation, which allows efficient templates reuse [16]. Templates are described not for a specific model, but for a meta-model [17], or in this case a visualization language. Each template consists of two parts – static and dynamic. The static part is the same for all models, and the dynamic part uses information "extracted" from a particular model.

In this paper, we propose to use the approach described in [15], which consists in creating and using transformation rules in the visual environment in the form of a constructor. Each rule consists of the left part – visual language metamodel objects and the corresponding right part – textual language constructs, which are templates. All language constructs, as well as created transformation rules, are stored in a multifaceted ontology.

Python and R are the preferred programming languages for data visualization purposes. Python is widely popular in this area due to its wide range of suitable libraries, including Matplotlib, Seaborn and Plotly, and its simple syntax. Although R is second only to Python in industry, it also has a rich arsenal of visualization tools and continues to be a popular choice in academia. Thus, as textual language constructs, it makes sense to use code fragments in Python and R containing calls to library functions for data visualization.

Once transformation rules are created, they can be applied to specific models to generate data visualization code. The code generation algorithm is based on traversing the internal representation of models, which can be modeled with a graph.

7. Conclusion

In this paper a structure of a knowledge-driven data visualization tool based on a language-oriented approach is proposed. This approach allows overcoming several limitations of existing visualization tools and providing users an ability to customize the data visualization models for different domain areas via creating special languages and reduces the requirements for end user's programming skills when creating DSL and charts.

The practical applicability of this approach is demonstrated through the example of creating a chart for assessing customer service quality. New base domain-specific language metamodel for data visualization was created with using the proposed approach. Then a diagram model was built with created DSL and data visualization was built according to the specified requirements.

The next stage of the research is implementing of the described ideas and expanding possibilities of interactive visualization via interpretation of the created models. The theoretical basis to developing this approach is Vega-Lite [18], a high-level grammar that enables rapid specification of interactive data visualizations.

References

[1]. Midway S. R. Principles of Effective Data Visualization. Patterns, 2020, vol. 1, issue 9, article 100141. DOI: 10.1016/j.patter.2020.100141.

[2]. Oral E., Chawla R., Wijkstra M., Mahyar N., Dimara E. From Information to Choice: A Critical Inquiry Into Visualization Tools for Decision Making. IEEE Transactions on Visualization and Computer Graphics, 2024, vol. 30, no. 1, pp. 359–369. DOI: 10.1109/TVCG.2023.3326593.

[3]. Morgan R., Grossmann G., Schrefl M., Stumptner M, Payne T. VizDSL: A Visual DSL for Interactive Information Visualization. In Proc. of the 30th International Conference "Advanced Information Systems Engineering", CAiSE 2018, 2018, pp. 440–455. DOI: 10.1007/978-3-319-91563-0_27.

[4]. Lyadova L., Sukhov A., Nureev M. An Ontology-Based Approach to the Domain Specific Languages Design. In Proc. of the 15th IEEE International Conference on Application of Information and Communication Technologies (AICT2021), 2021, 6 p. DOI: 10.1109/AICT52784.2021.9620493.

[5]. Kulagin G., Ermakov I., Lyadova L Ontology-Based Development of Domain-Specific Languages via Customizing Base Language. In Proc. of the 16th IEEE International Conference on Application of Information and Communication Technologies (AICT2022), 2022, 6 p. DOI: 10.1109/AICT55583.2022.10013619.

[6]. Qin X., Luo Y., Tang, N., Li G. Making Data Visualization More Efficient and Effective: a Survey. The VLDB Journal, 2019, vol. 29, no. 1, pp. 93–117. DOI: 10.1007/s00778-019-00588-3.

[7]. Cepero García M. T., Montané-Jiménez L. G. Visualization to Support Decision-Making in Cities: Advances, Technology, Challenges, and Opportunities. In Proc of the 8th International Conference in Software Engineering Research and Innovation (CONISOFT), 2020, pp. 198–207. DOI: 10.1109/CONISOFT50191.2020.00037.

[8]. Zelazny G. The Say It with Charts Complete Toolkit. New York, McGraw-Hill Professional, 2006, 312 p.

[9]. Kirk A., Data Visualization: A Successful Design Process. Birmingham, Packt Publishing Ltd, 2012, 189 p.

[10]. Smeltzer K., Erwig M., Metoyer R. A transformational Approach to Data Visualization. In the Proc. of the 2014 International Conference on Generative Programming: Concepts and Experiences (GPCE 2014), 2014, pp. 53–62. DOI: 10.1145/2658761.2658769.

[11]. Smeltzer K., Erwig M. A Domain-Specific Language for Exploratory Data Visualization. In Proc. of the 17th ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences (GPCE 2018), 2018, pp. 1–13. DOI: 10.1145/3278122.3278138.

[12]. Ledur C., Griebler D., Manssour I., Fernandes L. G. A High-Level DSL for Geospatial Visualizations with Multi-core Parallelism Support. In Proc. of the IEEE 41st Annual Computer Software and Applications Conference (COMPSAC), 2017, pp. 298–304. DOI: 10.1109/COMPSAC.2017.18.

[13]. Zayakin V., Lyadova L., Rabchevskiy E. Design Patterns for a Knowledge-Driven Analytical Platform. Proceedings of the Institute for System Programming of the RAS (Proceedings of ISP RAS), 2022, vol. 34, no. 2, pp. 43–56. DOI: 10.15514/ISPRAS-2022-34(2)-4.

[14]. Zayakin V. S., Lyadova L. N., Lanin V. V., Zamyatina E. B., Rabchevskiy E. A. An Ontology-Driven Approach to the Analytical Platform Development for Data-Intensive Domains. Knowledge Discovery, Knowledge Engineering and Knowledge Management. IC3K 2021. Communications in Computer and Information Science. Springer, Cham. 2023, vol. 1718, pp. 129–149. DOI: 10.1007/978-3-031-35924-8_8.

[15]. Lyadova L., Ermakov I., Lanin V., Proskuryakov K. Approach to the Development of Ontology-Driven Language Toolkits Based on Metamodeling. In Proc. of the IEEE 17th International Conference on Application of Information and Communication Technologies (AICT2023), 2023, 6 p. DOI: 10.1109/AICT59525.2023.10313152.

[16]. Kahani N., Bagherzadeh M., Cordy J. Survey and Classification of Model Transformation Tools. Software & Systems Modeling, 2019, vol. 18, pp. 2361–2397. DOI: 10.1007/s10270-018-0665-6.

[17]. Ding J., Lu J., Wang G., Ma J., Kiritsis D., Yan Y. Code Generation Approach Supporting Complex System Modeling Based on Graph Pattern Matching. IFAC-PapersOnLine, 2022, vol. 55, Issue 10, pp. 3004–3009. DOI: 10.1016/j.ifacol.2022.10.189.

[18]. Satyanarayan A., Moritz D., Wongsuphasawat K, Heer J. Vega-Lite: A Grammar of Interactive Graphics. IEEE Transactions on Visualization and Computer Graphics, 2016, vol. 23, no. 1, pp. 341–350. DOI: 10.1109/TVCG.2016.2599030.

Информация об авторах / Information about authors

Анна Даниеловна ДЖЕЙРАНЯН – студент бакалавриата Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь), образовательная программа «Программная инженерия». Сфера научных интересов: анализ и визуализация данных, генеративные модели, предметно-ориентированное моделирование.

Anna Danielovna DZHEIRANIAN – undergraduate student at the National Research University – Higher School of Economics (HSE University, Perm Branch), educational program “Software Engineering”. Research interests: data analysis and visualization, generative models, domain-specific modeling.

Иван Денисович ЕРМАКОВ – студент магистратуры Пермского государственного национального исследовательского университета (ПГНИУ), образовательная программа «Прикладная математика и информатика». Сфера научных интересов: предметно-ориентированное моделирование, языковые инструментарии, управляемые знаниями системы.

Ivan Denisovich ERMAKOV – master student at the Perm State National Research University (PSU), educational program “Applied Mathematics and Computer Science”. Research interests: domain-specific modeling, language toolkits, knowledge-driven systems.

Кирилл Александрович ПРОСКУРЯКОВ – студент магистратуры Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь), образовательная программа «Бизнес-информатика». Сфера научных интересов: предметно-ориентированное моделирование, языковые инструментарии, управляемые знаниями системы.

Kirill Alexandrovich PROSKURYAKOV – master student at the National Research University – Higher School of Economics (HSE University, Perm Branch), educational program “Business Informatics”. Research interests: domain-specific modeling, language toolkits, knowledge-driven systems.

Людмила Николаевна ЛЯДОВА – кандидат физико-математических наук, доцент, доцент кафедры информационных технологий в бизнесе Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ–Пермь). Сфера научных интересов: языки моделирования, предметно-ориентированное моделирование, языковые инструментарии, CASE-средства, системы имитационного моделирования.

Lyudmila Nikolaevna LYADOVA – Cand. Sci. (Phys.-Math.) in Computer Science, Associate Professor of the Department of Information Technologies in Business of the National Research University – Higher School of Economics (HSE University, Perm Branch). Research interests: modeling languages, domain specific modeling, language toolkits, CASE tools, simulation systems.

DOI: 10.15514/ISPRAS–2024–36(2)–11

Соревнования по формальной верификации VeHa-2023: опыт проведения

[1] *С.М. Старолетов,* ORCID: 0000-0001-5183-9736 <*serg_soft@mail.ru*>
[2,3] *Д.А. Кондратьев*, ORCID: 0000-0002-9387-6735 <*apple-66@mail.ru*>
[2] *Н.О. Гаранина*, ORCID: 0000-0001-9734-3808 <*garanina@iis.nsk.su*>
[4] *И.В. Шошмина,* ORCID: 0000-0001-5120-0385 <*shoshmina_iv@spbstu.ru*>

[1] *АлтГТУ им. И.И. Ползунова*,
Россия, 656038, Алтайский край, г. Барнаул, пр. Ленина, 46.
[2] *Институт систем информатики им. А.П. Ершова СО РАН*,
Россия, 630090, г. Новосибирск, пр. Академика Лаврентьева, 6.
[3] *Новосибирский государственный университет*,
Россия, 630090, г. Новосибирск, ул. Пирогова, 1.
[4] *Санкт-Петербургский политехнический университет Петра Великого*,
Россия, 195251, г. Санкт-Петербург, ул. Политехническая, 29.

Аннотация. Для создания современного конкурентоспособного и доверенного программного обеспечения необходимо использовать знания формальных методов. В настоящее время огромное количество студентов обучается специальностям, связанным с программированием. Однако при обучении в вузе сложно получить навык практического применения теоретических знаний. Короткие соревнования с нестандартными близкими к промышленным задачам могут пробудить интерес студентов к области формальных методов. В нашей статье описан первый опыт организации соревнования по формальной верификации программ среди студентов российских вузов. Соревнования проводились в связке с семинаром по семантике, спецификации и верификации программ (PSSV) в Иннополисе в ноябре 2023 года. Формат соревнования был близок к формату так называемых хакатонов. Участникам было предложено решить задачи по верификации с использованием заранее определенных инструментов проверки моделей и дедуктивной верификации. Мы рассмотрим вопросы организации такого мероприятия, предложенные задачи, результаты решений и обратную связь от участников.

Ключевые слова: формальные методы; соревнования по формальной верификации; проверка моделей; дедуктивная верификация; хакатон.

Для цитирования: Старолетов С.М., Кондратьев Д.А., Гаранина Н.О., Шошмина И.В. Соревнования по формальной верификации VeHa-2023: опыт проведения. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 141-168. DOI: 10.15514/ISPRAS–2024–36(2)–11.

VeHa-2023 Formal Verification Contest: The Experience

[1] *S.M. Staroletov,* ORCID: 0000-0001-5183-9736 <*serg_soft@mail.ru*>
[2,3] *D.A. Kondratyev,* ORCID: 0000-0002-9387-6735 <*apple-66@mail.ru*>
[2] *N.O. Garanina,* ORCID: 0000-0001-9734-3808 <*garanina@iis.nsk.su*>
[4] *I.V. Shoshmina,* ORCID: 0000-0001-5120-0385 <*shoshmina_iv@spbstu.ru*>

[1] *Polzunov Altai State Technical University,*
46, Lenin Ave., Barnaul, 656038, Russia.
[2] *Ershov Institute of Informatics Systems, Siberian Branch of the Russian Academy of Sciences,*
6, Acad. Lavrentjev pr., Novosibirsk 630090, Russia.
[3] *Novosibirsk State University,*
1, Pirogova street, Novosibirsk 630090, Russia.
[4] *Peter the Great St. Petersburg Polytechnic University,*
29, Politekhnicheskaya street, St. Petersburg, 195251, Russia.

Abstract. To create modern competitive and trusted software, it is necessary to use knowledge of formal methods. Currently, a huge number of students are studying specialties related to programming. However, when studying at a university, it is difficult to gain the skill of practical application of theoretical knowledge. Short competitions with non-standard, industrial-related problems can arouse students' interest in the field of formal methods. The article describes the first experience of organizing a competition in formal verification of programs among students of Russian universities. The competition was held in conjunction with a seminar on program semantics, specification and verification (PSSV) in Innopolis in November 2023. The format of the competition was close to the format of so-called hackathons. Participants were asked to solve verification problems using predefined model checking and deductive verification tools. We consider the issues of organizing such an event, proposed tasks, results of decisions and feedback from participants.

Keywords: formal methods; formal verification competitions; model checking; deductive verification; hackathon.

For citation: Staroletov S.M., Kondratyev D.A., Garanina N.O., Shoshmina I.V. VeHa-2023 Formal Verification Contest: The Experience. Trudy ISP RAN/Proc. ISP RAS, vol 36, issue. 2, 2024., pp. 141-168 (in Russian). DOI: 10.15514/ISPRAS-2024-36(2)-11.

1. Введение и мотивация проведения соревнования

Многие обучающиеся заинтересованы в собственном развитии и готовы изучать перспективные технологии. Однако большинство из будущих программистов при саморазвитии следуют текущим трендам, например, таким как нейронные сети или блокчейны. В подобных направлениях в интернете можно найти большое количество легко доступной информации, включая вводные лекции (тьюториалы) и описания разработки проектов с нуля. В результате студенты, пытаясь развиваться самостоятельно, увеличивают и без того огромный вклад в наиболее популярные направления.

При этом фундаментальные дисциплины не очень популярны, хотя влияние этих дисциплин на развитие области программирования велико и постоянно.

Непопулярность фундаментальных дисциплин легко объясняется: у них высокий порог входа для получения результатов, а также существуют сложности с выбором первых проектов, которые молодой специалист преодолеть не в состоянии.

Можно констатировать, что проектная деятельность в этой области в вузах практически отсутствует и, как правило, работа заканчивается либо на уровне индивидуальных проектов, либо она ограничивается выполнением упражнений в рамках учебных курсов.

С другой стороны, можно отметить рост интереса у студентов как к математике в целом, так и изучению вопросов, связанных с семантикой языков программирования и надежности программного обеспечения.

Задумываясь о проведении очередного семинара по спецификации и верификации программ PSSV [1], в оргкомитете возникло предложение компенсировать недостаточное практическое использование молодыми специалистами фундаментальных методов и провести соревнования по формальным методам верификации программ. Наш оптимизм основывался на личном опыте организаторов при решении практических задач, а также на опыте проведения подобных соревнований в смежных областях. Было предложено использовать для соревнования название VeHa (Verification Hackathon). Организаторами VeHa-2023 стали: Наталья Олеговна Гаранина (ИСИ СО РАН, НГУ), Дмитрий Александрович Кондратьев (ИСИ СО РАН, НГУ), Сергей Михайлович Старолетов (АлтГТУ, ИАиЭ СО РАН), Ирина Владимировна Шошмина (СПбПУ).

В последнее время в сфере программной инженерии среди разработчиков стали популярны так называемые *хакатоны*. Это слово комбинирует “to hack” и “марафон”, причём “to hack” здесь используется в значении “применять нестандартные методы решения сложных задач”, а не в значении “взламывать”. Традиционно, для концентрации на проектной деятельности, для быстрого решения поставленных задач хакатона собираются команды участников, организаторы проводят открытые лекции по используемым технологиям, а далее команды активно работают над проектами. При этом подразумевается, что проекты могут быть реализованы за короткое время и, возможно, продолжены в дальнейшем. Организаторы также фиксируют область соревнования, а если это соревнование решает индустриальную задачу, то задается стек технологий и предлагается вариант технического задания. В конце проводится финальная демонстрация решений и выборы победителей. Хакатоны отличает высокая мотивация и концентрация участников на проектах.

Опыт участия и проведения подобных мероприятий организаторами VeHa-2023 показывает, что знания и контакты, полученные во время такого рода мероприятий, способствуют профессиональному росту и формированию профессиональных социальных связей.

Несмотря на то, что только небольшое количество проектов хакатонов продолжается в дальнейшем, сформировавшиеся связи могут привести к долгосрочным глубоким исследованиям, постановка задачи которых началась в кулуарах соревнований [2].

Сейчас хакатоны проводятся как в рамках городских ИТ-сообществ (например, городской Барнаульский хакатон [3]), так и по решению задач для компаний (например, в нефтегазовом секторе [4]), а также для государственных компаний [5].

В современных университетах для развития у студентов навыков взаимодействия и работы над быстрыми проектами также организовываются хакатоны, например, [6].

Учитывая упомянутые тенденции, организаторы VeHa-2023 решили провести в рамках семинара своего рода хакатон по верификации программ.

Однако, чтобы расширить географию участников, было принято решение изменить пространственный формат соревнования на гибридный – онлайн (удаленно) и офлайн (очно) в Иннополисе. Такое решение не совсем соответствует концепции хакатона, поэтому мероприятие назвали “соревнованием”.

В качестве формальных методов верификации программ рассматривались метод проверки моделей и дедуктивная верификация, а также задачи верификации с использованием лямбда-исчисления и теории типов. Следует отметить, что последние два направления не вошли в описываемое соревнование и оставлены на будущее.

Методы формальной верификации программ осуществляют проверку корректности программы относительно ее формальных спецификаций. В зависимости от формального языка спецификаций отличаются возможности описываемых требований к программам, а также сами методы проверки.

В методах дедуктивной верификации программ спецификации задают ограничения на данные программы в её ключевых точках [7-8]: предусловия описывают свойства входных данных программы; постусловия – свойства выходных данных программы; инварианты

циклов [9] фиксируют свойства в ключевых точках. Эти спецификации, выраженные формулами, называются условиями корректности программы [10]. Таким образом, дедуктивная верификация сводит задачу проверки корректности программы относительно ее спецификаций к задаче проверки истинности формул.

Метод проверки модели позволяет автоматически проверить, выполняется ли заданная логическая формула на данной структуре [11-12]. Структура строится по разрабатываемой программной системе как система переходов с конечным числом состояний, модальная (темпоральная) логическая формула описывает требования к поведению системы. Чтобы проверить, верна ли спецификация на системе переходов, применяется эффективная процедура поиска, которая является полностью автоматической.

Существует большое количество инструментов верификации, реализующих дедуктивную верификацию и метод проверки модели. Для проведения соревнования мы остановили свой выбор на инструментах, которые было бы легко и быстро освоить.

Применение дедуктивной верификации на практике осложняется рядом проблем, требующих привлечения пользователя системы верификации [13]. Такими проблемами, в частности, являются задание инвариантов циклов [14-15] и доказательство условий корректности [16]. Поэтому мы предложили участникам соревнования программную систему дедуктивной верификации, которая может упростить и решение проблемы инвариантов циклов, и решение проблемы доказательства условий корректности программ для выбранных задач верификации, а именно – систему C-lightVer для программ на языке C-light [17-21], представительном подмножестве языка C. В отличие от других известных систем дедуктивной верификации [22-31], система C-lightVer даёт возможность автоматизировать задание инвариантов циклов для программ, циклы которых являются финитными итерациями [32]. Класс финитных итераций над последовательностями данных покрывает такие распространенные виды циклов, как циклы над массивами, циклы над списками, циклы над деревьями и т.д. [33]. Также система C-lightVer позволяет во многих случаях автоматизировать доказательство условий корректности с помощью применения системы доказательств ACL2 [34] и специальных стратегий доказательства [19, 32]. Для задания спецификаций и теории предметной области в системе C-lightVer используется язык Applicative Common Lisp [34], входной язык системы доказательства ACL2.

При использовании инструментов метода проверки модели [35-39] существуют несколько препятствий. Во-первых, существуют сложности с освоением языка моделирования, описывающего структуру переходов в системах. Во-вторых, пользователь должен знать язык темпоральной логики, с помощью которого описываются свойства систем. В-третьих, если верификатор генерирует опровержение свойства (контрпример), необходимо суметь интерпретировать этот вывод.

Нами было выбрано средство верификации SPIN с входным языком Promela [40]. Язык Promela является C-подобным языком с включением конструкций, облегчающих описание взаимодействия между независимыми процессами моделируемой системы. При этом формальную модель по введенному пользователем коду SPIN строит автоматически.

Свойства систем в SPIN задаются с помощью формул линейной темпоральной логики (Linear Temporal Logic – LTL). Эта логика является расширением логики высказываний с добавлением нескольких темпоральных модальностей. Кроме того, свойства, заданные с помощью формул линейной темпоральной логики, легко интерпретируются человеком, так как интуитивно ложатся на представление о линейном течении времени.

Контрпримеры в SPIN выдаются как последовательности шагов с указанием конкретных операторов программы, что позволяет легко анализировать ошибочную трассу.

Для SPIN и Promela существует плагин для популярной среди разработчиков среды Visual Studio Code (около 15000 инсталляций) [41], а подсветка синтаксиса также поддерживается

на GitHub и в Latex, что показывает, что язык Promela является зрелым и используемым достаточно большим количеством разработчиков и исследователей.

Кроме приведенных выше аргументов, для выбранных инструментов верификации авторами уже был разработаны презентации и методические материалы на русском языке.

Для подбора задач мы проводили обсуждение идей в доступных нам сообществах с предложениями задач, которые можно формализовать за два дня. Эти задачи могли поступать как от научного сообщества, так и от компаний, работающих с критическими и доверенными системами.

К сожалению, в этом году наши индустриальные партнеры не смогли предложить интересные задачи для соревнования и выделить менторов для работы с участниками, хотя впоследствии одна из компаний (“БАРС Групп”) обеспечила призы участникам. Таким образом, все задачи соревнования были предложены организаторами – авторами статьи.

Для оценки разных методов верификации мы сформировали подкомиссии из членов жюри.

Критерии для оценки количества баллов, которое получало решение по дедуктивной верификации, включали корректность постусловия и задания теории предметной области, успешность верификации полученной программы в системе.

При оценке решений по методу проверки модели использовались критерии, включающие качество моделирования и формулировки требований к модели, успешность верификации программы в системе.

В зависимости от выполнения данных критериев решение по каждому методу верификации получало оценку от 0 до 10 баллов.

Дальнейшее содержание статьи представляет собой обзор практики соревнований по формальным методам (раздел 2), структуры соревнования и статистические данные участников (раздел 3), подробные описания предложенных задач (раздел 4), способы подачи обучающего материала соревнований (раздел 5), отчет о проведении соревнования (раздел 6) и, наконец, анализ информации, полученной от участников, их решения и основные ошибки, а также результаты обратной связи (раздел 7).

2. Обзор соревнований по формальной верификации

Предшественниками первых соревнований по дедуктивной верификации программ можно назвать наборы задач, составленные в начале XXI века для проверки возможностей известных на тот момент подходов к автоматизации дедуктивной верификации [42-45]. Эти наборы оказали влияние на составление задач на первых соревнованиях в этой области [46].

В 2009–10 годах начался проект Verified Software Initiative по масштабному применению формальной верификации в индустриальном программировании [47], в рамках которого прошло первое соревнование по дедуктивной верификации программ [48].

Это соревнование было организовано совместно с конференцией по формальной верификации VSTTE в 2010 году [49] и проходило среди команд-участников конференции. Оно стало первым мероприятием серии VSComp. Второе и последнее соревнование этой серии, аффилированное с конференцией VSTTE 2011, прошло онлайн [50].

В 2011 году в рамках проекта COST IC0701 состоялось соревнование по дедуктивной верификации в связке с конференцией FoVeOOS 2011 [51], которое также проводилось среди команд. Оно положило начало самой успешной на данный момент серии соревнований по дедуктивной верификации VerifyThis. Опыт проведения этих соревнований описан в статье [52]. Авторы статьи [52] отмечают, что наиболее сложной проблемой является составление задач для соревнования. На первых соревнованиях по дедуктивной верификации организаторы сами формулировали задачи, но позднее была предложена идея перейти к сбору задач в области верификации от всех желающих, что описано в совместной статье организаторов серий VSComp и VerifyThis [53].

Серия соревнований VerifyThis проходит ежегодно с 2011 года (кроме 2020 года) в связке с различными конференциями по применению формальных методов: например, в 2012 году – с конференцией Formal Methods 2012 [46]. Начиная с 2015 года, серия соревнований VerifyThis аффилируется с серией конференций ETAPS [54-55]. Команды-участники соревнований VerifyThis должны присутствовать на месте проведения соревнований, онлайн участие не предусмотрено [56]. Начиная с 2012 года, организаторы соревнований заранее объявляют сбор задач от всех желающих помочь проведению соревнований. Задачи описываются на естественном языке с возможными включениями псевдокода. Команды-участники должны задать формальную спецификацию задачи, реализовать программу, соответствующую данным спецификациям, и доказать, что эта программа корректна относительно своих спецификаций. Жюри оценивает предоставленные решения на предмет корректности, полноты и элегантности.

Отметим, что время соревнований VerifyThis ограничено кратким временем проведения связанной конференции, поэтому для решения трудоёмких задач из области индустриального программирования с недавнего времени организаторы стали объявлять долговременные соревнования под эгидой VerifyThis [57-58]. Для таких задач отводится длительный промежуток времени между конференциями, связанными с VerifyThis. Подобные долговременные соревнования демонстрируют важную идею соревнований по дедуктивной верификации, которая состоит в том, что оценивается не скорость решения задач, а способность решить сложные задачи верификации программ.

В 2019 году соревнование VerifyThis стало частью мероприятия TOOLympics 2019 [59-60], в состав которого вошли несколько соревнований, связанных с применением формальных методов в программировании. Среди соревнований TOOLympics 2019 особо отметим соревнование RERS 2019 [61], которое входит в серию соревнований по методу проверки моделей RERS [62-65]. В соревновании RERS, в отличие от описываемого контеста VeHa-2023, участникам не было необходимости формализовать записанные на естественном языке требования, так как в задачах они заданы на языке логики линейного времени LTL. Поэтому RERS является скорее соревнованием программных инструментов проверки моделей, чем компетенций участников. Серия RERS включает специальное направление по решению задач для моделей параллельных программ, заданных на языке Promela [62], который также использовался в рамках нашего VeHa-2023. В 2019 году впервые в истории проведения RERS индустриальным партнером соревнования были предложены задачи верификации из индустриальных информационных технологий [61]. Итак, как в мероприятии TOOLympics 2019, так и в контесте VeHa-2023 проводились соревнования и по проверке моделей, и по дедуктивной верификации, однако участники VeHa-2023 были обязаны представить решения по обоим направлениям. Также отметим, что в рамках TOOLympics 2019 было проведено соревнование Termination Competition 2019 [66] (входящее в серию соревнований Termination Competition (termCOMP) [66-69]) по проверке выполнения свойства завершимости исполнения программ и систем переписывания термов.

В 2022 году прошло мероприятие SpecifyThis [70]. Названное по аналогии с VerifyThis, событие SpecifyThis не похоже на соревнование. Скорее это рабочий семинар, на котором участники делились, каким образом им приходилось решать проблему задания формальных спецификаций. Однако перспектива соревнований по заданию формальных спецификаций программ выглядит многообещающей в свете нетривиальности формализации свойств корректности индустриальных программ.

Из обзора родственных мероприятий можно сделать следующие выводы:

1. Соревнования по формальной верификации программ проводятся в связке с конференциями по применению формальных методов в программировании. Это позволяет привлекать участников конференции как в качестве организаторов соревнования, так и в качестве соревнующихся.

2. Современным трендом стало проведение таких контестов в рамках конференций по формальным методам в программировании, которые включают в себя несколько соревнований по разным направлениям применения формальных методов в программировании.
3. Все чаще к проведению соревнований по формальным методам привлекаются индустриальные партнеры – компании, работающие в области индустриального программирования. Они заинтересованы в применении формальной верификации в своей отрасли и могут предложить соответствующие задачи.
4. Новым направлением в развитии соревнований по формальной верификации стало проведение долговременных конкурсов между ежегодными проведениями аффилированных конференций. На таких соревнованиях во время проведения конференции объявляются крупные задачи, а решения проверяются через год. Такая схема позволяет участникам решать задачи формальной верификации программ из области индустриального программирования.
5. Традиционно на соревнованиях по формальной верификации за несколько месяцев до самого соревнования объявляют сбор задач. Каждый желающий может прислать организаторам свою задачу, которая может представлять интерес в области формальной верификации программ. Организаторы осуществляют отбор лучших задач.

3. Структура соревнования и участники

Согласно предварительному плану соревнования мы организовали конкурс на первоначальное обсуждение идей с предложениями задач. Как уже упоминалось во введении, промышленные партнеры не смогли предоставить задачи вовремя, поэтому авторы статьи предлагали задачи сами.

При составлении задач, кроме временного ограничения, нам было важно, чтобы задача затрагивала участников на эмоциональном уровне, чтобы задача касалась промышленных объектов.

В качестве задачи для метода проверки модели Н. О. Гараниной была предложена задача о миссии “Луны-25”, падение которой широко обсуждалось в прессе. Ошибка, которую предлагалось промоделировать участникам, касалась нарушения взаимодействия модулей. Окончательная формулировка этой задачи была составлена Н. О. Гараниной и И. В. Шошминой.

Предложение с задачей о миссии “Луны-25” было поддержано Д. А. Кондратьевым и в дедуктивной верификации. Для дедуктивной верификации базовая формулировка была близка к используемой в публичных изданиях, из которой выкристаллизовалась относительно простая задача.

Таким образом, с помощью задачи о миссии “Луны-25” мы пытались сформировать разные взгляды на ошибки, возникающие в программах. Это, с нашей точки зрения, соответствует задаче верификации сложных промышленных программ, где ошибка является следствием множества причин и требует комплексного исследования. Задачи, посвященные миссии “Луны-25”, оценивались нами как относительно простые. С. М. Старолетов предложил задачу верификации протокола телеграмм для европоездов и, совместно с Н. О. Гараниной – задачу верификации архитектурной последовательности работы (pipeline) GPGPU. Эти задачи были выделены авторами из их научно-исследовательских проектов. Сложность этих задач была выше.

Для привлечения большего числа участников мы выбрали гибридный формат проведения соревнования: допускалось и очное, и удаленное участие. От жюри очно на PSSV-2023 работал С. М. Старолетов, остальные члены жюри работали удаленно.

Для поддержки гибридного формата мы разработали сайт [71]. На сайте были размещены материалы по методу проверки моделей и дедуктивной верификации, выложены краткие описания задач соревнования и даны ссылки на Telegram-канал для обсуждений, Skype-митинг для тьюториалов и “живых” консультаций, а также GitHub-репозиторий для последующей загрузки решений.

В процессе создания сайта у нас была возможность быстрого совместного редактирования содержимого на основе подхода WYSIWYG, что позволило сосредоточиться в большей степени на задачах соревнования, чем на оформлении материалов. Для визуального наполнения сайта использовались популярные изображений из сети по теме багов и верификации, в частности изображение “первый баг” [72]. По отзывам участников, сайт соревнования получился удобный и красивый.

Основное взаимодействие с участниками осуществлялось через мессенджер Telegram. Выбор перечисленных технических средств обусловлен прежде всего быстротой разработки, поддержки браузеров как десктоп-систем, так и мобильных устройств.

Нами было установлено следующее расписание соревнований: 2 ноября 2023 – проведение тьюториалов по теоретическим и практическим аспектам формальной верификации онлайн в Skype и на месте проведения соревнований, утром 3 ноября – публикация полных текстов заданий на сайте, утром 5 ноября – завершение загрузка решений. Во время соревнований проводились онлайн- и очные консультации. К установленному сроку участники должны были загрузить решения с помощью pull-request [73] в заданный общий репозиторий на GitHub [74]. Этот репозиторий должен был быть предварительно клонирован участниками. Участники должны были создать в нём свою ветку, совершая по ходу соревнования загрузки (коммиты) в свой локальный репозиторий.

Кроме кода, участники могли снабдить решение текстовыми комментариями и описанием в свободном стиле, например, в виде диаграмм последовательности исполнения. Таким образом, для организации соревнования мы применяли известные программные решения в сфере индустриальной разработки.

По своему опыту наблюдения и участия в такого рода мероприятиях, авторы могут заключить, что участники приходят на соревнования с разными целями: кто-то хорошо программирует и просто ждет интересных задач, чтобы их решить и победить, кому-то нужно присоединиться к команде и чему-то научиться, сделав небольшую часть работы (это могут быть не только разработчики, но и дизайнеры или начинающие менеджеры проектов), а кому-то нравится находиться в ИТ-сообществе единомышленников. На таких мероприятиях можно встретить, в основном, студентов старших курсов, однако там бывают и недавние выпускники, представляющие компании, как в качестве участников, так и в качестве менторов или членов жюри оценки проектов. Рутинная работа разработчиком в компании после динамичной и разнообразной жизни в университете со сдачей множества работ и постоянными дедлайнами рождает тягу к вызовам, которые могут обеспечить соревнования.

На сайте мероприятия мы разместили опросник для предварительной регистрации с использованием готовых средств создания сайтов, форм и таблиц Google Site и Google Docs.

Это позволило нам проанализировать состав и интересы участников еще до начала соревнования.

По результатам предварительного опроса на сайте соревнований (опросник в виде Google-формы заполнили 35 человек), участники соревнования имели следующие аффилиации:

- Университет Иннополис;
- Университет ИТМО;
- Новосибирский государственный университет (НГУ);
- Санкт-Петербургский государственный университет (СПбГУ);
- Санкт-Петербургский политехнический университет Петра Великого (СПбПУ);

- Высшая Школа Экономики (ВШЭ);
- Астра Линукс (Группа Астра), Москва.

Следует отметить, что все без исключения участники были из российских вузов или организаций, при этом один из участников представлял Египет, будучи магистрантом Университета Иннополис.

Распределение участников по предпочтениям физического присутствия показано на рис. 1. Большинство участников были из Сибирских и Санкт-Петербургских вузов, где преподают организаторы соревнования, что объясняет получившееся распределение. Участники на месте были из Университета Иннополис, где проводился связанный семинар PSSV. Кроме того, один участник приехал для очного участия на место проведения соревнований из Москвы.

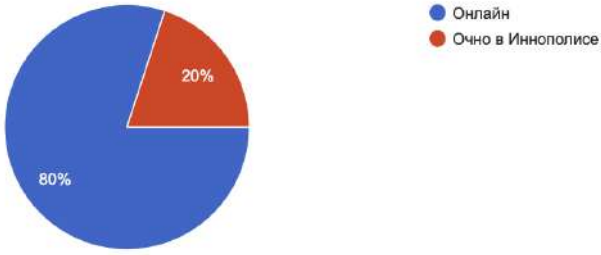


Рис. 1. Распределение участников по физическому присутствию
Fig. 1. Distribution of participants by physical presence

На рис. 2 показано распределение предпочтения участниками методов формальной верификации. Такие результаты отчасти можно снова объяснить тем, что большинство участников пришли на эти соревнования по приглашению от организаторов, которые преподают спецкурсы по проверке моделей, а также, возможно, из-за более интересных предварительных формулировок задач.

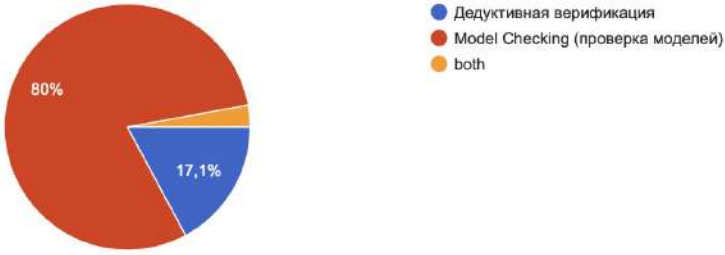


Рис. 2. Распределение участников по предпочтительному методу верификации
Fig. 2. Distribution of participants by a preferred verification method

Распределение по задачам, которые интересны участникам до начала контеста и объявления полных текстов задач, представлены на рис. 3.

В результате анализа этой статистики организаторы приняли решение о необходимости предоставления участниками решений как задачи по проверке моделей, так и задачи по дедуктивной верификации (а не одну из них), чтобы расширить компетенции участников. Соревнования были командными, и в состав команды могло входить от 1 до 3 человек.

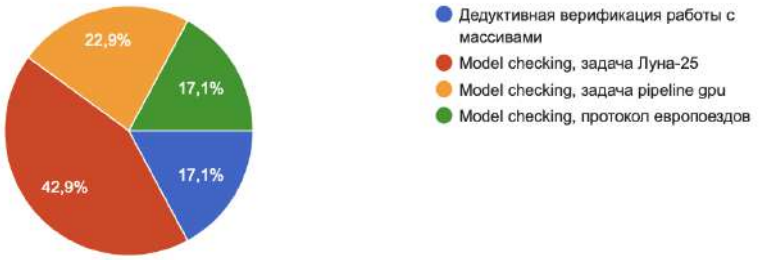


Рис. 3. Распределение участников по предпочтительному выбору задачи
Fig. 3. Distribution of participants by a preferred task

4. Обзор предложенных задач

Первоначальная концепция соревнования подразумевала, что задачи будут предложены как индустриальными партнерами исходя из их потребностей, так и организаторами на основе их научных интересов и публикаций, опыта разработки и преподавания курсов по формальной верификации. К сожалению, в этом году индустриальные партнеры не принимали участия в постановке задач, поэтому все задания были предложены организаторами-авторами статьи.

Поскольку формальная верификация тесно связана с надежными системами, а в отечественной прессе в августе и сентябре 2023 года циркулировало большое количество материалов по поводу недавнего крушения аппарата миссии “Луна-25” (например, [75]), то организаторы нашли интересным предложить в качестве задачи контеста моделирование и верификацию критической программной системы такого рода, несмотря на то, что все публичные материалы расследования причин аварии лишь очень приблизительно освещают детали программного устройства аппарата. Тема моделирования и верификации программных составляющих аппарата “Луна-25” стала основной на контесте. Полные условия всех задач можно найти на сайте контеста VeHa-2023 [71].

4.1 Дедуктивная верификация

Участникам соревнования была предложена задача по дедуктивной верификации программы, предназначенной для решения одной из потенциальных проблем миссии “Луна-25”.

В абстрактной постановке опубликованная версия причины крушения аппарата состоит в том, что в массив команд с одинаковым приоритетом попала команда, приоритет которой отличается от остальных. Программа, проверяющая, есть ли в массиве хотя бы один элемент, отличающийся от остальных, могла бы позволить избежать этой проблемы. Участники должны были верифицировать такую программу в системе C-lightVer.

Вначале рассмотрим общую постановку задачи. Предусловие программы было задано заранее, чтобы участники не проводили трудоёмкую работу с низкоуровневыми деталями. В итоге, задача состояла в том, чтобы участники соревнования:

1. задали постусловие программы в виде равенства результирующей переменной программы и рекурсивной функции, моделирующей цикл программы;
2. задали определение функции, используемой в постусловии и моделирующей цикл программы;
3. верифицировали программу с помощью системы C-lightVer.

Опишем детальную постановку задачи.

В качестве входных данных задачи участникам предоставлялся файл “element_equality.c” с одинаковым содержимым (см. листинг 1)

В этом файле предоставлено определение верифицируемой функции *element_equality* и предусловие данной функции, записанное на языке Applicative Common Lisp в комментарии до определения функции. Функция *element_equality* реализует проверку равенства элементов массива на языке C. Участникам нужно было задать постусловие на языке Applicative Common Lisp внутри пустого комментария после определения верифицируемой функции.

```
/* (and (integer-listp a) (natp n) (< 0 n)
 (<= n (length a))) */
int element_equality(int *a, int n) {
    int result = 1;
    for (int i = 1; i < n; i++)
    {
        if (a[i-1] != a[i])
        {
            result = 0;
            break;
        }
    }
    return result;
}
/*  */
```

Листинг 1. Входные данные (файл “element_equality.c”)
Listing 1. Input data (file “element_equality.c”)

Для решения задачи участникам нужно было выполнить следующую последовательность шагов:

1. Отредактировать файл “element_equality.c”, добавив в него постусловие в виде равенства переменной “result” и применения функции, проверяющей равенство элементов массива от нижней до верхней границы массива, заданных в качестве аргументов.
2. Создать файл с теорией предметной области, содержащий определение функции, примененной в постусловии и возвращающей число “1”, если все элементы массива от нижней до верхней границы равны, и число “0” в иных случаях.
3. Запустить систему C-lightVer, передав ей в качестве параметров отредактированный файл “element_equality.c” и полученный файл с теорией предметной области, и проверить, что информация, полученная в результате сеанса верификации, свидетельствует о корректности программы относительно своих спецификаций.

В качестве решения задачи командам-участникам следовало предоставить отредактированный файл “element_equality.c” с заданным постусловием и полученный файл с теорией предметной области, содержащий определение примененной в постусловии функции.

Критерии для оценки количества баллов, которое получало решение по дедуктивной верификации, были следующие:

1. правильно ли задано постусловие;
2. правильно ли задана теория предметной области с определением примененной в постусловии функции;
3. позволяет ли решение осуществить верификацию в системе.

Отметим, что относительно простая задача по дедуктивной верификации была уравнена в баллах с задачами по проверке на модели (model checking), так как для решения данной задачи участникам пришлось изучать такую сложную научную и практическую область, как дедуктивная верификация программ, а также использование системы C-lightVer. В будущем мы планируем определять количество баллов за задачи более гибкими способами.

4.2 Метод проверки моделей (Model Checking)

Для метода проверки модели были предложены три задачи. Для задачи о станции “Луна-25” было необходимо на языке Promela формализовать взаимодействие модулей станции и найти сценарий, приводящий к ошибке, описанной в СМИ, а затем предложить исправленную модель, в которой нет ошибочных сценариев взаимодействия модулей. Уровень этой задачи оценивался организаторами как простой.

В задаче о моделировании pipeline GPGPU было необходимо, описать pipeline процессов, происходящих в GPU при решении задачи на SIMT ядрах согласно известным публикациям.

В задаче про протокол телеграмм для европоездов требовалось реализовать алгоритм работы с бинарными сообщениями согласно официальному документу и проверить корректность кодирования/декодирования.

Последние две задачи имеют высокий уровень сложности и могут рассматриваться как примеры промышленного моделирования и верификации.

Для участия в соревновании достаточно было решить одну из трех задач. Несмотря на то, что уровень задач разный, для сложных задач было достаточно предложить простую, начальную версию решения. Таким подбором задач организаторы соревнования хотели стимулировать интерес участников к верификации промышленных систем.

При оценке решений по методу проверки модели использовались следующие критерии:

1. качество моделирования;
2. формулирование требований к модели на языке темпоральной логики;
3. наличие корректных вычислений и анализ контрпримера, выдаваемого верификатором;
4. качество алгоритма, исправляющего ошибку.

Две сложные задачи оценивались только по двум первым критериям.

4.2.1 Моделирование проблемы миссии “Луна-25”

В качестве основы для задачи были выбраны сообщения СМИ о крушении станции “Луна-25”, описывающие состав модулей аппарата и предполагаемые причины крушения [76]:

19 августа 2023 года при выдаче корректирующего импульса для перевода космического аппарата с круговой окололунной орбиты на эллиптическую предпосадочную орбиту двигательная установка “Луны-25” проработала 127 секунд вместо запланированных 84 секунд. В итоге станция перешла на нерасчетную незамкнутую орбиту и столкнулась с лунной поверхностью.

Для моделирования в задаче были выделены значимые модули станции “Луна-25”, согласно опубликованной в СМИ открытой информации:

1. бортовой комплекс управления (БКУ);
2. блок измерения угловых скоростей (БИУС-Л – Блок Измерения Угловых Скоростей-Луна) и датчики угловых скоростей;
3. двигатель;
4. другие модули.

“Бортовой комплекс управления” обеспечивает функционирование всего космического аппарата и разных его систем. В его задачи входит управление всеми компонентами.

“БИУС-Л” предназначен для определения ориентации космической станции и определения скорости полета при помощи оптоволоконных гироскопов и акселерометра. При моделировании рассматривались только датчики угловых скоростей.

“Двигатель” отвечает за перемещение аппарата. При посадке он должен обеспечить переход аппарата на эллиптическую орбиту.

Про “Другие модули” известно, что они могут принимать и передавать данные, и этого достаточно для их моделирования.

Дополнительной, хоть и неявной компонентой, требующей моделирования, являлась среда коммуникации, в которой имеется шина передачи команд и шина передачи данных.

В условии задачи было предложено смоделировать следующую версию ошибки, произошедшей на станции “Луна-25” (мы намеренно не рассматриваем версии, связанные с возможными ошибками при планировании процессов в операционных системах реального времени, а основываемся на одной из версий из СМИ). По неизвестным причинам БИУС-Л стал передавать слишком много “нулевых сигналов”. Этим он перегрузил входящий поток данных на БКУ. При такой проблеме БКУ подал БИУС-Л команду о перезагрузке. В массиве входящих команд БИУС-Л присутствовали команды об отслеживании угловых скоростей и о перезагрузке. БИУС-Л недетерминированно выбрал в массиве команду о перезагрузке. При перезагрузке БИУС-Л очистил свой массив входящих команд и таким образом пропустил команду об отслеживании угловых скоростей. Поскольку на БКУ не был запрограммирован анализ ответной реакции БИУС-Л на команду включения акселерометров, то БКУ не смог получить информацию о фактических значениях угловых скоростей. В результате при достижении необходимой орбиты БКУ не передал сигнал двигателю на своевременное отключение.

Для решения задачи от участников требовалось:

1. Спроектировать взаимодействие компонент, чтобы оно приводило к аварии в результате неправильного расположения в пуле данных БИУС-Л команды об отслеживании угловых скоростей аппарата. Доказать, что авария может происходить.
2. Изменить проект так, чтобы избежать аварии. Доказать, что авария не может происходить.

4.2.2 Моделирование архитектурной последовательности работы (pipeline) GPGPU

Эта задача предполагала расширение идей нашего подхода по моделированию на языке Promela поведения процессора видеокарты (GPU) при исполнении кода некоторой параллельной программы. Результирующая Promela-модель использовалась для поиска наименьшего времени исполнения этой параллельной программы, исходя из комбинаций параметров, от которых зависит время завершения исполнения [77].

В цитируемой статье мы использовали только сущности абстрактной архитектуры OpenCL [78], такие как хост, устройство, исполнительный модуль. Однако существуют работы, в которых с хорошей точностью промоделирован процесс работы реального GPU-процессора [79]. Эта модель принимает во внимание прежде всего систему команд (инструкции) и включает такие составляющие как кеш инструкций, буфер инструкций, планировщик исполнения инструкций, модуль сборки операндов и другие.

В процессорах их компоненты организуют pipeline – последовательность взаимодействий в соответствии с принципами архитектуры. Современные GPU реализуют SIMT архитектуру (single instruction, multiple thread), то есть одна и та же инструкция после декодирования исполняется на большом количестве потоков, способных выполнять простые операции, для чего необходимо наличие эффективного кэша и реализации алгоритмов работы с ветвлениями. Поскольку все эти операции представляют собой взаимодействия параллельно исполняющихся компонентов с синхронизацией, то они могут быть промоделированы на языках отвечающих подходам процессных алгебр, например алгебре CSP и языку Promela.

От участников соревнования ожидался результат, удовлетворяющий следующим пунктам:

- создана модель решения простой задачи, заданной в виде последовательности инструкций, сходных с приведенными примерами инструкций для некоторой OpenCL программы;
- взаимодействия компонентов должны соответствовать основным идеям из GPGPU Sim [79];
- все сущности должны быть описаны как параллельные процессы Promela, взаимодействующие через каналы (принятие и отправка сообщений);
- должно быть несколько экземпляров сущностей для симуляции мультипроцессоров GPU и планировщиков внутри них;
- решение должно быть сделано с предоставлением UML диаграмм деятельности по каждому компоненту.

Предлагалось верифицировать требование минимального времени исполнения решения задачи на GPU, согласно нашему подходу [77].

4.2.3 Моделирование протокола телеграмм для европоездов

Эта задача предполагала продолжение исследования по созданию выполнимой модели для побитовых операций с Eurobalise-телеграммами (передаваемыми сообщениями в современных железнодорожных системах) согласно их официальной спецификации с целью проверки кодирования и декодирования таких телеграмм [80]. В частности, ранее автором была реализована библиотека для работы с бинарными полиномами в поле GF(2) на Promela, и теперь предполагалось реализовать модель согласно предложенным в статье подходам. Планировалось, что в процессе соревнования будет проводиться обсуждение задачи с участниками, которое будет включать ознакомление с требованиями к телеграммам для бализы (передатчика) из официального стандарта и с методами работы с бинарными полиномами для вычисления контрольной суммы (CRC). В итоге ожидался какой-либо из следующих результатов:

- реализация на Promela побитовых операций работы с частями телеграмм в соответствии со стандартом с использованием библиотеки в виде макросов для полиномов;
- реализация кодирования и проверки получившихся диаграмм (возможно, сначала на языке C, с последующей формализацией на Promela с проведением абстрагирования);
- реализация декодирования;
- реализация верификации кодирования и декодирования путем случайном подмены битов в сообщении.

5. Обучение участников

Обучение участников проводилось в гибридном формате – дистанционно и на месте. В назначенное время по указанному на сайте контеста адресу были проведены лекции в Skype по дедуктивной верификации и языку Promela с записью, доступной в течение 30 дней. Кроме того, на сайте соревнования были выложены обучающие материалы. Дополнительно один из организаторов прочел лекции по практическим аспектам верификатора Spin в Университете Иннополис с трансляцией и записью в Skype. Также во время решения задач проводились онлайн и офлайн консультации.

5.1 Дедуктивная верификация

До начала соревнования были подготовлены и размещены на сайте следующие материалы [71]:

- мини-пособие для участников соревнования по дедуктивной верификации;
- презентация тьюториала для участников соревнования по дедуктивной верификации.

Кроме того, участникам соревнования было настоятельно рекомендовано установить систему C-lightVer до начала соревнования.

Пособие описывает дедуктивную верификацию, применение системы C-lightVer, и язык Applicative Common Lisp на примере решения демонстрационной задачи по верификации функции, вычисляющей сумму абсолютных значений элементов массива. Демонстрационная задача и ее решение предоставлялись участникам в качестве файлов “abs_sum.c” и “abs-sum.lisp” в дополнение к пособию.

Также отметим, что в тексте пособия были особенно подробно описаны те особенности программирования на языке Applicative Common Lisp и работы с системой C-lightVer, где при решении задачи соревнования можно было допустить ошибку.

Задача по верификации функции, вычисляющей сумму абсолютных значений элементов массива, схожа с задачей соревнования. В этих задачах цикл в программе совершает итерации над всеми элементами массива, а постусловие представляет собой равенство результата программы и применения функции к элементам массива от нижней до верхней границы массива. Кроме того, и там, и там теория предметной области содержит определение примененной в постусловии функции, моделирующей верифицируемую программу. Таким образом, обучение участников включало ознакомление с опытом организатора соревнования.

5.2 Метод проверки моделей

Материалы по теоретическим аспектам метода проверки моделей (структуры Крипке, синтаксис и семантика темпоральных логик, формализация требований) были основаны на курсе лекций летней школы Computer Science Summer in Russia 2019, а также собственных курсов авторов в НГУ. В частности, в опубликованной на сайте презентации рассматриваются известные ошибки в критическом программном обеспечении (в сфере космоса и медицины), основные методы проверки качества программного обеспечения, разница между дедуктивной верификацией и проверкой моделей, и далее приводятся примеры моделей и требований для разных классов программ. В дополнение к теоретическим материалам была выложена инструкция по запуску верификатора SPIN и среды iSpin в среде Windows (обычно это требует установки MinGV + gcc, что не всегда очевидно студентам). Тьюториал к соревнованию по методам проверки моделей включал обзор синтаксиса и семантики языка Promela верификатора Spin и основные способы работы с этим инструментом.

Также на сайт была выложена глава про метод проверки моделей из книги [81] авторов статьи, в которой кратко рассмотрены теоретические сведения о методе, работа с верификатором SPIN и элементы языка Promela.

Этот материал был использован для проведения лекции по практическим аспектам Model Checking, где участникам была показана сборка SPIN с его GitHub-репозитория [82] с помощью make в Unix-среде, написание кода на входном языке Promela в среде Visual Studio Code и запуск моделей некоторых коммуникационных протоколов в среде iSpin, проверка LTL свойств и способ задания параметров верификатора, связанных с используемой памятью, для проверки больших моделей.

5.3 Консультации

Поскольку контест проводился в основном в выходные и онлайн, а исходный GitHub-репозиторий для загрузки решений клонировали не все участники, то организатором сложно было отследить текущее состояние решения задач. Поэтому была организована очная консультация с объявлением в Telegram-канале соревнования, на которую пришли участники двух команд из Иннополиса. Участники консультации сообщили, что, по их мнению, задачи сформулированы нечетко. Они также отметили, что в задаче о станции “Луна-25” причины аварии в разных отчетах в СМИ не всегда соответствуют информации в описании задачи от организаторов. На вопрос о способе недетерминированного появления ошибки в модели системы, участникам было указано на возможности недетерминизма в операторах *if*, *do* и *select* [83]. Относительно задачи GPGPU участники высказали мнение, что задача слишком сложная, так как нужно долго изучать предварительные материалы, поэтому команда не успеет предоставить решение должного качества, так как приходится ещё заниматься задачей по дедуктивной верификации. Тем не менее, участники заверили, что в каком-то виде модели будут построены, так как с синтаксисом и семантикой языка Promela они в целом разобрались. Дополнительно консультант подчеркнул, что в нашем соревновании в решениях участников не ожидается заранее определенного протокола взаимодействия процессов системы, этот протокол они должны придумать и обосновать в своем решении.

Также позже состоялась онлайн-консультация в Skype, на которой желающие получили разъяснения по формулировке задачи о станции “Луна-25” и аспектам взаимодействия процессов через каналы в Promela.

6. Проведение соревнования

Соревнование по дедуктивной верификации оценивал Д. А. Кондратьев, а по методу проверки модели – Н. О. Гаранина, С. М. Старолетов, И. В. Шошмина.

6.1 Соревнование по дедуктивной верификации

Из пятнадцати команд контеста решения задачи по дедуктивной верификации прислали тринадцать команд.

Из них девять команд набрали максимальные десять баллов, три команды – девять баллов и одна команда – восемь баллов. Отдельно итоги соревнования по дедуктивной верификации в рамках VeHa-2023 не подводились, эти баллы были учтены в итоге контеста в целом.

Самым большим недостатком решений задачи было использование единицы вместо нуля в качестве значения нижней границы в применении функции в постусловии. Можно предположить, что к появлению такой ошибки привело использование единицы в качестве значения счетчика цикла в верифицируемой программе. Но итерации цикла в верифицируемой программе покрывают массив, начиная не с элемента с индексом *1*, а с элемента с индексом *0*, так как в качестве индексов в данном цикле используются выражения *i-1* и *i*.

Абсолютное большинство команд прислало решения, которые либо совпадают с точностью до имен идентификаторов с нашим эталонным решением, либо довольно близки к нему, поэтому здесь мы приводим тексты файлов этого решения “element_equality.c” и “element-equality.lisp”. Содержимое файла “element_equality.c” эталонного решения на листинге 2.

Эта часть решения состоит в задании постусловия в виде равенства применения функции *element-equality* и переменной *result* внутри комментария после определения верифицируемой функции. Аргументами применения *element-equality*, соответствующими нижней и верхней границе, являются выражения *0* и *(- n 1)*. Ошибка, о которой упоминалось ранее, состоит в использовании в качестве таких аргументов выражений *1* и *(- n 1)*.

```
/* (and (integer-listp a) (natp n) (< 0 n)
 (<= n (length a))) */
int element_equality(int *a, int n) {
  int result = 1;
  for (int i = 1; i < n; i++) {
    if (a[i-1] != a[i]) {
      result = 0;
      break;
    }
  }
  return result;
}
/* (= (element-equality 0 (- n 1) a) result)
*/
```

Листинг 2. Файл эталонного решения (“element_equality.c”)
Listing 2. Reference solution file (“element_equality.c”)

На листинге 3 приведено определение функции *element-equality* из постусловия в файле “element-equality.lisp”.

```
(defun element-equality(i j a)
 (if (or (not (natp i)) (not (natp j))) 0
   (if (>= i j) 1
     (if (not (equal (nth (- j 1) a)
      (nth j a))) 0
       (element-equality i (- j 1) a)))))
```

Листинг 3. Функция “element-equality”, примененная в постусловии (файл “element-equality.lisp”)
Listing 3. “element-equality” function applied in the postcondition (“element-equality.lisp”)

Функция *element-equality* возвращает число *1*, если все элементы последовательности от нижней до верхней границы равны, и число *0* в иных случаях. Отметим, что функция *element-equality* определена рекурсивно с уменьшением верхней границы на единицу при каждом рекурсивном вызове.

Рассмотрим решения задачи, которые организатор посчитал интересными. Все такие решения содержат особенности при решении следующей подзадачи: каким образом сделать возвращаемым значением функции из постусловия целое число *1* или *0* вместо того, чтобы с помощью конъюнкции равенства элементов массива и рекурсивного применения данной функции сделать возвращаемым значением булево значение. В эталонном решении эта подзадача решалась с помощью использования в определении функции из постусловия вместо конъюнкции условного выражения, показанного на листинге 4.

```
(if (>= i j) 1
  (if (not (equal (nth (- j 1) a)
   (nth j a))) 0
    (element-equality i (- j 1) a)))
```

Листинг 4. Условное выражение из функции “element-equality”, примененной в постусловии
Listing 4. Conditional expression in the “element-equality” function applied in the postcondition

В предварительном решении команды NastyaKrass (Группа Астра, Москва) было предложено использовать вместо этого выражения сравнение с помощью побитовых операций элементов массива и сравнение с помощью побитовых операций полученного результата с рекурсивным применением функции. Такая идея решения основана на том, что элементы массива представляют собой целые числа, сравнение которых можно реализовать с помощью побитовых операций вместо равенства. Но так как NastyaKrass решала задачу соревнования по дедуктивной верификации вручную (без использования программной системы C-lightVer), то получившееся решение содержит недочеты. За попытку решения задачи вручную (без использования C-lightVer) и за попытку решить задачу интересным способом команда NastyaKrass получила почетную грамоту “За волю к победе”.

Команда Cache-Invalidation (Университет Иннополис, Иннополис) предложила использовать иное выражение вместо вышеприведенного условного выражения (см. листинг 5).

```
(if (> i j) 1
(if (= i j) 1
  (min
   (if (= (nth j a) (nth (- j 1) a)) 1 0)
   (element-equality i (- j 1 ) a))))
```

Листинг 5. Альтернативный вариант условного выражения из функции, примененной в постусловии
Listing 5. Alternate version of conditional expression from the function applied in the postcondition

Это решение основано на выборе минимального элемента между результатом рекурсивного применения функции из постусловия и условным выражением со значением в виде числа *1* при равенстве элементов массивов и со значением в виде числа *0* в ином случае. Выбор минимального элемента в данном решении осуществляется с помощью стандартной функции *min* из языка Applicative Common Lisp. Получившееся решение можно развить следующим образом: сначала генерировать по массиву последовательность элементов, в которую будет попадать число *1*, если соседние элементы массива равны, и число *0* в ином случае, а потом искать минимум в полученной последовательности. Особенно отметим, что к функциям, используемым для задания спецификаций, не применяются требования к эффективности реализации, так как в процессе дедуктивной верификации исполнение этих функций не происходит. Поэтому, такое решение имеет полное право на существование, а команда Cache-Invalidation получила почетную грамоту “За оригинальное решение задачи “Луна-25” (дедуктивная верификация)”.

Элегантное решение предложил участник команды Kokorin (Группа Астра, Москва). С одной стороны, функция из постусловия этого решения зависит только от верхней границы. С другой стороны, рассмотрим нетривиальное определение функции *element-equality*, примененной в постусловии, показанное на листинге 6.

```
(defun element-equality-bool(j a)
 (if (not (natp j)) nil
 (if (= 0 j) t
   (and (= (nth (- j 1) a) (nth j a))
     (element-equality-bool (- j 1) a)))))

(defun element-equality(n a)
  (if (element-equality-bool (- n 1) a) 1 0))
```

Листинг 6. Функция “element-equality”, предложенная участником Kokorin (Москва)
Listing 6. “element-equality” function offered by participant Kokorin (Moscow)

Функция *element-equality* определена с помощью применения функции *element-equality-bool*. Таким образом, это единственное решение, где в теории предметной области было определено более одной функции. Использование двух функций позволило разбить решение задачи на две подзадачи: функция *element-equality-bool* возвращает истинное или ложное значение в зависимости от того, все ли элементы последовательности до верхней границы равны, а функция *element-equality* возвращает число *1* или *0* в зависимости от значения применения функции *element-equality*. Функция *element-equality* определена нерекурсивно, а функция *element-equality-bool* определена рекурсивно с помощью конъюнкции своего рекурсивного применения и равенства элементов массива. Также отметим, что кроме декомпозиции задачи, команде Kokorin удалось написать простое и понятное пояснение в дополнение к решению задачи. Так как в дополнение к такому элегантному решению этой команде удалось предоставить оригинальное решение задачи соревнования по проверки моделей, то Kokorin получил почетную грамоту “За оригинальное решение задач контеста”.

6.2 Соревнование по проверке моделей (Model Checking)

Все пятнадцать команд выполняли задание по методу проверки модели.

6.2.1 Ошибки и сложности, возникшие у участников соревнования

Рассмотрим основные трудности, с которыми столкнулись участники в части соревнования, связанного с методом проверки модели, на примере задачи о станции “Луна-25”.

Предложенная некорректная последовательность событий на станции “Луна-25” содержала ошибку синхронизации, что характерно для распределенных алгоритмов. Такой тип ошибок удобно моделируется с помощью языка Promela, поскольку Promela – язык межпроцессного взаимодействия.

При проверке моделирования оценивалось качество моделирования структур данных, качество моделирования механизмов синхронизации, соблюдение алгоритмов работы отдельных процессов/модулей, корректность использования механизмов абстракции – атомарности, чередования. Инструменты моделирования очень мощные, возникает соблазн упростить модель, то есть сформировать простую модель на высоком уровне абстракции. Такая модель может быть верифицируемой, но не будет в достаточной степени отражать деталей реализации.

Во многих моделях были введены лишние параметры, например, несколько значений показаний угловых скоростей. В данной постановке задачи алгоритм не предусматривал разной реакции на разные значения угловой скорости. Для самого тонкого моделирования достаточно было трех значений: 0 – скорость не проверяется, 1 – скорость проверяется, но аппарат не достиг окололунной орбиты, 2 – скорость проверяется, аппарат достиг окололунной орбиты. Более того, можно было обойтись и двумя значениями: 0 – скорость не проверяется, 1 – скорость проверяется, а переключение БКУ по достижению окололунной орбиты моделировать недетерминированно. Правда, такое моделирование потребовало бы более тонкого задания свойств. Ввод ненужных параметров или необоснованных диапазонов переменных приводят к экспоненциальному взрыву пространства состояний модели. Причина состоит в том, что состояние структуры Крипке, которая строится верификатором SPIN, содержит значения переменных и счетчики операторов, поэтому размеры структуры зависят от диапазона значений переменных.

Другим ненужным для верификации параметром были счетчики времени. В формулировке алгоритмов и свойств время не предусматривалось. Здесь рассматривался подход, сформулированный Л. Лэмпортом, что качественные распределенные алгоритмы должны работать правильно всегда, независимо от скорости процессов. Кроме того, линейная темпоральная логика (LTL), для которой реализован метод проверки модели в SPIN, предназначена для задания причинно-следственных зависимостей во временной последовательности событий. Количественные значения времени в этой логике не предусмотрены. Типичным вариантом моделирования событий, которые интуитивно связываются со временем, являются недетерминированные переходы, например, канал недетерминированно отправляет сообщения или теряет их.

Еще одной проблемой стало моделирование среды коммуникации. В условии задачи было сказано, что среда коммуникации проходит по циклу все процессы и в одном цикле либо получает, либо отправляет сообщения. Многие из участников ошибочно увеличили гранулярность модели с помощью операторов *atomic*, включив получение и отправку сообщений в один шаг. Кроме того, многие модели реализовали работу алгоритма, как последовательную работу процессов, то есть до того, как один процесс не выполнит свой шаг, связанный с обработкой сообщений, следующий процесс не приступает к своей работе. Таким образом, участники ошибочно посчитали, что среда коммуникации накладывает ограничение на последовательность работы остальных модулей. Хотя последовательный опрос каналов не означает последовательную работу модулей. Чтобы избежать такой ошибки

при моделировании, можно выделить среду коммуникации в отдельный процесс и аккуратно работать с гранулярностью.

Другим оцениваемым параметром было составление требования на языке LTL. Сложность заключается в переходе от естественного языка к формальному, при этом необходимо не ошибиться, составить требование в соответствии с формальной семантикой.

Некоторые участники не смогли составить требования. Большинство из составленных требований формулировались в терминах реализации модели и в виде “когда-нибудь БИУС-Л будет регистрировать угловую скорость”. При этом только несколько участников смогли отразить, что требование должно выполняться многократно и по условию: “всегда, если команда перехода на эллиптическую орбиту дана, то этот переход будет завершен”. Надо заметить, что на практике такой формулировки было бы недостаточно, поскольку, как правило, в подобных системах предусмотрена штатная обработка ошибок, а, значит, условия только с подачей команды будет недостаточно.

В число оцениваемых параметров входила работоспособность модели – то есть наличие бесконечных вычислений, а также анализ контрпримеров, то есть вычислений, нарушающих свойство.

Приблизительно трети участников не удалось разработать работоспособную модель, а именно, даже в режиме симуляции возникали взаимные блокировки процессов. Многие из этих команд не смогли обнаружить неработоспособность своих моделей самостоятельно. Здесь сказывается отсутствие опыта анализа контрпримеров, а также незнание базовых возможностей верификатора SPIN: в SPIN есть проверка на отсутствие взаимных блокировок процессов. Анализ причин, вызвавших блокировки процессов, показал, что некоторые участники не освоили семантики операторов в Promela: SPIN проверяет все операторы Promela на выполнимость, если оператор по какой-либо причине выполнить нельзя, то выполнение процесса блокируется.

В частности, ошибка состояла в том, что оператор выбора при невыполнимости всех условий блокирует процесс, а не продолжает вычисление, как это происходит в языках высокого уровня.

Всего 4 работы представили контрпримеры, сгенерированные SPIN при опровержении свойства. Ни один из участников не представил анализа контрпримеров. Некоторые из заявленных контрпримеров продемонстрировали непонимание авторами ошибок, найденных верификатором. Заметим, что несмотря на то, что в методе проверки модели верификатор сам генерирует контрпримеры, их анализ не тривиален. Во-первых, контрпример может свидетельствовать о нарушении базовых свойств распределенных систем, например, взаимная блокировка процессов или несоблюдение свойств на несправедливых вычислениях. Во-вторых, контрпример может свидетельствовать как о нарушении моделью свойства, так и о неправильно составленном свойстве при относительно корректной модели.

6.2.2 Интересные идеи и решения

Всего одной команде удалось полностью решить задачу о миссии “Луна-25”. Это команда VeriCheckTeam (СПбПУ).

На диаграмме последовательности сообщений процессов модели, разработанной участниками VeriCheckTeam (рис. 4), показан один из путей, приводящих к проблеме: команда *enable_bius* была отправлена, однако из-за переполнения канала данных от БИУС-Л нулями, отправлена команда *reset*, которая очистила буфер команд вместе с *enable_bius* и БИУС-Л не запустил измерение угловой скорости. Такая ошибка легко находится в результате верификации модели относительно LTL формулы

$$\mathbf{G}\ ((\text{isBiusEnableCommandSend} \,\&\&\, \text{simulationState} == 1) \rightarrow \mathbf{F}\ \text{isBiusEnabled}),$$

где *isBiusEnableCommandSend* – внутренняя переменная модели, устанавливаемая при каждой отправке сообщения *enable_bius, simulationState* – внутреннее состояние модели для

симуляции этого действия, а *isBiusEnabled* – переменная модели, устанавливаемая при актуальном выполнении команды *enable_bius*.

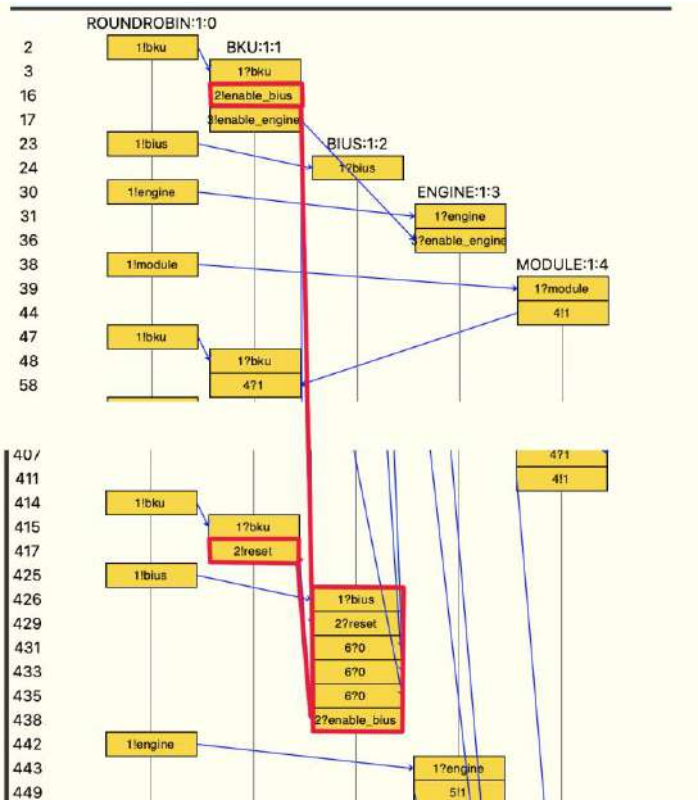


Рис. 4. Моделирование версии аварии станции в решении участников [74]
Fig. 4. Modeling a version of the Luna problem in the participants' solution [74]

Участники команды смогли смоделировать не только исходную задачу, но и исправленное решение. В качестве исправления VeriCheckTeam предложили на стороне БКУ обрабатывать за раунд не по одному сообщению, а все поступившие сообщения. В результате, по мнению команды, у БКУ не будет возникать переполнения, а, значит, не возникнет необходимость сбрасывать состояние БИУС-Л. Предложенное решение не совпадает с решением организаторов соревнования, последние предполагали изменить алгоритм взаимодействия с БИУС-Л после отправки команды на переход эллиптическую орбиту. Однако ценность решения команды VeriCheckTeam состоит в его законченности. Эта команда завоевала первое место.

Хотелось бы отметить несколько решений, представленных другими участниками.

Модель команды NSU (НГУ) отличалась системностью проектирования. Команде удалось выделить состояния каждого процесса и проработать переходы между состояниями в соответствии с описанием задачи, а также смоделировать коммуникационную среду. К сожалению, решение задачи получилось незавершенным. Эта команда завоевала третье место.

Команде Kokorin (Группа Астра) удалось разработать модель, наиболее четко соответствующую основам моделирования распределенных алгоритмов. Решение аккуратно работало с чередованием процессов и гранулярностью шагов модели. От призовых мест эту команду отделило слишком значительное упрощение алгоритма, изложенного в задаче.

Команда Maksina_Vozian_Kiseleva из СПбПУ предложила оригинальный способ очистки каналов посредством каскадной пересылки.

Некоторые из команд описали исправленное решение, согласно последовательности гипотетических взаимодействий, предложенной организаторами, но реализовать его не успели.

Команда Team Hype из Иннополиса снабдила свое, хоть и недостаточно полное решение, презентацией по найденным в сети Интернет материалам по поводу неудачных миссий в истории космонавтики с анализом причин возникших проблем.

Что касается других задач соревнования по Model Checking, по второй задаче (GPGPU) было отправлено только одно решение, предлагающее моделировать несколько планировщиков исполнения инструкций в отдельных процессах, SM модули [77] и кеш инструкции. Решение, однако, не было доделано, прежде всего, из-за того, что отправившая его команда никогда раньше не писала модели на такого рода языках (вручена почетная грамота за попытку решения). Попытки решения третьей задачи (Eurobalise), насколько нам известно, не предпринимались, но, тем не менее, очно приехавший участник заявил, что официальная спецификация этого протокола ему понравилась и он после соревнования планирует разбираться в данной теме самостоятельно.

7. Результаты и обратная связь

В конце соревнований все участники получили сертификаты участия в соревновании, дипломы за 1-3 места, а также почетные грамоты. Решение о выдаче почетных грамот в целом соответствует практикам проведения олимпиад и поощрения участников за достижения по конкретным задачам. На рис. 5 представлен процесс вручения таких грамот и дипломов победителям.

Рис. 5. Очное вручение почетных грамот участникам из Иннополиса и дипломов победителей участникам из СПбПУ и НГУ
Fig. 5. In-person presentation of certificates of honor to participants and winners

После объявления результатов был проведен анонимный опрос с целью получения обратной связи от участников. В результате было получено 9 мнений. В целом, соревнования понравились участникам (рис. 6). Все участники опроса поставили положительные оценки за соревнование, при этом хороших оценок было большинство (рис. 7).

Интересным оказалось мнение участников по поводу формулировок задач: многие отметили их неоднозначность (рис. 8). Однако заметим, что, в целом, формулировки задач по проверке моделей предполагали некоторый уровень креативности в разработке моделей, а при решении двух сложных задач ожидалось переосмысление моделей организаторов, что оказалось сложным для данного формата соревнований.

Уровень сложности задач был оценен по-разному, но в целом он оказался нормальным для большинства участников (рис. 9). Время на выполнение задач оказалось в целом достаточным, хотя были мнения и о необходимости его увеличить (рис. 10).

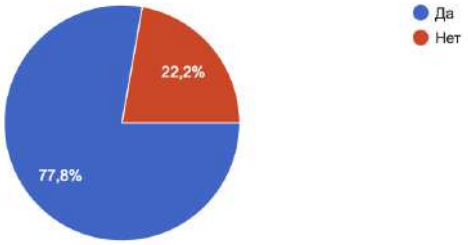


Рис. 6. Распределение участников опроса по общему впечатлению (нравится/не нравится)
Fig. 6. Distribution of survey participants by general impression

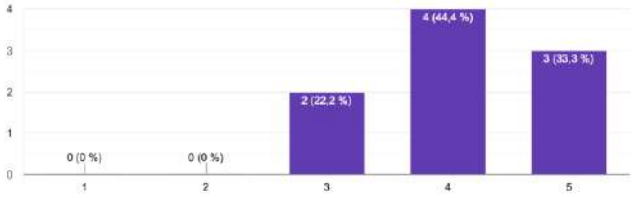


Рис. 7. Распределение участников опроса по оценкам соревнования по пятизначной шкале
Fig. 7. Distribution of survey participants according to competition ratings on a five-digit scale

Рис. 8. Распределение участников опроса по оценкам уровня формулировок задач
Fig. 8. Distribution of survey participants according to assessments of the level of task formulations

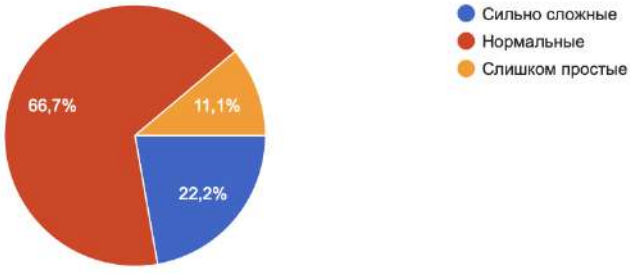


Рис. 9. Распределение участников опроса по оценкам уровня сложности задач
Fig. 9. Distribution of survey participants according to assessments of the level of difficulty of tasks

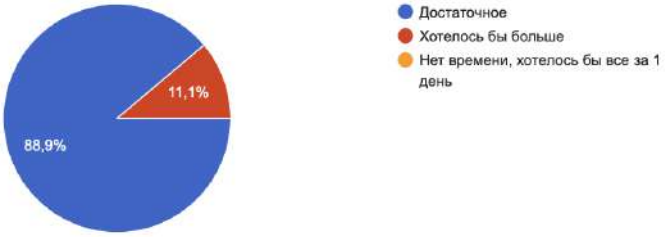


Рис. 10. Распределение участников опроса по оценкам времени на решение задач
Fig. 10. Distribution of survey participants according to estimates of time to solve problems

Среди дополнительных комментариев участники выказали желание проводить больше таких соревнований, поблагодарили организаторов за проведенное время, высказали мнение о необходимости презентации сделанных решений, а также был комментарий о необходимости использовать для дедуктивной верификации современные общеизвестные системы поддержки доказательств.

8. Заключение

Мы считаем, что соревнование VeHa-2023 прошло успешно, особенно в качестве первого опыта проведения такого рода мероприятий.

Абсолютное большинство команд справилось с предложенной на данном соревновании задачей по дедуктивной верификации. Поэтому мы пришли к выводу, что опасения, исходя из которых была выбрана относительно простая задача, оказались преувеличены, и на следующем соревновании по дедуктивной верификации в рамках контеста VeHa можно усложнить задачу и приблизить ее к миру индустриального программирования.

Решения соревнования по проверке моделей продемонстрировали не вполне радужную картину, так как почти все предоставленные модели систем и требований содержали различного рода ошибки. Но при этом участники также предлагали интересные и оригинальные способы моделирования различных составляющих систем. В целом, все участники на приемлемом уровне освоили язык Promela, однако для формирования более качественных решений требуется погружение в его тонкости и ограничения, которого сложно достичь без присутствия менторов рядом с командами.

Кроме того, соревнование показало, что требование решать задачи и по дедуктивной верификации, и по проверке моделей оказалось довольно трудным для исполнения, что, возможно, привело к снижению качества предоставленных решений и к отказу участников от решения сложных задач.

Уровень следующего контеста VeHa-2024 планируется поднять, взяв в качестве задач интересные проблемы верификации программ из области индустриального программирования, а также расширив список используемых инструментов верификации, включая собственные решения. Также мы будем учитывать нынешний опыт проведения, пожелания и предложения участников контеста VeHa-2023.

Список литературы / References

[1]. PSSV 2023. Available at: https://persons.iis.nsk.su/en/PSSVfrom2022towards2023, accessed Jul 06,2024.
[2]. Zavyalov A., Staroletov S. Flovver: A graphical functional language with a compiler focused on recursion optimization. Computing, Telecommunications and Control, vol. 16, no. 1, pp. 46–59, 2023.
[3]. Hackathon Barnaul 2023. Available at: https://innovaltai.ru/sobytiya/detail.php?ID=3161, accessed Jul 06, 2024.
[4]. Participants in the Neftecode hackaton have developed solutions that will help build more durable roads in Russia, 2022. Available at: https://news.itmo.ru/ru/startups_and_business/partnership/news/12855/, accessed Jul 06, 2024.

[5]. Tender Hack, 2023. Available at: https://tenderhack.ru/tenderhack-kazan.pdf?roistat_visit=1154183, accessed Jul 06, 2024.
[6]. Sadovykh A., Beketova M., Khazeev M. Hackathons as a part of software engineering education: Case in tools example. In Proc. Frontiers in Software Engineering Education: First International Workshop, FISEE 2019, Villebrumier, France, 2019, Invited Papers 1. November 11–13, Springer, 2020, pp. 232–245.
[7]. Hoare C. A. R. An axiomatic basis for computer programming. Communications of the ACM, vol. 12, no. 10, pp. 576–580, 1969.
[8]. Apt K. R., Olderog E.-R. Assessing the success and impact of Hoare's logic. In Proc. Theories of Programming: The Life and Works of Tony Hoare, 2021, pp. 41–76.
[9]. Apt K. R., Olderog E.-R. Fifty years of Hoare's logic. Formal Aspects of Computing, vol. 31, no. 6, pp. 751– 807, 2019.
[10]. Filliâtre J.-C. Deductive software verification. International Journal on Software Tools for Technology Transfer, vol. 13, no. 5, pp. 397–403, 2011.
[11]. Clarke J., Emerson E. Design and synthesis of synchronization skeletons using branching time temporal logic. In proc. Logic of Programs: Workshop, Yorktown Heights, ser. LNCS. Springer, 1981, vol. 131, pp. 52–71.
[12]. Quielle J., Sifakis J. Specification and verification of concurrent systems in CESAR. In Proc. of the 5th International Symposium on Programming, ser. LNCS. Springer, 1982, vol. 137, pp. 337–350.
[13]. Hähnle R., Huisman M. Deductive software verification: From pen-and-paper proofs to industrial tools. In Proc. Of Computing and Software Science, ser. LNCS. Springer, 2019, vol. 10000, pp. 345-373.
[14]. Furia C. A., Meyer B., Velder S. Loop invariants: Analysis, classification, and examples. ACM Computing Surveys, vol. 46, no. 3, pp. 1–51, 2014.
[15]. Ernst G. Loop verification with invariants and contracts. In Proc. Verification, Model Checking, and Abstract Interpretation, ser. LNCS. Springer, 2022, vol. 13182, pp. 69–92.
[16]. First E., Brun Y. Diversity-driven automated formal verification. In Proc. of the 44th International Conference on Software Engineering, 2022, pp. 749–761.
[17]. Maryasov I. V., Nepomniaschy V. A., Promsky A. V., Kondratyev D. A. Automatic C program verification based on mixed axiomatic semantics. Automatic Control and Computer Sciences, vol. 48, no. 7, pp. 407–414, 2014.
[18]. Kondratyev D. A., Promsky A. V. Developing a self-applicable verification system. Theory and practice. Automatic Control and Computer Sciences, vol. 49, no. 7, pp. 445–452, 2015.
[19]. Kondratyev D. A., Nepomniaschy V. A. Automation of C program deductive verification without using loop invariants. Programming and Computer Software, vol. 48, no. 5, pp. 331–346, 2022.
[20]. Nepomniaschy V. A., Anureev I. S., Mikhailov I. N., Promskii A. V. Towards verification of C programs. C-Light language and its formal semantics. Programming and Computer Software, vol. 28, no. 6, pp. 314–323, 2002.
[21]. Nepomniaschy V. A., Anureev I. S., Promskii A. V. Towards verification of C programs: Axiomatic semantics of the C-kernel language. Programming and Computer Software, vol. 29, no. 6, pp. 338–350, 2003.
[22]. Cuoq P., Monate B., Pacalet A., Prevosto V. Functional dependencies of C functions via weakest pre-conditions. International Journal on Software Tools for Technology Transfer, vol. 13, no. 5, pp. 405–417, 2011.
[23]. Correnson L. Qed. Computing what remains to be proved. In Proc. NASA Formal Methods, ser. LNCS. Springer, 2014, vol. 8430, pp. 215–229.
[24]. Baudin P., Bobot F., Bühler D., Correnson L., Kirchner F., Kosmatov N., Maroneze A., Perrelle V., Prevosto V., Signoles J., Williams N. The dogged pursuit of bug-free C programs: the Frama-C software analysis platform. Communications of the ACM, vol. 64, no. 8, pp. 56–68, 2021.
[25]. Mandrykin M. U., Khoroshilov A. V. High-level memory model with low-level pointer cast support for Jessie intermediate language. Programming and Computer Software, vol. 41, no. 4, pp. 197–207, 2015.
[26]. Mandrykin M. U., Khoroshilov A. V. Region analysis for deductive verification of C programs. Programming and Computer Software, vol. 42, no. 5, pp. 257–278, 2016.
[27]. Mandrykin M. U., Khoroshilov A. V. Towards deductive verification of C programs with shared data. Programming and Computer Software, vol. 42, no. 5, pp. 324–332, 2016.
[28]. Efremov D., Mandrykin M., Khoroshilov A. Deductive verification of unmodified Linux kernel library functions. In Proc. Leveraging Applications of Formal Methods, Verification and Validation. Verification, ser. LNCS. Springer, 2018, vol. 11245, pp. 216–234.

[29]. Volkov G., Mandrykin M. Efremov D. Lemma functions for Frama-C: C programs as proofs. In Proc. 2018 Ivannikov Ispras Open Conference (ISPRAS), 2018, pp. 31–38.
[30]. Sammler M., Lepigre R., Krebbers R., Memarian K., Dreyer D., Garg D. RefinedC: automating the foundational verification of C code with refined ownership types. In Proc. of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation, 2021, pp. 158–174.
[31]. Lepigre R., Sammler M., Memarian K., Krebbers R., Dreyer D., and Sewell P. VIP: verifying real-world C idioms with integer-pointer casts. In Proc. vol. 6 POPL, 2022, pp. 1–32.
[32]. Kondratyev D. A., Maryasov I. V., Nepomniaschy V. A. The automation of C program verification by the symbolic method of loop invariant elimination. Automatic Control and Computer Sciences, vol. 53, no. 7, pp. 653–662, 2019.
[33]. Nepomniaschy V. A. Symbolic method of verification of definite iterations over altered data structures. Programming and Computer Software, vol. 31, no. 1, pp. 1–9, 2005.
[34]. Moore J. S. Milestones from the pure Lisp theorem prover to ACL2. Formal Aspects of Computing, vol. 31, no. 6, pp. 699–732, 2019.
[35]. Cimatti A., Clarke E., Giunchiglia E., Giunchiglia F., Pistore M., Roveri M., Sebastiani R., Tacchella A. NuSMV Version 2: An OpenSource Tool for Symbolic Model Checking. In Proc. International Conference on Computer-Aided Verification (CAV 2002), ser. LNCS, vol. 2404. Copenhagen, Denmark: Springer, 2002.
[36]. Kwiatkowska M., Norman G., Parker D. PRISM 4.0: Verification of probabilistic real-time systems. In Proc. Computer Aided Verification, ser. LNCS. Springer Berlin Heidelberg, 2011, pp. 585–591.
[37]. Visser W., Păsăreanu C., Khurshid S. Test input generation with java PathFinder. In Proc. of the ACM/SIGSOFT International Symposium on Software Testing and Analysis. ACM Press, 2004, pp. 97– 107.
[38]. Hessel A., Larsen K., Mikucionis M., Nielsen B., Pettersson P., Skou A. Testing real-time systems using Uppaal. In Proc. Formal Methods and Testing, 2008, pp. 77–117.
[39]. Kuppe M. A., Lamport L., Ricketts D. The TLA+ toolbox. In Proc. Fifth Workshop on Formal Integrated Development Environment, F-IDE@FM 2019, Porto, Portugal, 7th October 2019, ser. EPTCS, R. Monahan, V. Prevosto, and J. Proenca, Eds., vol. 310, 2019, pp. 50–62.
[40]. Holzmann G. J. The Spin model checker, primer and reference manual. 2003.
[41]. Victor Duarte da Silva. Syntax highlighting for Promela and Spin Simulation on debugger. Available at: https://marketplace.visualstudio.com/items?itemName=dsvictor94.promela, accessed Jul 06, 2024.
[42]. Jacobs B., Kiniry J. Warnier M. Java program verification challenges. In Proc. Formal Methods for Components and Objects, ser. LNCS. Springer, 2003, vol. 2852, pp. 202–219.
[43]. Leavens G. T., Leino K. R. M., Müller P. Specification and verification challenges for sequential object-oriented programs. Formal Aspects of Computing, vol. 19, no. 2, pp. 159– 189, 2007.
[44]. Leino K. R. M., Moskal M. VACID-0: verification of ample correctness of invariants of data-structures, edition 0. In Proc. of Tools and Experiments Workshop at VSTTE, 2010.
[45]. Weide B. W., Sitaraman M., Harton H. K., Adcock B., Bucci P., Bronish D., Heym W. D., Kirschenbaum J., Frazier D. Incremental benchmarks for software verification tools and techniques. In Proc. Verified Software: Theories, Tools, Experiments, ser. LNCS. Springer, 2008, vol. 5295, pp. 84–98.
[46]. Huisman M., Klebanov V., Monahan R. VerifyThis 2012. International Journal on Software Tools for Technology Transfer, vol. 17, no. 6, pp. 647–657, 2015.
[47]. Hoare C. A. R., Leavens G. T., Shankar N. The verified software initiative: A manifesto. ACM Computing Surveys, vol. 41, no. 4, pp. 1–8, 2009.
[48]. Müller P., Shankar N. The first fifteen years of the verified software project. In Proc. Theories of Programming: The Life and Works of Tony Hoare, 2021, pp. 93–124.
[49]. Klebanov V., Müller P., Shankar N., Leavens G. T., Wüstholz V., Alkassar E., Arthan R., Bronish D., Chapman R., Cohen E., Hillebrand M., Jacobs B., Leino K. R. M., Monahan R., Piessens F., Polikarpova N., Ridge T., Smans J., Tobies S., Tuerk T., Ulbrich M., Weiß B. The 1st verified software competition: Experience report. In Proc. FM 2011: Formal Methods, ser. LNCS. Springer, 2011, vol. 6664, pp. 154 – 168.
[50]. Filliâtre J.-C., Paskevich A., Stump A. The 2nd verified software competition: Experience report. In Proc. of the 1st International Workshop on Comparative Empirical Evaluation of Reasoning Systems, ser. CEUR Workshop Proceedings. RWTH Aachen University, 2012, vol. 873, pp. 36–49.
[51]. Bormer T., Brockschmidt M., Distefano D., Ernst G., Filliâtre J.-C., Grigore R., Huisman M., Klebanov V., Marché C., Monahan R., Mostowski W., Polikarpova N., Scheben C., Schellhorn G., Tofan B.,

Tschannen J., Ulbrich M. The COST IC0701 verification competition 2011. In Proc. Formal Verification of Object-Oriented Software, ser. LNCS. Springer, 2012, vol. 7421, pp. 3–21.

[52]. Huisman M., Klebanov V., Monahan R. On the organisation of program verification competitions. In Proc. of the 1st International Workshop on Comparative Empirical Evaluation of Reasoning Systems, ser. CEUR Workshop Proceedings. RWTH Aachen University, 2012, vol. 873, pp. 50–59.

[53]. Beyer D., Huisman M., Klebanov V., Monahan R. Evaluating software verification systems: Benchmarks and competitions (Dagstuhl reports 14171). Dagstuhl Reports, vol. 4, no. 4, pp. 1–19, 2014.

[54]. Huisman M., Klebanov V., Monahan R., Tautschnig M. VerifyThis 2015. International Journal on Software Tools for Technology Transfer, vol. 19, no. 6, pp. 763–771, 2017.

[55]. Dross C., Furia C. A., Huisman M., Monahan R., Müller P. VerifyThis 2019: a program verification competition. International Journal on Software Tools for Technology Transfer, vol. 23, no. 6, pp. 883–893, 2021.

[56]. Ernst G., Huisman M., Mostowski W., Ulbrich M. VerifyThis – verification competition with a human factor. In Proc. Tools and Algorithms for the Construction and Analysis of Systems. Ser. LNCS. Springer, 2019, vol. 11429, pp. 176-195.

[57]. Huisman M., Monti R., Ulbrich M., Weigl A. The VerifyThis collaborative long term challenge. In Proc. Deductive Software Verification: Future Perspectives, ser. LNCS. Springer, 2020, vol. 12345, pp. 246 – 260.

[58]. Ernst G. Weigl A. Verify This: Memcached–a practical long-term challenge for the integration of formal methods. In Proc. iFM 2023, ser. LNCS. Springer, 2024, vol. 14300, pp. 82–89.

[59]. Bartocci E., Beyer D, Black P. E., Fedyukovich G., Garavel H., Hartmanns A., Huisman M., Kordon F., Nagele J., Sighireanu M., Steffen B., Suda M., Sutcliffe G., Weber T., Yamada A. TOOLympics 2019: An overview of competitions in formal methods. In Proc. Tools and Algorithms for the Construction and Analysis of Systems, ser. LNCS. Springer, 2019, vol. 11429, pp. 3–24.

[60]. Beyer D., Huisman M., Kordon F., Steffen B. TOOLympics II: competitions on formal methods. International Journal on Software Tools for Technology Transfer, vol. 23, no. 6, pp. 879–881, 2021.

[61]. Jasper M., Mues M., Murtovi A., Schluter M., Howar F., Steffen B., Schordan M., Hendriks D., Schiffelers R., Kuppens H., Vaandrager F. W. RERS 2019: Combining synthesis with real-world models. In Tools and Algorithms for the Construction and Analysis of Systems, ser. LNCS. Springer, 2019, vol. 11429, pp. 101–115.

[62]. Geske M., Jasper M., Steffen B., Howar F., Schordan M., van de Pol J. Rers 2016: Parallel and sequential benchmarks with focus on LTL verification. In Proc. Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications, ser. LNCS. Springer, 2016, vol. 9953, pp. 787–803.

[63]. Jasper M., Fecke M., Steffen B., Schordan M., Meijer J., van de Pol J., Howar F., Siegel S. F. The Rers 2017 challenge and workshop (invited paper). In Proc. of the 24th ACM SIGSOFT International SPIN Symposium on Model Checking of Software, 2017, pp. 11–20.

[64]. Jasper M., Mues M., Schlüter M., Steffen B, Howar F. Rers 2018: CTL, LTL, and reachability. In Proc. Leveraging Applications of Formal Methods, Verification and Validation of Systems, ser. LNCS. Springer, 2018, vol. 11245, pp. 433-447.

[65]. Howar F., Jasper M., Mues M., Schmidt D., Steffen B. The Rers challenge: towards controllable and scalable benchmark synthesis. International Journal on Software Tools for Technology Transfer, vol. 23, no. 6, pp. 917– 930, 2021.

[66]. Giesl J., Rubio A., Sternagel C., Waldmann J., Yamada A. The termination and complexity competition. In Proc. Tools and Algorithms for the Construction and Analysis of Systems, ser. LNCS. Springer, 2019, vol. 11429, pp. 156–166.

[67]. Marché C., Zantema H. The termination competition. In Proc. Term Rewriting and Applications, ser. LNCS. Springer, 2007, vol. 4533, pp. 303–313.

[68]. Giesl J, Mesnard F., Rubio A., Thiemann R., Waldmann J. Termination competition (TermCOMP 2015). In Proc. Automated Deduction - CADE-25, ser. LNCS. Springer, 2015, vol. 9195, pp. 105–108.

[69]. Yamada A. Termination of term rewriting: Foundation, formalization, implementation, and competition. In Proc. 8th International Conference on Formal Structures for Computation and Deduction (FSCD 2023), ser. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl – Leibniz-Zentrum fur Informatik, 2023, vol. 260, pp. 4:1–4:5.

[70]. Ahrendt W., Herber P., Huisman M., Ulbrich M. SpecifyThis – bridging gaps between program specification paradigms. In Proc. Leveraging Applications of Formal Methods, Verification and Validation. Verification Principles, ser. LNCS. Springer, 2022, vol. 13701, pp. 3-6.

[71]. VeHa contest, 2023. Available at: https://sites.google.com/view/veha23/, accessed Jul 06, 2024.
[72]. The First Computer Bug, 1945. Available at: https://commons.wikimedia.org/wiki/File:First_Computer_Bug,_1945.jpg, accessed Jul 06, 2024.
[73]. GitHub, Inc, Creating a pull request, 2023. Available at: https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-a-pull-request, accessed Jul 06, 2024.
[74]. VeHaContest2023 on GitHub. Available at: https://github.com/SergeyStaroletov/VeHaContest2023, accessed Jul 06, 2024.
[75]. Luna 25 mission, RBC. Available at: https://www.rbc.ru/technology_and_media/03/10/2023/651bbeab9a794768782f8d0d, accessed Jul 06, 2024.
[76]. Luna-25 mission, 2023. Available at: https://www.roscosmos.ru/39790/, accessed Jul 06, 2024.
[77]. Garanina N., Staroletov S., Gorlatch S. Model checking meets auto-tuning of high-performance programs," in International Symposium on Logic-Based Program Synthesis and Transformation. Springer, 2022, pp. 63-82.
[78]. Khronos. OpenCL Specification by Khronos Working Group, 2021. Available: https://www.khronos.org/registry/OpenCL/specs/3.0-unified/html/OpenCL_API.html, accessed Jul 06, 2024.
[79]. Tor A. GPGPU-Sim. Available at: http://gpgpu-sim.org, accessed Jul 06, 2024.
[80]. Staroletov S. Towards Modeling and Verification of Eurobalise Telegram Encoding Algorithm. Transportation Research Procedia, vol. 61, pp. 447–454, 2022.
[81]. Basics of Software Testing and Verification (in Russian). Lanbook publishing, Saint Petersburg, p. 344, 2020. – EDN SGQVLL. Available at: https://e.lanbook.com/book/319445, accessed Jul 06, 2024.
[82]. An Efficient Logic Model Checker for the Verification of threaded Code, 2023. Available at: https://github.com/nimble-code/Spin, accessed Jul 06, 2024.
[83]. select - non-deterministic value selection, 2021. Available at: http://spinroot.com/spin/Man/select.html, accessed Jul 06, 2024.

Информация об авторах / Information about authors

Сергей Михайлович СТАРОЛЕТОВ – кандидат физико-математических наук, доцент (ВАК). Сфера научных интересов: формальная верификация, проверка моделей, киберфизические системы, операционные системы.

Sergey Mikhailovich STAROLETOV – Cand. Sci. (Phys.-Math.), associate professor. Research interests: formal verification, model checking, cyber-physical systems, operating systems.

Дмитрий Александрович КОНДРАТЬЕВ – кандидат физико-математических наук, научный сотрудник ИСИ СО РАН, старший преподаватель НГУ. Сфера научных интересов: формальная верификация, дедуктивная верификация, логика Хоара и автоматическое доказательство теорем.

Dmitry Alexandrovich KONDRATYEV – Cand. Sci. (Phys.-Math.), researcher at Ershov Institute of Informatics Systems Siberian Branch of the RAS, senior lecturer at Novosibirsk State University. Research interests: formal verification, deductive verification, Hoare logic and automated theorem proving.

Наталья Олеговна ГАРАНИНА – кандидат физико-математических наук, старший научный сотрудник ИСИ СО РАН, доцент НГУ. Сфера научных интересов: формальная верификация, проверка моделей, распределенные системы, инженерия требований.

Natalia Olegovna GARANINA – Cand. Sci. (Phys.-Math.), senior researcher at Ershov Institute of Informatics Systems Siberian Branch of the RAS, associate professor at Novosibirsk State University. Research interests: formal verification, model checking, distributed systems, requirements engineering.

Ирина Владимировна ШОШМИНА – кандидат технических наук, доцент. Сфера научных интересов: формальная верификация, распределенные системы и алгоритмы.

Irina Vladimirovna SHOSHMINA – Cand. Sci. (Tech.), associate professor. Research interests: formal verification, model checking, distributed systems and algorithms.

DOI: 10.15514/ISPRAS-2024-36(2)-12

Исследование электровихревого течения между плоскостями с помощью различных вычислительных подходов

[1,2] *Е.А. Михайлов*, ORCID: 0000-0002-9747-4039 <*ea.mikhajlov@physics.msu.ru*>
[2,3] *А.А. Таранюк*, ORCID: 0000-0002-7836-8468 <*taranyuk.anton@gmail.com*>
[2] *А.П. Степанова*, ORCID: 0000-0002-6557-9901 <*nastasya_stepanova@mail.ru*>

[1] *Физический институт имени П.Н. Лебедева РАН,*
119991, Россия, г. Москва, Ленинский проспект, д. 53.
[2] *Московский государственный университет имени М.В. Ломоносова,*
119991, Россия, Москва, Ленинские горы, д. 1.
[3] *Институт теории прогноза землетрясений и математической геофизики РАН*
117997, Россия, Москва, улица Профсоюзная, 84/32.

Аннотация. Электровихревые течения возникают при прохождении электрического тока меняющейся плотности через хорошо проводящую жидкость (например, кислоту или расплав металла). В таком случае ток порождает магнитное поле, которое приводит к возникновению электромагнитной силы, вызывающей закрученные течения среды. Существуют разные методы теоретического исследования подобных течений. Как правило, чтобы избежать необходимости находить зависимость давления от координат, используются переменные «векторный потенциал скорости – завихренность» («скалярная функция тока – завихренность» в случае осесимметричных течений). В таком случае достаточно эффективно использование автомодельных переменных, позволяющих понизить размерность задачи. Это дает возможность искать решение для введенной функции в виде разложения по параметру электровихревого течения, пропорционального квадрату магнитного числа Рейнольдса. Также данное решение может быть получено численно, например с помощью конечно-разностных методов. В настоящее время все чаще решения исследуются методами прямого численного моделирования, когда не делается автомодельных приближений, снижающих точность решения. Тем не менее, в таком случае объем вычислений может оказаться достаточно большим и требует использования суперкомпьютерных ресурсов. Отдельную сложность представляют граничные условия: так, для векторного потенциала скорости получается уравнение четвертого порядка, что накладывает существенные ограничения на шаги по времени в эволюционном уравнении. Проблемы можно избежать, используя приближенные граничные условия, однако это вновь снижает точность решения. В настоящей работе на примере электровихревого течения между плоскостями рассмотрены решения, которые можно получить с использованием различных вычислительных подходов, указанных выше. Проводится сравнение полученных результатов, также они сравниваются с аналитическими приближениями.

Ключевые слова: электровихревые течения; автомодельные переменные; приближенные решения; конечно-разностные методы; скалярная функция тока.

Для цитирования: Михайлов Е.А., Таранюк А.А., Степанова А.П. Исследование электровихревого течения между плоскостями с помощью различных вычислительных подходов. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 169–180. DOI: 10.15514/ISPRAS–2024–36(2)–12.

Благодарности: Работа выполнена с использованием оборудования Центра коллективного пользования сверхпроизводительными вычислительными ресурсами МГУ им. М. В. Ломоносова.

Investigation of Electro-Vortex Flow between Planes using Different Computational Approaches

[1,2] *E.A. Mikhailov*, ORCID: *0000-0002-9747-4039<ea.mikhajlov@physics.msu.ru>*
[2,3] *A.A. Taranyuk*, ORCID: *0000-0002-7836-8468<taranyuk.anton@gmail.com>*
[2] *A.P. Stepanova*, ORCID: *0000-0002-6557-9901<nastasya_stepanova@mail.ru>*
[1] *P.N. Lebedev Physical Institute of the Russian Academy of Sciences, 53, Leninsky Prospekt, Moscow, 119991, Russia.*
[2] *Lomonosov Moscow State University, 1, Leninskie Gory, Moscow, 119991, Russia.*
[3] *Institute of Earthquake Prediction Theory and Mathematical Geophysics of the Russian academy of sciences, 84/32, Profsoyuznaya str, Moscow, 117997, Russia.*

Abstract. Electrovortex flows arise when an electric current of varying density passes through a well-conducting fluid (e.g., acid or metal melt). In such a case, the electric current generates a magnetic field, which leads to the Lorentz force causing swirling currents of the medium. There are different methods of theoretical study of such currents. As a rule, to avoid the necessity to find the pressure dependence on coordinates, the variables "vector potential of velocity - swirl" ("scalar current function – swirl" in the case of axisymmetric flows) are used. In such a case, it is quite effective to use automodel variables, which allow to reduce the dimensionality of the problem. In this case, the solution for the introduced function can be sought in the form of an expansion by the electro-vortex flow parameter proportional to the square of the magnetic Reynolds number. Also, this solution can be obtained numerically, for example, using finite-difference methods. Nowadays, more and more often the solutions are investigated by means of direct numerical modeling methods, when no automodel approximations are made, which reduce the accuracy of the solution. Nevertheless, in such a case the number of computations can be quite large and requires the use of supercomputer resources. A separate difficulty is presented by the boundary conditions: for example, for the velocity vector potential we obtain a fourth-order equation, which imposes significant restrictions on the time steps in the evolution equation. The problem can be avoided by using approximate boundary conditions, but this again reduces the accuracy of the solution. In this paper, using the example of electro-vortex flow between planes, the solutions that can be obtained using the various computational approaches mentioned above are examined. The results obtained are compared, and they are also compared with analytical approximations.

Keywords: electro-vortex flows; automodel variables; approximate solutions; finite-difference methods; scalar current function.

For citation: Mikhailov E.A., Taranyuk A.A., Stepanova A.P. Investigation of electro-vortex flow between planes using different computational approaches. Proceedings of ISP RAS, vol. 36, issue 2, 2024, pp. 169-180 (in Russian). DOI: 10.15514/ISPRAS–2024–36(2)–12.

Acknowledgements. The work was performed using the equipment of the Center for Collective Use of Ultra-Performance Computing Resources of Lomonosov Moscow State University.

1. Введение

Электровихревые течения представляют большой интерес как с точки зрения теоретической гидродинамики, так и с точки зрения технических приложений. Они возникают при прохождении электрического тока переменной плотности через хорошо проводящие жидкости [1]. К ним относятся растворы кислот и их солей, расплавы различных металлов. В таком случае неоднородный ток порождает магнитное поле. При взаимодействии тока с магнитным полем возникает сила Лоренца, которая приводит к появлению закрученных течений среды.

Существует большое количество технологических процессов, для которых электровихревые течения играют большую роль. Так, в настоящее время активно развиваются методы электродугового переплава металлов. При прохождении электрического тока через расплав в нем возникают закрученные течения [2]. Они могут играть как положительную роль

(способствуя активному перемешиванию среды), так и отрицательную – в плавильных печах могут возникать потоки, которые приводят к их преждевременному износу. Не меньшее значение электровихревые течения играют при создании аккумуляторных батарей. В процессе зарядки в электролите также возникают закрученные движения, которые могут представлять определенную проблему. Ситуация становится особенно сложной в случае быстрой зарядки батареи, которая оказывается необходимой с точки зрения удобства использования устройств. Все это говорит о том, что электровихревые течения требуют детального исследования.

Первые работы, посвященные изучению подобных явлений, относятся к 1970-м годам. Необходимо упомянуть исследования, проводящиеся в латвийском Институте физики (Саласпилс), которые были связаны как с лабораторными экспериментами, так и с теоретическими моделями [3]. В настоящее время значимые результаты накоплены также в Институте механики сплошных сред Уральского отделения РАН (Пермь) [4]. Большое количество прикладных исследований проводится в Магнитогорском государственном техническом университете. Важные эксперименты проводятся на базе Объединенного института высоких температур РАН (Москва) [5]. Если говорить об исследованиях в зарубежных странах, то нельзя не упомянуть значимые теоретические работы, проведенные в Университете Шеффилда (Великобритания) [6] и Университете Леобена (Австрия) [7].

Первые теоретические исследования, связанные с электровихревыми течениями, были связаны с использованием различных упрощенных моделей. Они относились к построению автомодельных решений, которые могли быть найдены с помощью тех или иных приближений. Между тем, по мере развития вычислительной техники стало очевидно, что гораздо более эффективно решать подобные задачи численно [8]. С одной стороны, возможно решать уравнения, полученные в рамках автомодельных приближений, с другой – решать исходную («неупрощенную») задачу с помощью прямого численного моделирования. Как правило, задача решается с использованием переменных «векторный потенциал – завихренность», чтобы избежать необходимости определения поля давления. В случае, если задача характеризуется определенной осевой симметрией, данный подход сводится к использованию переменных «скалярная функция тока – завихренность».

Между тем, подобные методы сталкиваются с некоторыми существенными трудностями. Так, в случае применения прямого численного моделирования требуется использование достаточно подробных сеток. Ситуация усугубляется тем, что, как правило, электровихревые течения характеризуются большими градиентами. Это требует использования суперкомпьютерной техники, что предусматривает подготовку кодов для параллельных вычислений, и затрудняет применение неявных конечно-разностных методов. Поскольку уравнения для векторного потенциала скорости (или скалярной функции тока) имеют четвертый порядок по пространству, это накладывает исключительно жесткие требования на шаг по времени.

Возможным выходом из ситуации является применение приближенных условий, например широко известных в гидродинамике условий Тома [9]. Тем не менее, они являются нелокальными, и порождают дополнительные сложности для решения. Кроме того, данные условия – приближенные, поэтому в данном случае мы получаем дополнительный источник возможных ошибок. Все это требует детального анализа результатов, полученных разными способами, и их сравнения.

В настоящей работе рассмотрен достаточно интересный и важный с точки зрения технических приложений пример электровихревого течения между двумя плоскостями [10-11]. Электрический ток растекается от двух плоских электродов положительной полярности к удаленному электроду отрицательной полярности, который охватывает область на довольно большом расстоянии (рис. 1). Ранее для данной задачи было получено уравнение в автомодельных переменных, предполагающий бесконечным расстояние до электрода отрицательной полярности [12-13]. Его решение можно получить с помощью итерационного

метода ван Дайка, раскладывая скалярную функцию тока по степеням параметра электровихревого течения. (Его величина по порядку величины равна квадрату магнитного числа Рейнольдса). В предшествующих работах мы получили приближения вплоть до третьего порядка включительно.

В настоящей работе мы решаем задачу об электровихревом течении между плоскостями (рис.1), используя различные вычислительные модели. Сначала мы решаем задачу численно в автомодельной постановке, пользуясь как точными граничными условиями (тогда возникает необходимость решения уравнения четвертого порядка для скалярной функции тока), так и приближенными, когда исследование сводится к необходимости решения пары уравнений второго порядка – для скалярной функции тока и для завихренности течения. Наконец, решается исходная физическая задача. Результаты решения, полученные разными методами, сравниваются как друг с другом, так и с приближенной асимптотической моделью, результаты использования которой для данного течения также приводятся в статье.

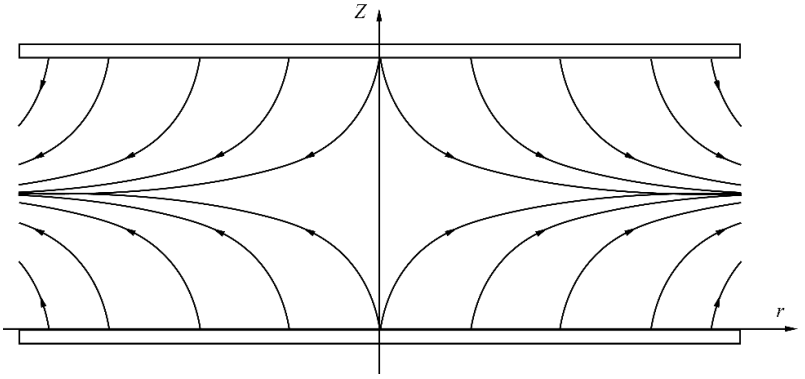


Рис. 1. Схематичное изображение линий электрического тока
Fig. 1. Schematic representation of electric current lines

2. Основные уравнения

Рассмотрим течение между двумя плоскостями. Его скорость $\vec{\upsilon}$ описывается с помощью уравнения Навье-Стокса:

$$\frac{\partial \vec{\upsilon}}{\partial t}+\left(\vec{\upsilon}\cdot\vec{\nabla}\right)\vec{\upsilon}=-\frac{1}{\rho}\vec{\nabla}p+\frac{1}{\rho}\left[\vec{j}\times\vec{B}\right]+\nu\Delta\vec{\upsilon};$$
$$\vec{\upsilon}\Big|_{\Gamma}=0 \qquad (1)$$

где p – давление, ρ – плотность среды, $\vec{j}$ – плотность электрического тока, $\vec{B}$ – индукция магнитного поля, ν – кинематическая вязкость среды.

Одной из сложностей при решении подобных уравнений является необходимость вычислять поле давления. С учетом того, что жидкость с большой степенью точности можно считать несжимаемой, поле скорости удовлетворяет условию $\nabla\cdot\vec{\upsilon}=0$. По этой причине скорость можно представить в виде $\vec{\upsilon}=\vec{\nabla}\times\vec{\psi}$, где $\vec{\psi}$ – так называемый векторный потенциал скорости (он имеет примерно тот же смысл, что и векторный потенциал для магнитного поля, которое также является соленоидальным). Другой важной характеристикой течения является так называемая завихренность, определяемая как ротор скорости течения $\vec{\omega}=\vec{\nabla}\times\vec{\upsilon}$.

Учитывая осевую симметрию задачи, можно отметить, что в отсутствие внешних магнитных полей движение среды будет полоидальным ($\vec{\upsilon}=\upsilon_r\vec{e}_r+\upsilon_z\vec{e}_z$), поэтому векторный потенциал скорости будет иметь лишь одну ненулевую компоненту и может быть представлен с

помощью так называемой скалярной функции тока: $\vec{\psi} = \psi \vec{e}_{\varphi}$. То же самое можно сказать и про завихренность, которая в данном случае также будет описываться с помощью скалярной функции: $\vec{\omega} = \omega \vec{e}_{\varphi}$.

Уравнение для ее эволюции может быть получено путем взятия операции ротора от обеих частей уравнения Навье-Стокса:

$$\frac{\partial \omega}{\partial t} - \frac{\partial \psi}{\partial z}\frac{\partial \omega}{\partial r} + \frac{\partial \psi}{\partial r}\frac{\partial \omega}{\partial z} = \left(\vec{\nabla} \times \left[\vec{j} \times \vec{B}\right]\right)_{\varphi} + \nu\left(\Delta\omega - \frac{\omega}{r^2}\right). \quad (2)$$

Нас будет интересовать стационарное решение, не зависящее от времени. Кроме того, данное уравнение удобно переписать в безразмерном виде. Будем измерять длины в единицах расстояния между пластинами. Также введем параметр электровихревого течения $S = \frac{\mu_0 I^2}{\rho \nu^2}$

Тогда первое из уравнений можно записать в виде [12]:

$$-\frac{\partial \psi}{\partial z}\frac{\partial \omega}{\partial r} + \frac{\partial \psi}{\partial r}\frac{\partial \omega}{\partial z} = Sr^2\left(z - \frac{1}{2}\right) + \Delta\omega - \frac{\omega}{r^2}. \quad (3)$$

Кроме того, необходимо решать уравнение, позволяющее связать завихренность и скалярную функцию тока:

$$\Delta\psi - \frac{\psi}{r^2} = -\omega. \quad (4)$$

В качестве граничных условий в случае прилипания к поверхности следует выбрать следующее:

$$\psi\big|_{z=0} = \psi\big|_{z=1} = \psi\big|_{r=R} = \frac{\partial \psi}{\partial z}\bigg|_{z=0} = \frac{\partial \psi}{\partial z}\bigg|_{z=1} = \frac{\partial \psi}{\partial r}\bigg|_{r=R} = 0. \quad (5)$$

Данную систему уравнений можно решать различными способами, как аналитическими, так и численными.

3. Автомодельная постановка задачи

Как для аналитических, так и для численных расчетов удобно пользоваться автомодельным приближением [13]. Если полагать, что $R \to \infty$, то можно ввести вспомогательную функцию *F(z)*, посредством которой скалярная функция тока будет выражаться с помощью выражения [12]:

$$\psi(r, z) = r^2 F(z). \quad (6)$$

Тогда для нее можно сформулировать уравнение:

$$F\frac{d^3F}{dz^3} = \frac{d^4F}{dz^4} + S\left(z - \frac{1}{2}\right) \quad (7)$$

с граничными условиями:

$$F\big|_{z=0} = F\big|_{z=1} = \frac{dF}{dz}\bigg|_{z=0} = \frac{dF}{dz}\bigg|_{z=1} = 0. \quad (8)$$

3.1 Асимптотическое разложение решения

Для решения поставленной задачи мы разложим функцию *F* по степеням параметра электровихревого течения *S*. Разложение тогда будет выглядеть следующим образом [12]:

$$F(z) = S^1 F_1(z) + \ldots + S^n F_n(z) + \ldots \quad (9)$$

В зависимости от требований к точности решения мы можем далее решать задачу с разным количеством слагаемых в представленном выше разложении. Для поиска решений будем пользоваться методом последовательных приближений: сначала найдём функцию F_1, учитывая в разложении только слагаемые, пропорциональные параметру электровихревых течений в первой степени, а затем будем вычислять каждую последующую через предыдущие. Для дальнейшего нахождения каждого из приближений подставим общее разложение в полученное выше стационарное уравнение:

$$S\frac{d^4F_1}{dz^4}+S^2\frac{d^4F_2}{dz^4}-\left(SF_1(z)+S^2F_2(z)\right)\cdot\left(S\frac{d^3F_1}{dz^3}+S^2\frac{d^3F_2}{dz^3}\right)+S\left(z-\frac{1}{2}\right)=0 \tag{10}$$

Итак, найдём первое приближение по описанному нами алгоритму:

$$\frac{d^4F_1}{dz^4}+z-\frac{1}{2}=0 \tag{11}$$

Граничные условия для данного уравнения нам уже знакомы:

$$F_1\big|_{z=0}=F_1\big|_{z=1}=F_1^{'}\big|_{z=0}=F_1^{'}\big|_{z=1}=0 \tag{12}$$

Теперь мы решаем полученное уравнение четырёхкратным интегрированием с поиском констант интегрирования из поставленных граничных условий. Тогда получим:

$$F_1(z)=-\frac{1}{120}(z^5-\frac{5}{2}z^4+2z^3-\frac{1}{2}z^2) \tag{13}$$

Полученное первое приближение применимо для значений параметра электровихревого течения $S < 10^2$. В связи с этим возникает необходимость найти второе приближение. Продолжая аналогично данные действия, мы можем получить уравнение для второго приближения, которое выглядит следующим образом (граничные условия будут аналогичны):

$$\frac{d^4F_2}{dz^4}=F_1\frac{d^3F_1}{dz^3} \tag{14}$$

Его решение будет таким [12]:

$$F_2(z)=\frac{1}{199584000}z^2(z-1)^2(z-\frac{1}{2})(105z^6\text{-}315z^5+295z^4\text{-}65z^3\text{-}19z^2\text{-}z+3) \tag{15}$$

Мы нашли первые два приближения решения поставленной задачи. Но трудность состоит в том, что для больших значений параметра электровихревого течения S этого вновь оказывается недостаточно. Поэтому возникает необходимость поиска третьего приближения. Для нахождения соответствующей функции получим уравнение:

$$\frac{d^4F_3}{dz^4}=(F_1(z)\frac{d^3}{dz^3}F_2(z)+F_2(z)\frac{d^3}{dz^3}F_1(z)) \tag{16}$$

Пошагово интегрируя обе части этого уравнения четыре раза и находя константы интегрирования с помощью описанных выше граничных условий, получим [13]:

$$F_3(z)=\frac{-4299750}{53353114214400000}z^2(z-1)^2\left(z-\frac{1}{2}\right)\left(\frac{2124643}{143325}z^{10}+\frac{1898801}{143325}z^8+\frac{141433}{716625}z^6+\frac{169082}{2149875}z^4+\frac{46556}{716625}z^5+\right.$$
$$\left.+z^{12}-6z^{11}-\frac{128}{716625}-\frac{3116}{2149875}z-\frac{10544}{2149875}z^2-\frac{27364}{2149875}z^3-\frac{612614}{143325}z^7-\frac{548068}{28665}z^9\right) \tag{17}$$

Таким образом, мы получили третье приближение, которое дает вполне приемлемую точность при $S < 10^5$. Такой параметр электровихревых течений соответствует уже довольно высоким токам и в связи с эти удовлетворяет требованиям многих экспериментов.

Полученные аналитически результаты являются хорошим материалом для сравнения с численным решением автомодельной задачи. Также можно предполагать качественное соответствие с более полной задачей, где не используются автомодельные подходы.

3.2 Численное решение автомодельной задачи

Как можно видеть из сказанного выше, решение задачи о возникновении электровихревого течения с помощью асимптотических разложений оказывается хотя и вполне разрешимой задачей, но полученные решения оказываются крайне громоздкими, а вычисление конкретных значений скалярной функции тока по данным формулам само по себе требует значительных вычислительных ресурсов.

По причинам, обозначенным выше, гораздо более применимыми выглядят численные подходы, связанные с решением возникающих уравнений конечно-разностными методами. Для этого можно использовать метод счета на установление, который, по сути, будет являться возвратом к исходной нестационарной задаче [15,16]. Тем не менее, нельзя не отметить следующую сложность. Так, уравнение для скалярной функции тока (а также для функции F) имеют четвертый порядок по координате, и его решение сопряжено с определенными проблемами. В случае использования явных схем (а в случае неявных, кроме определенной сложности в реализации, оказывается затруднено их распараллеливание) шаг по времени оказывается пропорционален четвертой степени шага по координате. Это требует огромных вычислительных ресурсов. По этой причине можно воспользоваться вспомогательной функцией, смысл которой перекликается с завихренностью [13]:

$$W = -\frac{\partial^2 F}{\partial z^2}. \tag{18}$$

Обсудим, как можно поставить граничные условия. В окрестностях границы (например, $z = 0$) значение функции может быть выражено так:

$$F\big|_{z=\Delta z} \cong F\big|_{z=0} + \left.\frac{\partial F}{\partial z}\right|_{z=0} \Delta z + \frac{1}{2}\left.\frac{\partial^2 F}{\partial z^2}\right|_{z=0} \Delta z^2 = -\frac{1}{2} W\big|_{z=0} \Delta z^2. \tag{19}$$

Тогда мы можем поставить так называемое нелокальное граничное условие Тома [6, 17] (в качестве Δz можно использовать шаг разностной схемы):

$$F\big|_{z=\Delta z} = -\frac{1}{2} W\big|_{z=0} \Delta z^2. \tag{20}$$

Как и говорилось выше, задача может быть решена с помощью метода счета на установление. Тогда мы приходим к системе следующей системе уравнений:

$$\frac{\partial W}{\partial t} = F\frac{\partial W}{\partial z} + \frac{\partial^2 W}{\partial z^2} - S\left(z - \frac{1}{2}\right); \tag{21}$$

$$\frac{\partial F}{\partial t} = W + \frac{\partial^2 F}{\partial z^2}. \tag{22}$$

$$F\big|_{z=0} = F\big|_{z=1} = F\big|_{z=\Delta z} + \frac{1}{2} W\big|_{z=0} \Delta z^2 = F\big|_{z=1-\Delta z} + \frac{1}{2} W\big|_{z=1} \Delta z^2 = 0. \tag{23}$$

Подобная задача решалась нами с помощью собственной компьютерной программы, представленной в одном из репозиториев для свободного использования [19]. Отметим, что алгоритм основан на использовании явной численной схемы с применением метода счёта на установление. Это обеспечивает достаточную точность и позволяет решать задачу довольно быстро даже на отдельном процессоре (учитывая ее простоту). В таком случае количество операций имеет порядок $O(N \cdot M)$, где N – количество узлов сетки по оси Z, M – количество узлов по оси времени. Учитывая условия устойчивости разностной схемы, $M=O(N^2)$, поэтому в качестве оценки для числа операций мы получаем величину $O(N^3)$. В табл. 1 приведены данные о времени, затраченном на расчет при использовании персонального компьютера с процессором Intel Core i5-5250U (столбец $N = 100$).

Между тем, возникает вопрос о том, насколько численное решение, полученное таким способом, соответствует задаче. Для этого с помощью метода счета на установление нами решалась исходная автомодельная задача (7) – (8) для функции F. Как говорилось выше, объем вычислений оказывается довольно значительным, поэтому это потребовало использования параллельных вычислений на суперкомпьютере [18]. Код для вычислений также представлен в репозитории для свободного использования [19]. Для данной задачи также использовалась явная численная схема, количество операций для которой имеет порядок $O(N \cdot M)$, где N – количество узлов сетки по оси Z, M – количество узлов по оси времени. Здесь условие устойчивости приводит к соотношению $M=O(N^4)$, поэтому для общего числа операций имеем величину $O(N^5)$. Хотя это и довольно большое значение, недостатки данной схемы отчасти компенсируются весьма несложным алгоритмом распараллеливания. В табл. 1 приведены данные о времени, затраченном на расчет при использовании суперкомпьютера «Ломоносов-2» (столбец N = 1000).

Табл. 1. Время счёта
Table 1. Counting time

Задача	**N=100**	**N=1000**
Автомодельная задача с использованием условий Тома	37 с (на одном ядре)	163 с (на одном ядре)
Автомодельная задача 4-ого порядка	7148 с (на 25 ядрах)	73851 с (на 250 ядрах)
Двумерная задача	8795 с (на 25 ядрах)	93186 с (на 250 ядрах)

Результат сравнения результатов, полученных с помощью различных вычислительных подходов, показан на рис. 2. Также там показано асимптотическое решение, полученное аналитически. Можно видеть, что все способы получения решения демонстрируют сходство. Если говорить о двух численных подходах (черная и красная кривые), то они практически неразличимы даже для использованных в расчете грубых сетках. Ввиду того, что вычислительная сложность алгоритма, использующего приближенные условия Тома, на два порядка меньше, это означает, что в приложениях есть смысл использовать именно его. Что касается аналитического выражения, то оно дает гораздо большую ошибку и должно использоваться лишь для грубых оценок.

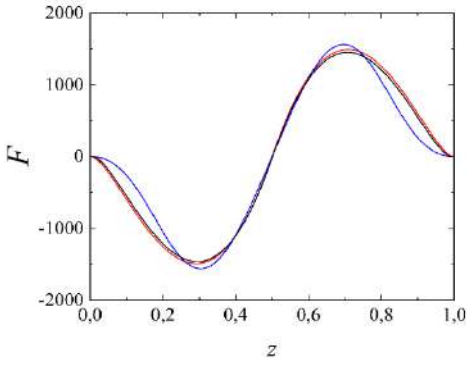


Рис. 2. Зависимость F(z), полученная различными подходами: черная линия – решение дифференциального уравнения 4-ого порядка, красная – с использованием граничных условий Тома, синяя – аналитическое приближение

Fig. 2. The dependence F(z) obtained by various approaches: the black line is the solution of a differential equation of the 4th order, the red one is using the boundary conditions of the Volume, the blue one is an analytical approximation

4. Решение двумерной задачи

Вместе с тем, кроме решения автомодельных задач, оказывается достаточно интересным численное решение исходной двумерной задачи для нахождения скалярной функции тока. Тогда в стационарном случае необходимо решить следующую систему уравнений:

$$\frac{\partial\psi}{\partial r}\frac{\partial\omega}{\partial z}-\frac{\partial\psi}{\partial z}\frac{\partial\omega}{\partial r}=\frac{\partial^2\omega}{\partial z^2}+\frac{\partial^2\omega}{\partial r^2}+\frac{1}{r}\frac{\partial\omega}{\partial r}-\frac{\omega}{r^2}+Sr^2\left(\frac{1}{2}-z\right); \tag{24}$$

$$\frac{\partial^2\psi}{\partial z^2}+\frac{\partial^2\psi}{\partial r^2}+\frac{1}{r}\frac{\partial\psi}{\partial r}-\frac{\psi}{r^2}+\omega=0. \tag{25}$$

Данную задачу мы решали численно с помощью счета на установление, поставив такую задачу:

$$\frac{\partial\psi}{\partial t}=-\frac{\partial\psi}{\partial r}\frac{\partial\omega}{\partial z}+\frac{\partial\psi}{\partial z}\frac{\partial\omega}{\partial r}+\frac{\partial^2\omega}{\partial z^2}+\frac{\partial^2\omega}{\partial r^2}+\frac{1}{r}\frac{\partial\omega}{\partial r}-\frac{\omega}{r^2}+Sr^2\left(\frac{1}{2}-z\right); \tag{26}$$

$$\frac{\partial\omega}{\partial t}=\frac{\partial^2\psi}{\partial z^2}+\frac{\partial^2\psi}{\partial r^2}+\frac{1}{r}\frac{\partial\psi}{\partial r}-\frac{\psi}{r^2}+\omega; \tag{27}$$

$$\psi\big|_{z=0}=\psi\big|_{z=1}=\psi\big|_{r=R}=\left.\frac{\partial\psi}{\partial r}\right|_{r=0}=0; \tag{28}$$

$$\psi\big|_{z=\Delta z}+\frac{1}{2}\omega\big|_{z=0}\Delta z^2=\psi\big|_{z=1-\Delta z}+\frac{1}{2}\omega\big|_{z=1}\Delta z^2=\psi\big|_{r=R-\Delta r}+\frac{1}{2}\omega\big|_{r=R}\Delta r^2=0. \tag{29}$$

Для решения задачи была составлена программа, доступная в репозитории для свободного использования. Аналогично предыдущим алгоритмам, данный также использует явную численную схему, число операций для которой имеет порядок $O(N^2 \cdot M)$, где N – количество узлов сетки по координатным осям R и Z, M – количество узлов по оси времени. Отметим, что условие устойчивости позволяет записать соотношение $M=O(N^2)$. Тогда количество операций будет порядка $O(N^4)$. С учетом использования параллельных вычислений, данная задача считается в течение удовлетворительного времени. В табл. 1 приведены данные о времени, затраченном на расчет при использовании суперкомпьютера «Ломоносов-2». Результаты при R=10 представлены на рис. 3. Было бы интересно сравнить ее с результатами автомодельной задачи. Нельзя не отметить, что полной аналогии между ними быть не может (поскольку одна из задач ставится в бесконечной области, а другая – в ограниченной. Тем не менее, во внутренних областях на большом расстоянии от внешней границы подобное сравнение вполне допустимо, и можно ожидать, что разные подходы будут давать качественно схожие результаты. Действительно, рис. 4 показывает, что около оси z скалярные функции тока близки друг к друга, но при удалении от нее различие начинает нарастать. Экстремумы функции тока в полной задаче смещаются к верхнему и нижнему электроду.

5. Заключение

В представленной работе мы рассмотрели электровихревое течение между двумя плоскостями. Для него было построено автомодельное решение с применением условий Тома, с использованием исходных граничных условий, а также с приближенным аналитическим решением. Было показано, что условия Тома дают вполне приемлемую точность, позволяя существенно сэкономить вычислительные ресурсы. Это означает, что в реальных прикладных задачах имеет смысл использовать данную приближенную постановку, которая практически не уступают по аккуратности даваемых результатов, однако являются намного более экономной в плане вычислительных затрат.

Также автомодельное решение сравнивалось с данными прямого численного решения задачи для скалярной функции тока. Можно отметить, что на небольшом расстоянии от центра оно дает приемлемые результаты, однако по мере удаления различие начинает возрастать.

В данных задачах особое значение имеет производительность вычислений и затраченное на них время. Сравнение затраченного времени для разных задач и вычислительных инструментов приведено в табл.1. Можно видеть, что время расчета для выбранных параметров остается вполне приемлемым, хотя для дальнейшего развития задачи имело бы смысл модифицировать численные схемы, в частности – использовать схемы переменных направлений.

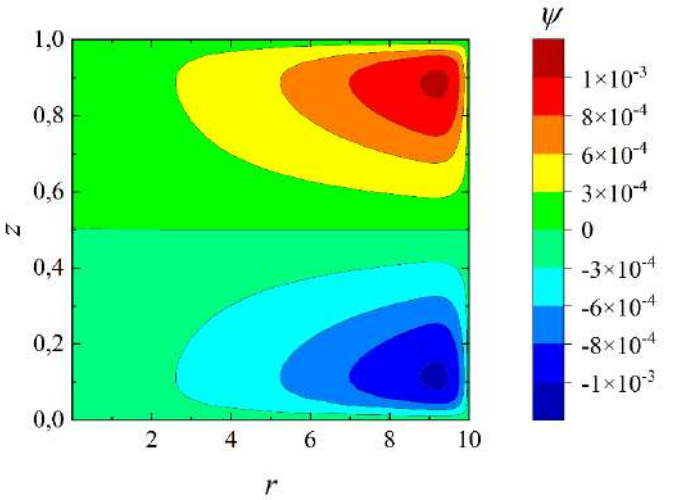


Рис. 3. Линии уровня функции ψ (r, s) для неавтомодельного решения
Fig. 3. Lines of the function ψ (r, s) level for a non-automatic solution

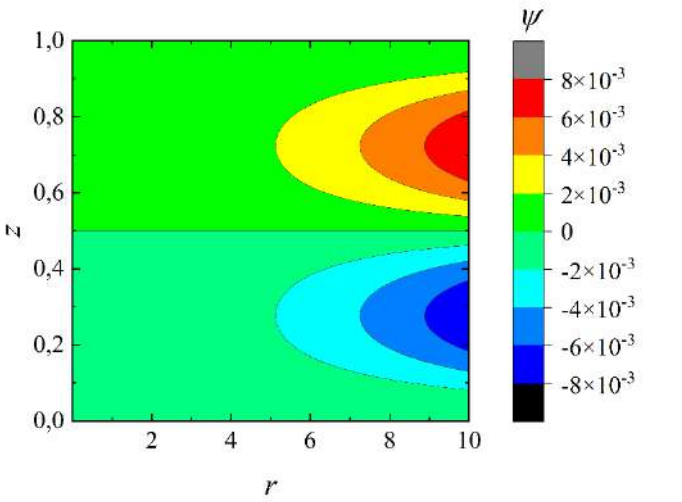


Рис. 4. Линии уровня функции ψ (r, s) для автомодельного решения
Fig. 4. Lines of the ψ (r, s) function level for a self-similar solution

Список литературы / References

[1]. Гельфгат Ю.М., Лиелаусис О.А., Щербинин Э.В. Жидкий металл под действием электромагнитных сил. – Рига: Зинатне, 1975

[2]. Pavlovs S., Jakovics A., Baake E., Nacke B. Melt flow patterns in metallurgical MHD devices with combined inductive and conductive power supply, Magnetohydrodynamics, 50 (2014), 303–315

[3]. Бояревич В.В., Фрейберг Я.Ж., Шилова Е.И., Щербинин Э.В. Электровихревые течения. \\ Рига: Зинатне, 1985.

[4]. Мандрыкин, С.Д., Колесниченко, И.В., Лосев, Г.Л., Фрик, П.Г. Электровихревое течение жидкого металла в цилиндрическом канале / С.Д. Мандрыкин, И.В. Колесниченко, Г.Л. Лосев, П.Г. Фрик. // Вестник Пермского университета. Физика. – Вып.2 (40). – С.20 – 26.

[5]. Ивочкин, Ю.П., Тепляков, И.О., Гусева, А.А., Токарев, Ю.Н. Численное и экспериментальное исследование структуры закрученного электровихревого течения / Ю.П. Ивочкин, И.О. Тепляков, А.А. Гусева, Ю.Н. Токарев // Тепловые процессы в технике, 2012. – 8. – С.345 – 352.

[6]. Sozou, C., Pickering, W.M. Magnetohydrodynamic flow in a container due to the discharge of an electric current in a hemispherical container / C.Sozou, W.M.Pickering // Journal of Fluid Mechanics, 1976. – V.73. – P.641 – 650

[7]. Kharicha, A., Karimi-Sibaki, E., Wu, M., Ludwig, A. Bohacek Review on Modeling and Simulation of Electroslag Remelting / A.Kharicha, E.Karimi-Sibaki, M.Wu, A.Ludwig, J.: Steel Res. Int., 2018. – V.89 - 1700100.

[8]. Liu, K. Numerical and experimental investigation of electro-vortex flow in a cylindrical container / K. Liu, F. Stefani, N. Weber, T. Weier, B.W. Li // Magnetohydrodynamics. 2020. Vol. 56. No. 1.

[9]. Weinan, E., Liu, J.G. Vorticity Boundary Condition and Related Issues for Finite Difference Schemes / E. Weinan, J.G. Liu // Journal of Computational Physics, 1996. – V.124.

[10]. Kaudze M., Chudnovsky A. Axisymmetric electrovortex flow between two planes induced by AC, Magnetohydrodynamics, 25 (1989), 187–194 Zentralblatt MATH.

[11]. Bojarevich V., Saramkin V. MHD flows due to electrical current discharge in an axisymmetric layer of limited depth, Magnetohydrodynamics, 13 (1977), 172–177.

[12]. Михайлов, Е.А., Чудновский, А.Ю. Асимптотическое разложение решения уравнения для медленного осесимметричного электровихревого течения между двумя плоскостями / Михайлов Е.А., Чудновский А.Ю. // Сибирский журнал индустриальной математики, 2020. – Т.23. – С.88 – 100.

[13]. Михайлов Е.А., Степанова А.П., Таранюк А.А. Анализ и модель системы электровихревых течений между двумя плоскостями при больших токах \\ Труды НГТУ им. Р.Е. Алексеева, 2022, 1. С.32-42.

[14]. Смирнов Е.М. Автомодельные решения уравнений Навье-Стокса для закрученного течения несжимаемой жидкости в круглой трубе // Прикладная математика и механика, 1981, 45, с.833 – 839

[15]. Калиткин Н.Н. Численные методы. М., Наука, 1978.

[16]. Калиткин Н.Н., Белов А.А. Аналог метода Ричардсона для логарифмически сходящегося счета на установление // Доклады Академии наук, 2013, 452, с.261 – 265.

[17]. Mikhailov E.A., Teplyakov I.O. Construction asymptotic solution while studying electrovortex flow in hemispherical container usung Stokes approximation // Journal of Physics: Conference Series, 2017, 891, 012060.

[18]. Vl. Voevodin, A. Antonov, D. Nikitenko, P. Shvets, S. Sobolev, I. Sidorov, K. Stefanov, Vad. Voevodin, S. Zhumatiy: Supercomputer Lomonosov-2: Large Scale, Deep Monitoring and Fine Analytics for the User Community. In Journal: Supercomputing Frontiers and Innovations, Vol.6, No.2 (2019). pp.4–11. DOI:10.14529/jsfi190201

[19]. https://github.com/Azilrib/EVT_Planes.

Информация об авторах / Information about authors

Евгений Александрович МИХАЙЛОВ – доктор физико-математических наук, старший научный сотрудник Физического института имени П.Н. Лебедева с 2021 года, доцент физического факультета МГУ имени М.В. Ломоносова с 2022 года. Сфера научных интересов: магнитная гидродинамика, космические магнитные поля, электровихревые течения, математическое моделирование в физике.

Evgeny Alexandrovich MIKHAILOV – Dr. Sci. (Phys.-Math.), Senior Researcher of Lebedev Physical Institute since 2021, Associate Professor of Lomonosov Moscow State University since 2022. Research interests: magnetohydrodynamics, cosmic magnetic fields, electrovortex flows, mathematical modelling in physics.

Антон Александрович ТАРАНЮК – студент физического факультета МГУ имени М.В, Ломоносова, лаборант-исследователь ИТПЗ РАН. Сфера научных интересов: магнитная гидродинамика, электровихревые течения, математическое моделирование в физике.

Anton Alexandrovich TARANYUK – student of Lomonosov Moscow State University, laboratory assistant researcher at IEPT RAS. Research interests: magnetohydrodynamics, electrovortex flows, mathematical modelling in physics.

Анастасия Павловна СТЕПАНОВА – студент физического факультета МГУ имени М.В, Ломоносова. Сфера научных интересов: магнитная гидродинамика, электровихревые течения, математическое моделирование в физике.

Anastasia Pavlovna STEPANOVA – student of Lomonosov Moscow State University. Research interests: magnetohydrodynamics, electrovortex flows, mathematical modelling in physics.

DOI: 10.15514/ISPRAS-2024-36(2)-13

Пространственное POD-разложение эпидемиологических данных COVID-19

С.А. Елистратов, ORCID: 0000-0002-7006-6879 <*sa.elist-ratov@yandex.ru*>

Институт системного программирования им. В.П. Иванникова РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25,
Институт математики им. С.Л. Соболева СО РАН,
630090, Россия, г. Новосибирск, ул. Академика Коптюга, 4.

Аннотация. В работе исследуются пространственно-временные ряды основных эпидемиологических показателей COVID-19 (распространенность, смертность, показатель выздоровления) для различных регионов России. С целью выявления пространственной корреляции применено POD-разложение, выделены основные моды, получены соответствующие временные зависимости; к последним применено шумоподавление с помощью Empirical Mode Decomposition. Показано, что вследствие разного характера временных коэффициентов для исследуемых параметров совместное POD-разложение нецелесообразно. Исследована сходимость разложения к исходным данным в зависимости от числа мод разложения; выявлен экспоненциальных характер этой зависимости.

Ключевые слова: эпидемиология; COVID-19; POD; EMD.

Для цитирования: Елистратов С.А. Пространственное POD-разложение эпидемиологических данных COVID-19. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 181–192. DOI: 10.15514/ISPRAS–2024–36(2)–13.

Благодарности: Работа выполнена при поддержке Российского научного фонда (проект №23-71-10068).

COVID-19 Epidemiological Indicators POD Spatial Decomposition

S.A. Elistratov, ORCID: 0000-0002-7006-6879 <*sa.elist-ratov@yandex.ru*>

Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia,
Sobolev Institute of Mathematics, Siberian Branch of the Russian Academy of Sciences,
4, Acad. Koptyug avenue, Novosibirsk, 630090, Russia.

Abstract. The epidemiological indicators (prevalence, mortality, convalescence) for COVID-19 are investigated as a temporary-spatial dependencies. Proper Orthogonal Decomposition is applied for the first time to this type of data; the main modes and corresponding coefficients are obtained. Due to this method, it is shown that there are modes concentrated in the particular regions which means there are independent factors for disease spreading. Additionally, it is showed that Empirical Mode Decomposition can be successfully applied for noise-reduction and better understanding time dependencies. The exponential nature of the decomposition error decree shows the accuracy of the decomposition. Despite the ability of POD to reveal hidden dependencies, it requires rows of simultaneous data, which in fact may not be so. In the article the correction applied is discussed, however it may not be enough because of mistakes in raw data. The method is recommended to use unless the data it is applied to is inaccurate.

Keywords: epidemiology; COVID-19; POD; EMD.

For citation: Elistratov S.A. COVID-19 epidemiological indicators POD spatial decomposition. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 181-192 (in Russian). DOI: 10.15514/ISPRAS-2024-36(2)-13.

Acknowledgements. The investigation was conducted under the help of RSF (project №23-71-10068).

1. Введение

Установление пространственной корреляции между эпидемиологическими показателями является важной задачей при оценке рисков и осуществлении мер по противодействию распространению инфекционных заболеваний. В настоящее время прогностические модели развиты слабо, поэтому важную роль в изучении этого вопроса играет обработка и интерпретация полевых данных.

Основные модели распространения инфекционных заболеваний сосредоточены [1-2] на эволюции показателей в конкретной области; существующие модели пространственной передачи инфекции основаны на клеточных автоматах и описывают небольшие модельные объекты с высокой плотностью и простой геометрией [3-5]. Интересным представляется подход [6] на основе имитационного моделирования, однако его применение для разбиения на большое число регионов представляется затруднительным.

Учесть пространственную динамику в дифференциальных уравнениях не представляется возможным вследствие бесструктурного расположения населенных пунктов, а модели сплошной среды в масштабах регионов неприменимы из-за малого размера последних по сравнению с расстояниями между ними.

Тем не менее, оказывается возможным связать показатели заболеваемости во времени и пространстве с помощью POD-разложения. Этот метод уже зарекомендовал себя в динамике сплошных сред, где используется для выделения характерных мод течений; однако область его применения не ограничивается задачами с непрерывной средой. Метод лишен необходимости использовать какие-либо сетки и воспринимает зависимости в разных точках как именованные временные ряды, что позволяет применять его для записей данных в ячейках разного размера, разделенных сложной границей, роль которых в текущем исследовании играют субъекты административного деления. Такое разложение позволяет отследить пространственную корреляцию между данными из разных регионов и выделить характерные моды эволюции эпидемиологических показателей.

2. Методы

2.1 Методы разложения

В общем виде POD-разложение позволяет представить двумерный массив данных в виде суммы слагаемых с разделенными индексами:

$$u_{ij} = \sum_k T_i^k \Phi_j^k \quad (1)$$

Для разложения используется следующая процедура: составляется матрица ковариации $C = u^T u$, собственные вектора которой являются векторами разложения Φ^k. Собственные значения называются энергиями мод и отражают вклад каждой компоненты в разлагаемую функцию; в случае разложения физического поля скорости они пропорциональны кумулятивной кинетической энергии моды. Предполагается, что собственные вектора разложения ортогональны, поэтому временные коэффициенты можно найти путем умножения исходной матрицы $[u_{ij}] \equiv U$ на транспонированную матрицу мод. Для возможности сравнения временных коэффициентов моды разложения нормируют в L^2, что в тоже время делает конкретные числовые значения коэффициентов весьма условными из-за масштабирования.

В случае больших данных поиск собственных значений может занимать длительное время; однако в силу специфики матрицы ковариации те же моды и энергии могут быть получены путем не как собственных вектора матрицы ковариации C, а как сингулярные вектора исходной матрицы U, а энергии – как квадраты сингулярных чисел. В самом деле, пусть ξ и η – правый и левый вектора сингулярного разложения U соответственно; σ – сингулярное число:

$$U\xi = \sigma\eta;\ U^T\eta = \sigma\xi$$

Умножая слева на U^T, получим:

$$U^T U\,\xi = \sigma U^T\eta = \sigma^2\xi$$

или

$$C\xi = \sigma^2\xi$$

Сингулярное разложение (SVD) быстрее поиска собственных значений, что позволяет сократить время преобразования. Тем не менее, этот алгоритм является довольно затратным в смысле оперативной памяти, и данные с большим числом признаков могут приводить к ее переполнению. Для ее решения может быть применен алгоритм с использованием машинного обучения [7], для которого требуется лишь память под необходимое число мод, однако в случае, когда число пространственных точек невелико, предпочтительным является алгоритм SVD как более точный.

В непрерывных переменных разложение (1) можно интерпретировать следующим образом: первый индекс соответствует времени, второй – некоторому признаку:

$$u(x,t) \approx \sum_k T^k(t)\,\Phi^k(x), \qquad (2)$$

где t – время, x – обобщенная координата. При этом функции $T^k(t)$ называются временными коэффициентами разложения, а $\Phi^k(x)$ – модами. Поскольку само разложение (1) никак не ограничивает выбор параметра x, это могут быть как именованные признаки, так и координаты; кроме того, возможно совместное разложение нескольких признаков по координатам (например, разных компонент скорости [8]). POD-разложение по координатам находит широкое применение в задачах механики сплошных сред [9-13] для выделения пространственных мод течения; случаи применения пространственного POD-разложения в эпидемиологии неизвестны. Поскольку разложение не накладывает ограничение на признак, для координатного разложения не требуется специальная сетка. Это преимущество позволяет выделить в качестве мод региональное распределение исследуемой функции.

2.2 Данные и предобработка

В качестве данных использовались данные о заболеваемости COVID-19 с марта 2020 по октябрь 2023 года. Эти данные содержат распределение числа заражений, выздоровлений и смертей, а также их кумулятивные показатели по регионам России. Из-за наличия в данных ошибочных записей (отрицательное число новых смертей, убывание кумулятивных показателей и т.д.) была проведена их предобработка: записи (ряд данных для определенной даты в конкретном регионе) с отрицательными дифференциальными данными удалялись, и кумулятивные данные восстанавливались по дифференциальным (был использован такой подход, потому что ошибки в кумулятивных данных ведут к накоплению ошибки, и ошибка в одной записи требует реконструкции всех последующих). Из-за того, что сделаны в разных регионах могут быть записаны не каждый день и приходятся на разные даты, был определен общий для всех период наблюдения (с 17 апреля 2020 по 24 сентября 2023 года) и проведена процедура интерполяции на посуточную сетку, поскольку для POD-разложения требуется матрица одновременных записей. Затем кумулятивные данные дифференцировались для восстановления дифференциальных величин. Полученная выборка содержит 1255 срезов по времени. Сразу оговоримся, что будем рассматривать только дифференциальные (суточные) показатели, поскольку для них временная динамика видна более наглядно.

Пространственные моды у соответствующих суточных и кумулятивных показателей одинаковы, поскольку они связаны суммированием только по времени (см. выражение (2)).

3. Результаты

3.1 Распространенность

Разложение заболеваемости (числа новых случаев заражения) оказалось нерепрезентативным, поскольку основной вклад приходится на первую моду (>85%), которая целиком сосредоточена на крупных городах (Москва и Санкт-Петербург на рис. 1). Это происходит из-за неодинаковой численности населения в разных регионах. Поэтому для POD-разложения имеет смысл рассматривать не заболеваемость, а распространенность (число новых случаев заражения на 100.000 человек); дальнейшие данные также рассматриваются нормированными по населению региона.

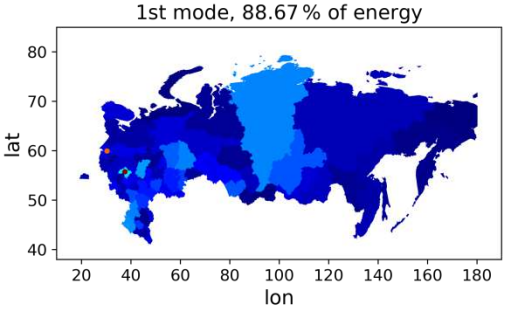


Рис. 1. Первая мода разложения заболеваемости
Fig. 1. First POD mode of incidence

На рис. 2 представлены временные коэффициенты разложения. Первая мода с наиболее высоким (64%) вкладом в разложение несмотря на нормировку имеет довольно неравномерное распределение числа новых случаев заражения (рис. 4), причем наибольшее значение наблюдаются в Ямало-Ненецком автономном округе, в то время как в Москве наблюдается довольно низкая заболеваемость. Причина такой высокой заболеваемости в ЯМАО неясна и требует отдельного исследования.

Отметим, что несмотря на неотрицательность исходного временного ряда коэффициенты POD-разложения могут принимать отрицательные значения, например, мода гашения. Такая мода уменьшает значение исследуемой функции на своем носителе по сравнению с более высокоэнергетическими компонентами.

В случае анализа числа заболеваний подобное поведение присуще моде 2 со значительным вкладом (17%), которая сосредоточена на Москве, Санкт-Петербурге и Якутии. Мода была активна в первом квартале 2022 года; в то же время наблюдался максимум временного коэффициента основной моды. Интересно, что в период подъема заболеваемости мода 2 имеет положительные значения (то есть заболеваемость в этих регионах росла быстрее), а в период спада, наоборот, отрицательные, что говорит о том, что процесс роста и спада заболеваемости происходит быстрее на фоне средней картины.

Зашумленность данных проявляется в виде высокочастотных осцилляций временных коэффициентов. Сами по себе осцилляции с характерным периодом в несколько дней представляют мало интереса, поэтому разумно будет выделить основную динамику. Осреднение в окнах не является эффективным, поскольку не осредняет должным образом функции с резкими изменениями.

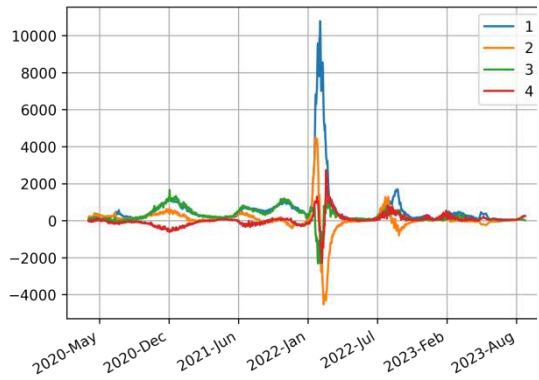


Рис. 2. Временные коэффициенты разложения для распространенности

Fig.2 Temporal POD coefficients of prevalence

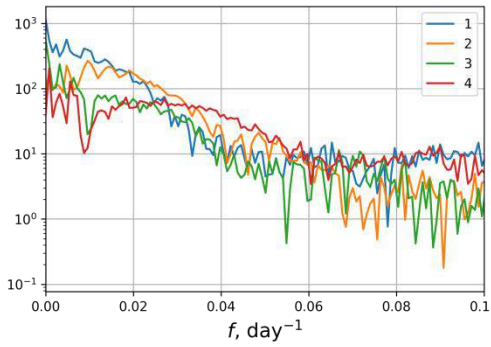


Рис. 3. Спектры временных коэффициентов

Fig.3. Prevalence POD coefficients spectra

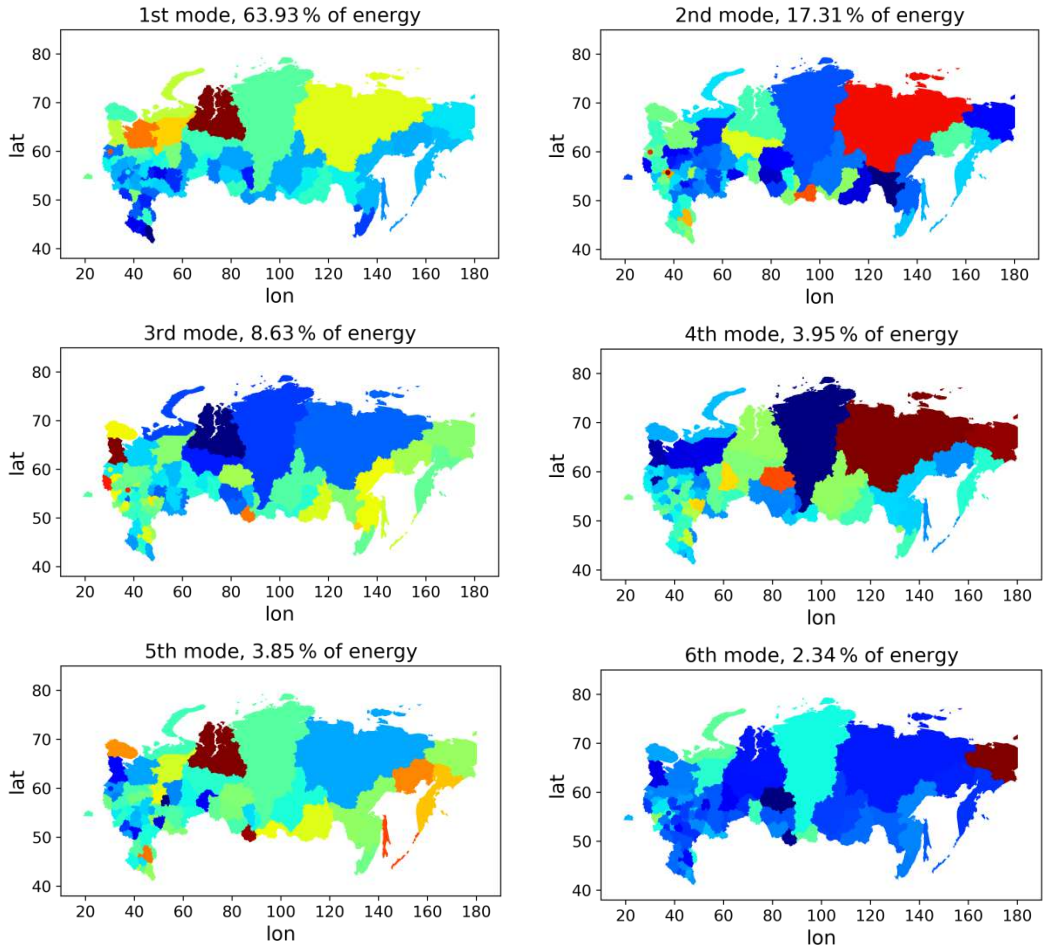


Рис. 4. Моды разложения распространенности

Fig. 4. POD modes of prevalence

Чтобы избавиться от высокочастотного шума, предлагается использовать разложение по эмпирическим модам (преобразование Гильберта-Хуанга, EMD [14]). Отдельные моды не могут быть использованы сами по себе, однако учитывая, что моды выделяются, начиная с наиболее высокочастотной компоненты, для осреднения можно вычесть первые несколько эмпирических мод из исходного временного ряда. Результат такого преобразования для временных коэффициентов разложения зависимости числа новых заболеваний представлен на рис. 5 (вычитались первые 3 моды); устранение осцилляций позволяет сделать этот график более наглядным (ср. с рис. 2). При этом такое осреднение ведет к выполаживанию спектра в области высоких частот (рис. 3, 6); тем не менее, EMD в том виде, в котором оно используется, является лишь способом улучшить репрезентативность временных коэффициентов и никак не сказывается на пространственных модах.

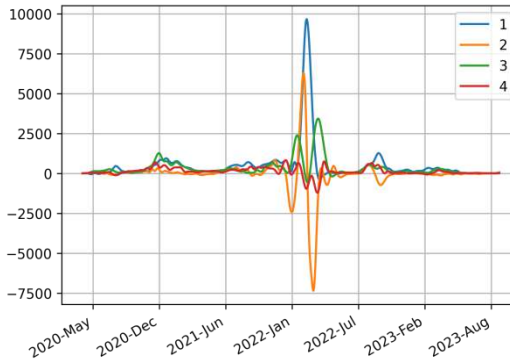


Рис. 5. Временные коэффициенты разложения для распространенности (EMD-фильтрация)

Fig.5 Temporal POD coefficients of prevalence (with EMD filtration applied)

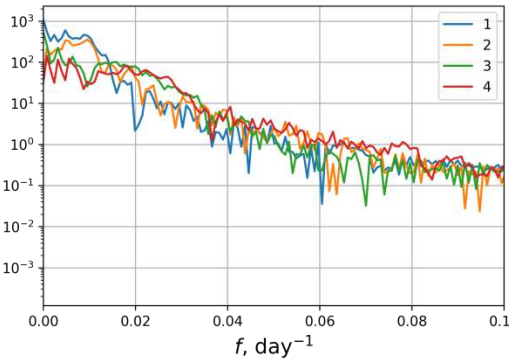


Рис. 6. Спектры временных коэффициентов (EMD-фильтрация)

Fig.6 Temporal POD coefficients of prevalence (with EMD filtration applied)

3.2 Смертность

Несмотря на то, что датасет содержит несколько эпидемиологических показателей, а POD-разложение позволяет проводить процедуру декомпозиции для нескольких разлагаемых функций, мы намеренно не делаем такое разложение на эпидемиологических данных. Дело в том, что совместное разложение имеет жесткое условие в виде общего временного коэффициента:

$$\begin{pmatrix} u(x,t) \\ v(x,t) \end{pmatrix} \approx \sum_k \begin{pmatrix} \Phi_u^k(x) \\ \Phi_v^k(x) \end{pmatrix} T^k(t) \qquad (3)$$

где u и v — две совместно разлагаемые функции. Из вида разложения (3) видно, что несмотря на разные пространственные составляющие временной коэффициент у них общий. Поэтому такое разложение, например, для числа новых случаев заражения и числа летальных исходов, вообще говоря, имеет мало смысла, поскольку даже при абсолютной корреляции между заражениями и смертями смертельный исход не наступает одновременно с заражением.

Временные коэффициенты разложения и соответствующие осредненные кривые представлены на рис. 7-8. Видно, что они не имеют такой четкой структуры, как коэффициенты разложения числа заражения, и более наглядно демонстрируют работу EMD.

Первая мода разложения (рис. 9) имеет очень неравномерный характер и максимумы совсем не в тех регионах, что заболеваемость. Вторая мода сосредоточена на Ненецком автономном округе, а характер соответствующего коэффициента указывает на то, что смертность в этом регионе в период повышенной смертности в целом по стране (с июня 2021 по март 2022 года)

значительно колебалась. Причины такого поведения в одном конкретном регионе неизвестны и требуют исследования факторов, не включенных в датасет.

Различие в пространственном распределении этих мод для заболеваемости и смертности свидетельствует о том, что на эти показатели влияют разные региональные факторы, а также подтверждает, что они должны быть исследованы отдельно друг от друга.

Третья и четвертая мода, локализованы в отдельных восточных регионах (рис. 9), что позволяет предположить наличие в них независимых факторов, влияющих на смертность.

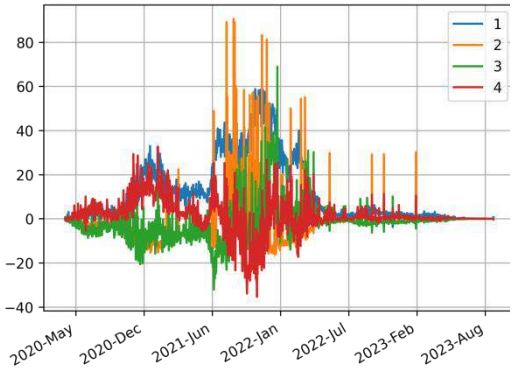


Рис. 7. Временные коэффициенты разложения смертности
Fig.7 Temporal POD coefficients of mortality

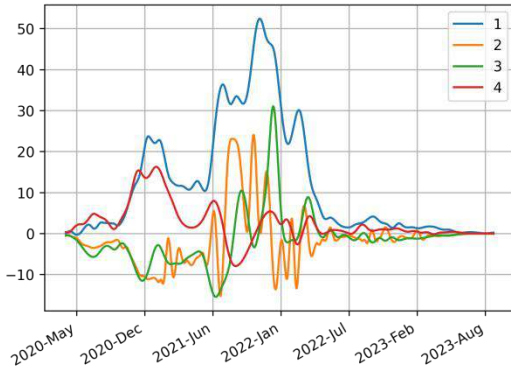


Рис. 8. Временные коэффициенты разложения смертности (EMD-фильтрация)
Fig.8 Temporal POD coefficients of mortality (with EMD filtration applied)

3.3 Показатель выздоровления

Пространственные моды числа выздоровлений (рис. 11) вкупе с их временными коэффициентами (рис. 14) обнаруживают сходство этого показателя с числом новых заболеваний и свидетельствует о пространственной корреляции между ними; корреляция между заболеваемостью/выздоровлениями и смертностью довольно слабая. Это может быть вызвано тем, что период выздоровления варьируется незначительно, а время от заболевания до смерти в значительной степени индивидуально для каждого летального пациента. Тем не менее, несмотря на схожесть мод и временных коэффициентов упомянутых эпидемиологических показателей, распределение энергий для их разложения (рис. 12-13) различается.

3.4 Сходимость разложения

Выделение первых мод дает представление о поведении решения и основных трендах его эволюции, однако если POD-декомпозиция применяется для сжатия данных (при хранении первых n мод и их временных коэффициентов) необходимо исследовать сходимость такого разложения к исходным данным. Для этого рассмотрим ошибку в виде:

$$error = \left\| \sum_{k=1}^{n} T_i^k \Phi_j^k - U_{ij} \right\|_2^2 / \left\| U_{ij} \right\|_2^2$$

где U — исследуемая совокупность признаков, как функцию от числа рассматриваемых мод n, $\|\cdot\|_2$ – квадратичная норма по времени и пространству.

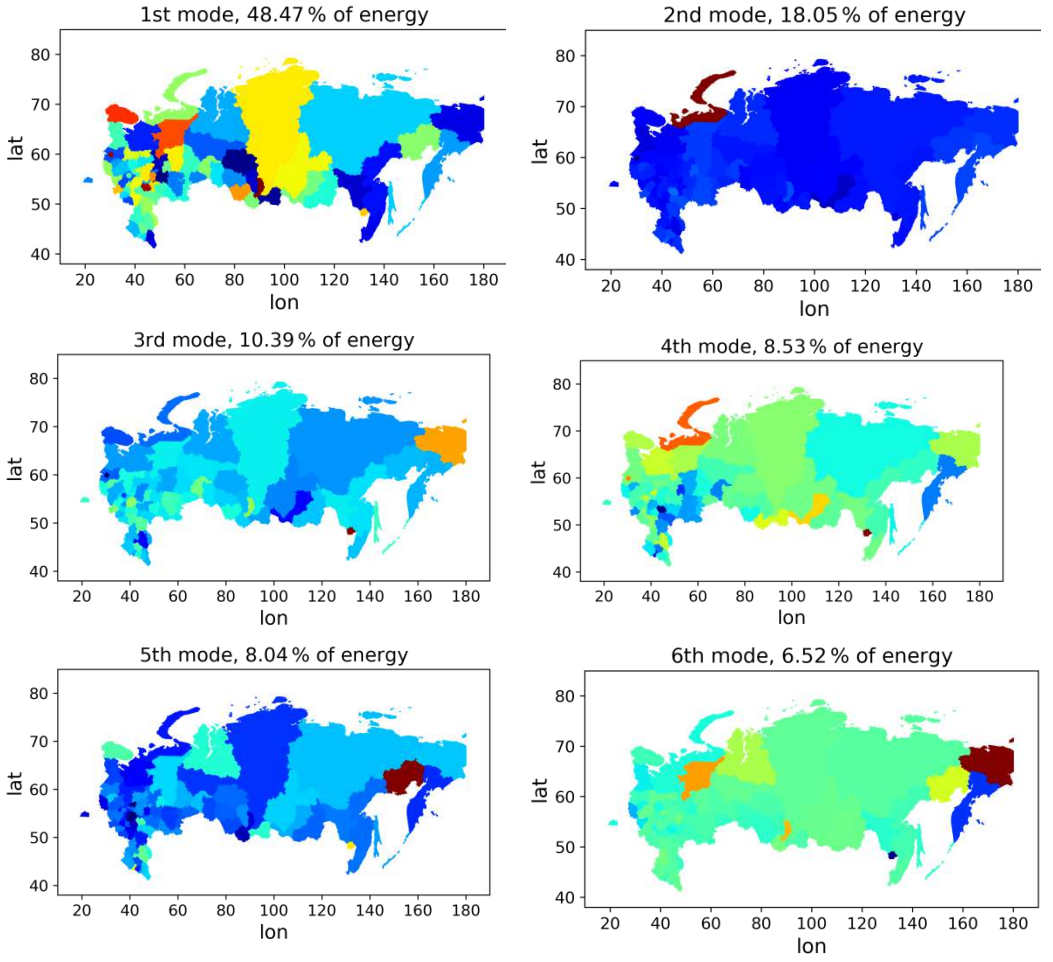


Рис. 9. Моды разложения смертности
Fig.9 POD modes of mortality

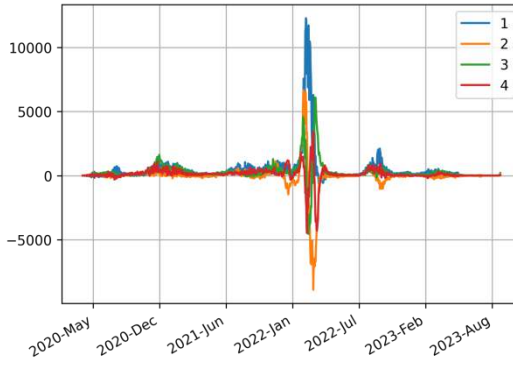


Рис. 10. Временные коэффициенты показателя выздоровления
Fig.10 Temporal POD coefficients of convalescence)

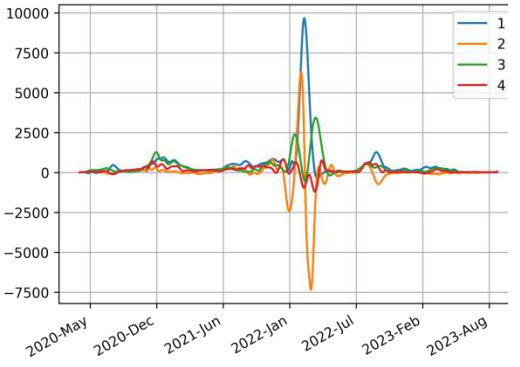


Рис. 11. Временные коэффициенты разложения показателя выздоровления (EMD-фильтрация)
Fig.11 Temporal POD coefficients of convalescence (with EMD filtration applied)

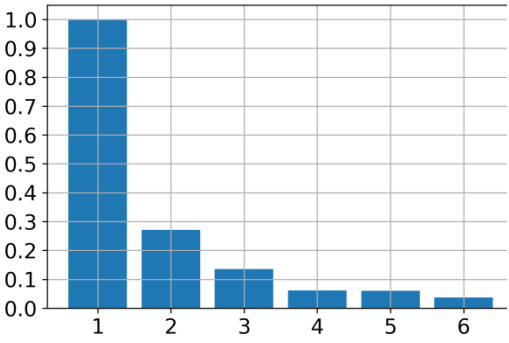


Рис. 12. Энергии мод разложения распространенности

Fig.12 Prevalence POD eigenvalues

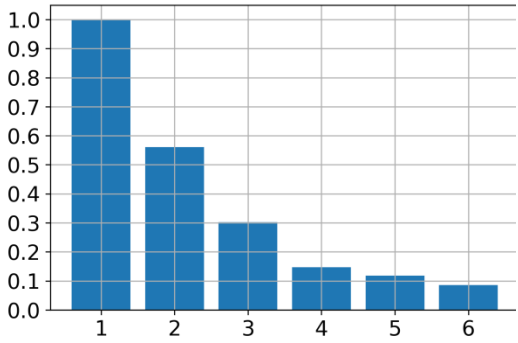


Рис. 13. Энергии мод разложения показателя выздоровления

Fig.13 Convalescence POD eigenvalues

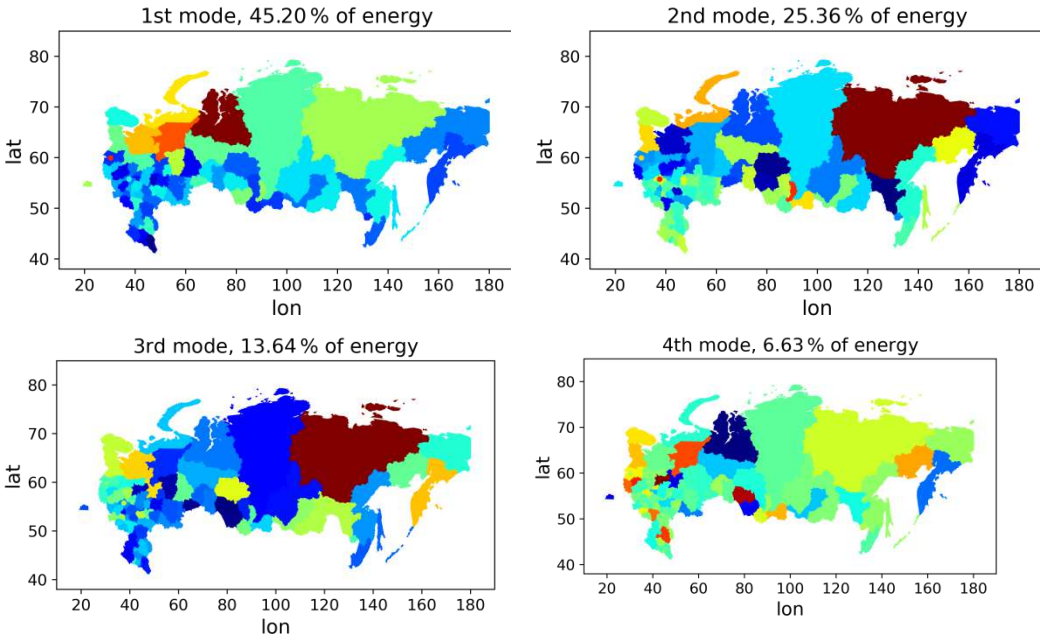


Рис. 14. Моды разложения показателя выздоровления

Fig.14. POD modes of convalescence

На рис. 15 представлены зависимости ошибки для разложений отдельных параметров (распространенность, смертность, показатель выздоровления). Рассматривать число мод >85 не имеет смысла, поскольку это полное число регионов. Видно, что на большей части интервала затухание ошибки имеет экспоненциальный характер — на графике приведены аппроксимирующие зависимости. Показатели затухания (множители в экспоненте) близки, однако неодинаковы, причем наиболее отличается показатель не для смертности, а для распространенности. Экспоненциальный характер затухания мод свидетельствует о том, что POD может эффективно применяться для компрессии этих данных. О неэффективности совместного разложения также свидетельствует рис. 16, показывающий корреляцию первых десяти мод разложений трех эпидемиологических показателей. Видно, что вне диагонали довольно слабая корреляция наблюдается между первыми модами распространенности и показателя выздоровления, что вполне логично; однако в остальном корреляция не

превышает 0.5, что можно расценивать как отсутствие линейной связи между модами различных показателей.

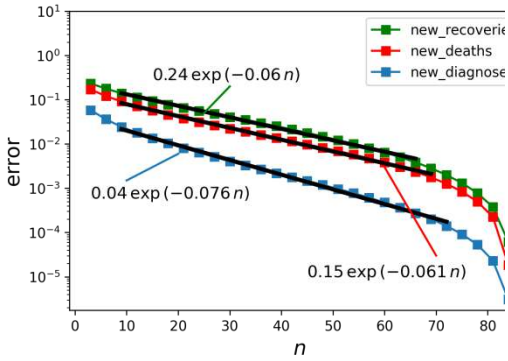


Рис. 15. Ошибка разложения
Fig.15 Decomposition error

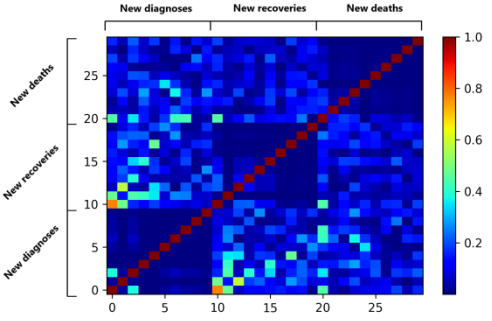


Рис. 16. Коэффициенты корреляции мод различных показателей
Fig.16 Different indicators modes correlation

4. Заключение

POD-разложение применяется в эпидемиологии довольно редко, а его пространственный вариант рассматривается впервые. Несмотря на это, POD является достаточно удобным инструментом при анализе эпидемиологических данных и позволяет проследить их связь. На примере трех рассматриваемых показателей (показателей заболеваемости, смертности и выздоровления) показано, что поведение смертности значительно отличается от поведения остальных показателей, что может быть обусловлено внутренними причинами этих явлений. Несмотря на то, что POD-разложение не может объяснить эти причины, анализ данных с его помощью поможет скорректировать направление исследований эпидемиологов для более глубокого понимания механизма распространения заболеваний.

К числу недостатков метода можно отнести, во-первых, то, что выявленные таким способом зависимости являются сугубо линейные, и в случае нелинейного характера отношений между различными показателями необходимо применять более алгоритмы. Во-вторых, несмотря на отсутствие привязки к каким-либо пространственным сеткам, для разложения необходимы ряды единовременных записей, чего на практике может не быть. Данные об эпидемиологических показателях зачастую нерегулярны, а порой содержат откровенно неприемлемые ошибки, которые приходится искусственно корректировать при обработке вследствие отсутствия альтернативных данных. Применявшаяся при исследовании корректировка исключением ошибочных записей с последующей интерполяцией может влиять на получаемое разложение, а также никак не исключает скрытые ошибки, поэтому в подобного рода исследованиях необходимо использовать как можно более аккуратные и подробные данные.

Список литературы / References

[1]. Hethcote H. The Mathematics of Infectious Diseases / SIAM Review, vol. 42(4), 2000, pp. 599–653.
[2]. Beckley R., C. Weatherspoon C., Alexander M. et al. Modeling epidemics with differential equations / Tennessee State University Internal Report.
[3]. Santos G., Alves T., Alves G., Filho A. Asynchronous SIR model on Two-Dimensional Quasiperiodic Lattices, vol 01, 2019.
[4]. Bisin A., Moro A. Learning Epidemiology by Doing: The Empirical Implications of a Spatial-SIR Model with Behavioral Responses / Journal of Urban Economics, vol. 127, 2022.

[5]. von Csefalvay C. Spatial dynamics of epidemics: Epidemics in discrete and continuous space / Computational Modeling of Infectious Disease, vol. 13, 2023, pp. 257–303.
[6]. Derevich I., Panova A. (2019). Effects of Random Migration on the Growth of the Population of a Biological System. Journal of Engineering Physics and Thermophysics. 92. 10.1007/s10891-019-02054-x.
[7]. https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html.
[8]. Elistratov S.A. POD-based Hydrodynamical Structures Visualization in Flows with an Internal Wave Attractor / Scientific Visualization, vol. 15(2), 2023, pp. 125 – 133.
[9]. Yilmaz I., Aylı E., Aradag S. Investigation of the Effects of Length to Depth Ratio on Open Supersonic Cavities Using CFD and Proper Orthogonal Decomposition / The Scientific World Journal, vol. 01, 2013, — p. 810175.
[10]. Berkooz G, Holmes P, Lumley J L. The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows / Annual Review of Fluid Mechanics, vol. 25(1), 1993, pp. 539–575, DOI: 10.1146/annurev.fl.25.010193.002543.
[11]. Brunton, Steven L. and Kutz, J. Nathan. 7 Data-driven methods for reduced-order modeling / Snapshot-Based Methods and Algorithms: Volume 2, edited by Peter Benner, Stefano Grivet-Talocia, Alfio Quarteroni, Gianluigi Rozza, Wil Schilders and Luís Miguel Silveira, Berlin, Boston: De Gruyter, 2021, pp. 307-344, DOI: 10.1515/9783110671490-007.
[12]. Nakamura T., Fukami K., Fukagata K. Identifying key differences between linear stochastic estimation and neural networks for fluid flow regressions / SciRep, vol. 12(3), 2020, p. 3726.
[13]. Duan Fan, Wang Jinjun. Fluid–structure–sound interaction in noise reduction of a circular cylinder with flexible splitter plate / Journal of Fluid Mechanics, vol. 920(8), 2021.
[14]. Huang N.E., Shen Z., Long S.R. et al. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis / Proceedings of the Royal Society of London, vol. A4, 1998.

Информация об авторах / Information about authors

Степан Алексеевич ЕЛИСТРАТОВ – сотрудник Лаборатории цифрового моделирования технических систем Института системного программирования с 2021 года. Сфера научных интересов: математическое моделирование, прогностические модели, методы пониженной размерности.

Stepan Alekseevich ELISTRATOV – employee of Technical Systems Digital Modelling Laboratory of the Institute for System Programming of the RAS since 2021. Research interests: numerical simulation, forecast models, reduced order methods.

DOI: 10.15514/ISPRAS-2024-36(2)-14

Исследование структурно незавершённых высказываний в речи советской и российской молодёжи

А.А. Чуев, ORCID: 0000-0002-3352-0446 <*alexanderchuev@yandex.ru*>

Институт системного программирования РАН,
Россия, 109004, г. Москва, ул. А. Солженицына, д. 25.

Аннотация. Целью исследования является сравнение устной речи советских и российских молодых людей в возрасте от 13 до 23 лет. Анализ проводился по единственному показателю: обилию структурно незавершённых высказываний в расшифрованных устных сообщениях. Образцы речи как советских, так и современных российских подростков взяты из СМИ. В результате исследования было показано, что советские школьники и студенты в 7 раз меньше использовали структурно незавершённые высказывания в своей устной речи, чем современная молодёжь. Исследование является продолжением более обширного сравнения устной речи школьников и студентов советской и российской эпох, а также образцов для речевого подражания, которые транслировали советские и российские СМИ. Основанием для выбора материала исследований стала нетривиальная задача установить роль воздействия на речь подростков современной медийной среды. Если советские школьники и студенты потребляли медиаконтент с лучшими для своего времени лингвистическими образцами для подражания, то современные подростки формируются в среде свободного интернета. В ходе исследования разработан метод объективного сравнения речи разных спикеров. Корпусы текстов сравнивались по количеству окказионализмов, авторских синтагм, фразеологизмов, профессионализмов, канцеляризмов, вульгаризмов, структурно незавершённых высказываний, обсценной лексики и т.д. В серии исследований было показано, что речевые образцы советской эпохи, то есть советской группы, лучше как в количественном, так и в качественном отношении, чем современные. Научная новизна заключается в оценке устной речи спикеров информационных и развлекательных медиапродуктов СМИ одной русскоязычной страны, но в разные эпохи, хотя и близкие по времени, с помощью нового комплексного метода лингвистического анализа. Ключевыми факторами, повлиявшими на речевые подходы создателей этих медийных продуктов, стали быстрые социальные и технические изменения.

Ключевые слова: лингвистика; языкознание; грамматика; текст; молодёжь; речь.

Для цитирования: Чуев А. А. Исследование структурно незавершённых высказываний в речи советской и российской молодёжи. Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 193–198. DOI: 10.15514/ISPRAS–2024–36(2)–14

The Study of Structurally Incomplete Statements in the Speech of Soviet and Russian Youth

A.A. Chuev ORCID: 0000-0002-3352-0446 <*alexanderchuev@yandex.ru*>

Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia.

Abstract. The aim of the study is to compare the oral speech of Soviet and Russian young people aged 13 to 23 years. The analysis was carried out according to a single indicator: the abundance of structurally incomplete statements in transcribed oral messages. Speech samples of both Soviet and modern Russian teenagers are taken from the media. As a result of the study, it was shown that Soviet schoolchildren and students used structurally incomplete statements in their oral speech 7 times less than modern youth. The study is a continuation of a more extensive comparison of the oral speech of schoolchildren and students of the Soviet and Russian eras, as well as speech role models that were broadcast by the Soviet and Russian media. The basis for the choice of research material was the non-trivial task of establishing the role of the influence of the modern media environment on the speech of adolescents. If Soviet schoolchildren and students consumed media content with the best linguistic role models for their time, then modern teenagers are formed in the free Internet environment. In the course of the study, a method was developed for objective comparison of the speech of different speakers. Text corpora were compared by the number of occasionalisms, author's syntagmas, phraseological units, professionalisms, clericalisms, vulgarisms, structurally incomplete statements, obscene vocabulary, etc. In a series of studies, it was shown that the speech samples of the Soviet era, that is, the Soviet group, are better both quantitatively and qualitatively than modern ones. The scientific novelty lies in the assessment of the oral speech of the speakers of information and entertainment media products of the media of one Russian-speaking country, but in different eras, although close in time, using a new complex method of linguistic analysis. The key factors that influenced the speech approaches of the creators of these media products were rapid social and technical changes.

Keywords: linguistics; grammar; text; youth; speech.

For citation: Chuev A.A. The study of structurally incomplete statements in the speech of Soviet and Russian youth. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 193-198 (in Russian). DOI: 10.15514/ISPRAS-2024-36(2)-14.

1. Введение

Настоящее сравнение является частью более масштабного исследования устной речи советских и российских молодых людей в возрасте от 13 до 23 лет. Поводом для анализа послужили современные мультимедийные технологии, которые стали формировать новую коммуникативную среду в конце XX и начале XXI веков. Этот процесс совпал по времени с масштабными социальными преобразованиями в России. Было интересно посмотреть, как в контексте политических, бытовых и технических изменений трансформировалась устная речь.

Старшеклассники и студенты выбраны для исследования по нескольким причинам. Молодёжь составляет самобытную социальную группу, многочисленную, но главное, условия для существования которой поддерживаются сферой занятости большинства её акторов: длительным обучением в том или образовательном учреждении. Такая ярко выраженная социальная группа хорошо сохраняет свои особенности из поколения в поколение. А изменения, даже самые небольшие, можно легко проследить простым сравнением жизни учащихся разных эпох. Кроме того, в подростковом возрасте происходит половое созревание, которое существенно влияет на речь. Из-за жестко ограниченных условий взаимодействия и отсутствия различных преимуществ взрослой жизни именно речь становится одним из главных инструментов самоутверждения среди окружающих. В такой обстановке ещё большую важность речь приобретает в общении с противоположным полом. Позже, в зрелом возрасте, значимость самих этих вопросов падает, а также появляются другие социальные преимущества, поэтому речь в жизненном успехе человека перестаёт играть такую важную роль [1].

Следует отметить ещё одну особенность исследования. Образцы устной речи взяты из СМИ. Во время публичного выступления спикер стремится показать себя с лучшей стороны и для этого в том числе прикладывает усилия для решения языковых задач [2]. В качестве материалов для исследования устной речи советской молодёжи использовались выпуски программы «До 16 и старше...», которые выходили на Центральном телевидении СССР, а затем на «1 канале Останкино» и «ОРТ» с 1983 по 2001 год. Выборка ограничена 1991 годом. Для сравнения были отобраны выпуски современных видеоблогеров того же возрастного диапазона, которые размещали свои ролики на видеохостинге YouTube с 2015 по 2021 год. Государственный Всероссийский центр изучения общественного мнения в 2019 году присвоил YouTube звание «телевидения XXI века» [3]. По данным ВЦИОМ, большинство россиян пользуются сервисом. Четверть из них – ежедневно, пятая часть – регулярно на неделе, десятая – эпизодически в течение месяца. Только 20 % россиян вообще не заходят на сайт или в приложение. Примерно столько же россиян не пользуются Интернетом: 22 % [4].

Среди молодёжи YouTube не просто популярен, он стал неотъемлемым атрибутом жизни молодого человека 18-24 лет. По данным ВЦИОМ, видеохостингом пользуются 88 % опрошенных юношей и девушек. Примерно те же цифры у тех, кто постарше. 90 % мужчин и женщин 25–34 лет потребляют контент на YouTube. Число зрителей падает у старшего поколения — граждан 35-44 лет — до 73 %. У россиян 45-59 лет аудитория сервиса снижается до 51 %, а после 60 лет YouTube смотрит только четверть опрошенных.

У телевидения обратные показатели. По данным ВЦИОМ, каждый третий россиянин вообще не смотрит телевизор – 28 %. Это, в частности, 70 % молодых людей 18–24 лет. И почти каждый второй житель страны 25-34 лет. Выбирают телевидение в качестве приоритетного источника информации только 17 % опрошенных граждан. Каждый второй из них – старше 60 лет.

В 2015-2021 годы верхние строчки рейтингов самых популярных видеоблогеров на YouTube в возрасте от 13 до 23 лет возглавляли: Влад А4 (более 30 миллионов подписчиков), Алишер Моргенштерн (более 10 миллионов подписчиков), Катя Клэп (более 7 миллионов подписчиков), Саша Спилберг (более 6 миллионов подписчиков), Марьяна Ро (более 6 миллионов подписчиков), Николай Соболев (более 5 миллионов подписчиков), Инстасамка (более миллиона подписчиков).

В качестве материалов для исследования устной речи российской молодёжи использовались фрагменты интервью с обозначенными видеоблогерами, а также — другими медийными лицами указанного возрастного диапазона. Это артисты и видеоблогеры Face, Pharaon, Кирилл Бледный, Монеточка, Тима Белорусских, Slava Marlow, Даня Милохин, Юля Гаврилина, Молодой Платон, Аслан Шукаша и Глеб Калюжный.

2. Методы и результаты

Ранее автором настоящего исследования было показано, что для сравнения устной речи советской и российской молодёжи можно использовать разные подходы [5]. Не все они одинаково эффективны. Особого внимания заслуживает метод поиска структурно незавершённых высказываний, подробно описанных в докторской диссертации Татьяны Колокольцевой: «Структурно незавершённые высказывания в русской разговорной речи». Защита состоялась ещё в советское время, в 1984 году [6]. Как следует из названия, автор рассматривает причины возникновения структурно незавершённых высказываний, их признаки, конструктивно-сешшческие и интонационные особенности.

Структурно незавершённые высказывания (СНВ) Татьяна Колокольцева определяет как «широко распространенные в спонтанном общении построения, характеризующиеся в конструктивном плане обрывом синтагматической цепи, в содержательном — наличием невербализованных смыслов и возможностью субъективно-модальных или подтекстных

коннотаций, а в просодическом — специфическим интонационным оформлением». Например, «Валерка в общем-то тоже способный парень, но Женька!».

Некоторые исследователи считают такие построения ненормативными. Например, О.А. Лаптева пишет: «разговорная речь изобилует незаконченными построениями, неоформленными фразами, «самоперебивами», самыми различными приёмами вставок и добавлений (подчас же не получающих конструктивного завершения)» [7]. Другие — рассматривают СНВ как норму, а именно как разновидность эллипсиса в разговорной речи [8]. Например, «Боюсь, не увижу: у меня зрение», то есть слабое зрение.

Главное, исследователи сходятся во мнении, что СНВ экономят усилия говорящего. Вместо того, чтобы закончить фразу, спикер обрывает её по принципу достаточности и тем самым предлагает слушающему достроить предложение самостоятельно, исходя из контекста. Причём обрывает в любом месте — пишет Колокольцева — «для них не существует каких-либо структурных ограничений или запретов. Здесь встречаются и такие незамещенные позиции, которые невозможны ни в каких иных типах высказываний. Так, например, СНВ могут заканчиваться словами, морфолого-синтаксические особенности которых предопределяют обязательное наличие при них каких-либо других слов или предикативных конструкций (союзы, предлоги, связки, некоторые частицы)». Например, «у вас здесь жарко, а у нас...».

Известный в лингвистике принцип экономии усилий говорящего и слушающего в 1940 году описал ученик Фердинанда де Соссюра и один из издателей его знаменитого «Курса общей лингвистики» — Альбер Сеше [9]. Говорящий старается уменьшить расход своей энергии по аналогии с любым другим видом человеческой деятельности. Однако это возможно до тех пор, пока экономия не угрожает самой цели коммуникации. То есть слушающий должен понимать, что ему сообщают и при этом не прикладывать для расшифровки послания больше усилий, чем спикер.

Татьяна Колокольцева в своей диссертации не проводит количественного сравнения СНВ в устной речи разных спикеров и у социальных групп. В настоящем исследовании опробован новый подход. Для сравнения были составлены два корпуса текстов. Это расшифровки фрагментов интервью и выступлений спикеров разных эпох: советской группы (1983-1991) и российской (2015-2021). В каждом корпусе были отобраны СНВ. Например, в интервью с первокурсниками МГУ в 1986 году встречаются такие формулировки: «Мы пытались прервать эту цепь как-то, но видимо…» или «Мы приехали сюда с такими, довольно настроениями...». Их российские сверстники тоже допускают в своей устной речи СНВ. Например, видеоблогер Владислав Бумага обрывает фразу без пояснений: «Я даже перестал…» или другой молодой шоумен Алишер Моргенштерн безуспешно пытается сформулировать свои мысли: «это действительно та вещь, которая вот...».

Расшифрованные интервью советских и российских молодых людей в возрасте от 13 до 23 лет составили два корпуса текстов: корпус советской группы и корпус российской группы. Каждый объёмом более 10 тысяч слов. Извлечённые из корпусов СНВ — это некий процент от общего количества слов в том или ином корпусе, то есть каждый корпус был разделён на две категории слов. В первую вошли не сформулированные сообщения, которые можно интерпретировать как не сформулированные предложения. Например: «А мне реально было... ну прям боль... ну прям больной живот». Или ещё пример: «Это не то, что там как-то... какие-то дяди вмешались там за меня или прочее, прочее... или мы как-то со своей подачи... то есть это официальная тема и... вот, как-то так...». Иногда такие не сформулированные сообщения вклинивались в повествование как отступления: таким образом, что при извлечении подобного сообщения окружающие его части другого сообщения приобретали целостность. Например, «Да не знаю, мне кажется, у молодёжи всегда есть такое желание что-то запретное сделать, что-то запретное обсуждать, что-то такое дур... дурное и всё такое... где-то... в какой-то обстановке, где нет их родителей и так далее... интернет это самое подходящее место, я считаю, для этого». Если извлечь «что-то такое дур...

дурное и всё такое... где-то... в какой-то обстановке, где нет их родителей и так далее... », то получится целостное сообщение: «Да не знаю, мне кажется, у молодёжи всегда есть такое желание что-то запретное сделать, что-то запретное обсуждать — интернет это самое подходящее место, я считаю, для этого».

Извлечённые из корпуса не сформулированные сообщения составили массив слов. Далее было посчитано какой процент этот массив занимает от общего количества слов в корпусе. Соответственно, вторую категорию составили все остальные слова корпуса.

Дальше были посчитаны пропорции: какую долю от общего количества слов корпуса занимает первая категория, а какую — вторая. В результате оказалось, что первая категория у советской группы составила 4,5%. А вторая, соответственно, – 95,5%. У российской группы первая категория составила 30%, а вторая – 70%. Таким образом первые цифры у советской и российской групп различаются в 7 раз. Иными словами, молодые российские видеоблогеры в 7 раз больше наполняли свою речь СНВ, чем их советские сверстники.

3. Заключение

Почти треть расшифровок устной речи спикеров из второй группы – это образцы одного из способов экономии собственных усилий на формулирование мыслей, а именно СНВ. Такие допущения в устной речи негативно сказываются на восприятии повествования слушателями. У советской группы СНВ встречаются эпизодически и на речь почти не влияют.

Из вышеописанного можно сделать вывод, что в отличие от советских подростков, российские молодые люди сильно упрощают свою речь, то есть максимально экономят на ней свои усилия. И уже зрителям приходится самим достраивать речь ораторов исходя из контекста, что существенно усложняет коммуникацию.

Ранее автором настоящего исследования было показано, что речевые образцы молодые люди в значительной степени заимствуют именно из средств массовой информации. На слух подражать проще, чем текстам художественной литературы. Исследование прагмалингвистических конструкций в кинолентах и сериалах, популярных у советской и российской молодёжи подтвердило, что упрощение устной речи в фильмах, востребованных у подростков, коррелирует со снижением общего уровня владения русским языком у самих школьников и студентов. Юные спикеры берут на вооружение наиболее примитивные речевые приёмы, и позволяют себе экономить усилия на повествовании также, как это делают их современники из СМИ.

С появлением социальных сетей это явление усугубилось тем, что доступ к массовой аудитории получили все участники коммуникативной среды. Поэтому образцом для подражания, в том числе речевого, теперь может стать любой человек. Вне зависимости от возраста, пола, профессиональных навыков и способностей. Как было показано выше, на смену телевидению, советским переводчикам и сценаристам пришли любительские видеоблоги, и они не меньше, а в некоторых случаях даже больше востребованы у молодёжи. Сами выходцы из школьной и студенческой среды стали важными объектами для подражания, в том числе речевого, у своих сверстников. При этом любительский контент молодых спикеров с точки зрения языка в основном является трудным для восприятия. Несет в себе образцы экономии усилий, а не выразительных средств, если сравнивать со СМИ советской эпохи. Одним из примеров способа упрощения речи являются СНВ, которые современные видеоблогеры используют несравненно чаще, чем ораторы советской группы, что и было показано в настоящем исследовании.

В результате новый метод сравнения устной речи разных спикеров и групп подтвердил свою эффективность. Он демонстрирует большие количественные различия в использовании СНВ. Позволяет оценить качество речи с точки зрения слушателя и сделать выводы о масштабе экономии усилий говорящего. Исследование выявило, что современные видеоблогеры в

сравнении со своими советскими сверстниками в несколько раз чаще используют СНВ. Речь видеоблогеров значительно труднее для восприятия. В связи с чем можно сделать вывод, что участники российской группы чрезмерно экономят усилия.

Список литературы / References

[1]. Савельев С.В. Морфология сознания. М., ВЕДИ, 2021, Т. 2, 208 с. / Savel`ev S.V. Morfologiya soznaniya. M., VEDI, 2021, T. 2, 208 p. (in Russian).

[2]. Савельев С.В. Нищета мозга. М., ВЕДИ, 2014, 192 с. / Savel`ev S.V. Nishheta mozga. M., VEDI, 2014, 192 p. (in Russian).

[3]. Всероссийский центр изучения общественного мнения, электронный адрес: https://wciom.ru/analytical-reviews/analiticheskii-obzor/youtube-televidenie-xxi-veka, доступно 25.11.2019. / Vserossijskij centr izucheniya obshchestvennogo mneniya, Available at: https://wciom.ru/analytical-reviews/analiticheskii-obzor/youtube-televidenie-xxi-veka, accessed 25.11.2019.

[4]. Всероссийский центр изучения общественного мнения, электронный адрес: https://wciom.ru/analytical-reviews/analiticheskii-obzor/mediapotreblenie-rossijan-monitoring, доступно 03.03.2021. / Vserossijskij centr izucheniya obshchestvennogo mneniya, Available at: https://wciom.ru/analytical-reviews/analiticheskii-obzor/mediapotreblenie-rossijan-monitoring, accessed 03.03.2021.

[5]. Мисонжников Б.Я., Чуев А.А. Исследование прагмалингвистических конструкций в кинолентах и сериалах, популярных у советской и российской молодёжи, Мир лингвистики и коммуникации: электронный научный журнал, вып. 68, 2022 г., стр. 165-181, электронный адрес: www.tverlingua.ru / Misonzhnikov B.YA., Chuev A.A. Issledovanie pragmalingvisticheskih konstrukcij v kinolentah i serialah, populyarnyh u sovetskoj i rossijskoj molodyozhi, Mir lingvistiki i kommunikacii: elektronnyj nauchnyj zhurnal, vol. 68, 2022, pp. 165-181 (in Russian), Available at: www.tverlingua.ru

[6]. Колокольцева Т.Н. Структурно незавершенные высказывания в русской разговорной речи, на дис. на соискание степени доктора филологических наук: 10.02.01, Саратов, 1984, 197 с., электронный адрес: https://search.rsl.ru/ru/record/01003428014 / Kolokol'ceva T.N. Strukturno nezavershennye vyskazyvaniya v russkoj razgovornoj rechi, na dis. na soiskanie stepeni doktora filologicheskih nauk: 10.02.01, Saratov, 1984, 197 p. (in Russian), Available at: https://search.rsl.ru/ru/record/01003428014

[7]. Лаптева О.А. Теория современного русского литературного языка: Учебник [Для вузов], М., Высш. шк., 2003, 351 с. / Lapteva O.A. Teoriya sovremennogo russkogo literaturnogo yazyka: Uchebnik [Dlya vuzov], M., Vyssh. shk., 2003, 351 p. (in Russian).

[8]. Сигал К.Я. Эллипсис как речевой процесс (на материале словосочетаний), Вопросы психолингвистики, вып. 1(27), 2016 г., электронный адрес: https://cyberleninka.ru/article/n/ellipsis-kak-rechevoy-protsess-na-materiale-slovosochetaniy / Sigal K.YA. Ellipsis kak rechevoj process (na materiale slovosochetanij), Voprosy psiholingvistiki, vol. 1(27), 2016, Available at: https://cyberleninka.ru/article/n/ellipsis-kak-rechevoy-protsess-na-materiale-slovosochetaniy

[9]. Сеше А. Программа и методы теоретической лингвистики: Психология языка, М., Едиториал УРСС, 2010, 212 с. / Seshe A. Programma i metody teoreticheskoj lingvistiki: Psihologiya yazyka, M., Editorial URSS, 2010, 212 p. (in Russian).

Информация об авторах / Information about authors

Александр Александрович ЧУЕВ является специалистом сектора Сопровождения научно-исследовательских работ и образовательной деятельности ФГБУН Институт системного программирования имени В.П. Иванникова Российской академии наук, с 2023 года. Сфера научных интересов: прагмалингвистика, синтаксис, лексика.

Alexander Alexandrovich CHUEV is a specialist of the sector Support of research and educational activities Ivannikov Institute for System Programming of the Russian Academy of Sciences, since 2023. Research interests: pragmalinguistics, syntax, vocabulary.

DOI: 10.15514/ISPRAS-2024-36(2)-15

Functional and Semantic Analysis of the Case Markers in Vakh Khanty (a Case Study of the Latest Field Data on LingvoDoc)

[1,2] *V.V. Vorobeva*, ORCID: 0000-0002-8729-0375 <*vorobeva@tpu.ru*>
[3] *I.V. Novitskaya*, ORCID: 0000-0003-1559-8810 <*irno2012@yandex.ru*>

[1] *Information Systems Department,*
Ivannikov Institute for System Programming of the RAS
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia.
[2] *National Research Tomsk Polytechnic University,*
30, Lenina Avenue, Tomsk, 634050, Russia.
[3] *National Research Tomsk State University*
36, Lenina Avenue, Tomsk, 634050, Russia.

Abstract. In this paper we argue that the category of nominal case in the Vakh Khanty dialect needs reviewing which is due to the availability of the recent field data obtained in 2019. This objective is considered to be topical because the results of the study will equip researches with all necessary data to develop a unified notation system for the case markers to be used in an annotated corpus of Vakh Khanty on the LingvoDoc platform. The declension system of the Eastern Khanty dialect under analysis is heterogeneous and relatively vast. The controversy around the synthetic means of expression of semantic relations concerns the terminology of case markers, their quantity, morphemic status and functional features. The purpose of this paper is to clarify the functional and semantic aspects of the Vakh Khanty case markers drawing on the recently obtained field material. This objective is achieved by means of the tools available on the LingvoDoc platform. The latest field data on the Vakh Khanty dialect was collected in the village of Korliki in 2019. More than 6,000 words of data are posted on the endangered languages documentation platform LingvoDoc. Part of the material used in the analysis is being processed and integrated on the LingvoDoc platform. The underlying opinion of this study draws on the seminal work by N.I. Tereshkin, as well as other prominent researchers of the Khanty language, and their description of the declension system of the Eastern dialect. The latest field data contains texts and questionnaires allowing researches to examine functioning of the morphological markers and compare results with the findings previously presented in the literature. Such approach entails observations leading to the systematization of the case category in this dialect at its current state of development. The examination of the range of meanings conveyed by the Vakh Khanty case markers at present has made it possible to distribute them into groups of semantic and syntactic case markers. As a result, the study confirmed that the nominal declension system encompasses such case markers as the abessive, ablative, allative, distributive, comitative, comparative, lative, locative, nominative, oblicative, and translative cases. The recent field data confirms the presence of the distributive case within the case paradigm and the status of the comparative element *niŋit* as a postposition. Each case marker of the Vakh Khanty dialect is terminologically specified which is essential for developing a parser for this dialect on the LingvoDoc platform. The development of the parser in the future will speed up the process of processing and analyzing language material and lead to the elimination of errors.

Keywords: Khanty language; Vakh dialect; case category; case markers; field data; text corpora; LingvoDoc.

For citation: Vorobeva V.V., Novitskaya I.V. Functional and semantic analysis of the case markers in Vakh Khanty (a case study of the latest field data on LingvoDoc). *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 2, 2024. pp. 199-210. DOI: 10.15514/ISPRAS-2023-35(2)-15.

Acknowledgements. The authors are grateful to the anonymous reviewers for their careful reading of the work. The study was funded by Russian Science Foundation, project no. 20-18-00403 “Digital Description of Uralic Dialects Based on Big Data Analysis”.

Функционально-семантический анализ падежных маркеров ваховского хантыйского языка (на материале аннотированных полевых данных в системе ЛингвоДок)

[1,2] *В.В. Воробьева,* ORCID: 0000-0002-8729-0375 <*vorobeva@tpu.ru*>
[3] *И.В. Новицкая,* ORCID: 0000-0003-1559-8810 <*irno2012@yandex.ru*>

[1] *Институт системного программирования РАН,*
109004, Россия, г. Москва, ул. А. Солженицына, д. 25.
[2] *Национальный исследовательский Томский политехнический университет,*
634050, Россия, Томск, пр. Ленина, д. 30.
[3] *Национальный исследовательский Томский государственный университет,*
634050, Россия, Томск, пр. Ленина, д. 36.

Аннотация. В данной статье мы утверждаем, что описание категории именного склонения в ваховском хантыйском языке нуждается в пересмотре, в связи с наличием последних полевых данных, полученных в 2019 году. Эта задача считается актуальной, поскольку результаты исследования предоставят ученым необходимые данные для разработки единой системы обозначений падежных маркеров, которые будут использоваться при обработке текстового материала для размещения на платформе ЛингвоДок. Система склонения анализируемого диалекта восточных ханты неоднородна и относительно обширна. Дискуссия о синтетических средствах выражения семантических отношений в языковых единицах касается терминологии падежных маркеров, их количества, морфемного статуса и функциональных особенностей. Целью данной работы является уточнение функциональных и семантических аспектов падежных маркеров ваховских ханты на основе недавно полученного полевого материала. Эта цель достигается с помощью инструментов, доступных на платформе ЛингвоДок. Последние полевые данные по ваховскому хантыйскому диалекту были собраны в селе Корлики в 2019 году. Данные объемом более 6000 слов размещены на платформе документирования исчезающих языков ЛингвоДок. Часть материала, используемая при анализе, постепенно обрабатывается и интегрируется на платформу ЛингвоДок. В основе исследования лежат работы Н.И. Терёшкина (1961) и других исследователей (Я. Гуя, Л. Хонти и др.) хантыйского языка, которые ранее описывали систему склонения ваховского диалекта. Последние полевые данные содержат тексты и анкеты, позволяющие исследователям изучить функционирование морфологических маркеров и сравнить результаты с данными, ранее представленными в литературе. Такой подход предполагает наблюдение, ведущее к систематизации падежной категории в этом диалекте на современном этапе его развития. Изучение спектра значений, передаваемых падежными маркерами, в настоящее время позволило распределить их на группы семантических и синтаксических падежных маркеров. В результате исследования подтверждено, что система именного склонения включает в себя такие падежные маркеры, как аблативный, абессивный, аллативный, лативный, локативный, комитативный, дистрибутивный, обликативный и транслативный. Последние полевые данные подтверждают наличие дистрибутивного падежа в системе и статус сравнительного элемента *niŋit* как послелога. Каждый падежный маркер ваховско-хантыйского диалекта терминологически конкретизирован, что необходимо для разработки парсера этого диалекта. Развитие парсера в будущем ускорит процесс обработки и анализа языкового материала и приведет к устранению ошибок при разметке текстов.

Ключевые слова: хантыйский язык; ваховский диалект; категория падежа; падежные маркеры; полевые данные; корпуса текстов; ЛингвоДок.

Для цитирования: Воробьева В.В., Новицкая И.В. Функционально-семантический анализ падежных маркеров ваховского хантыйского языка (на материале аннотированных полевых данных в системе ЛингоДок). Труды ИСП РАН, том 36, вып. 2, 2024 г., стр. 199–210 (на английском языке). DOI: 10.15514/ISPRAS–2024–36(2)–15.

Благодарности. Авторы статьи выражают благодарность анонимным рецензентам за тщательное и детальное прочтение работы. Исследование выполнено при поддержке РНФ, проект № 20-18-00403 «Цифровое описание диалектов уральских языков на основании анализа больших данных».

1. Introduction

In this paper we argue that the process of digital processing of the recently obtained field data on the Vakh Khanty dialect requires a review of its nominal declension system with an ensuing prospect of developing a valid parser for this idiom.

"Morphological case on nominals is a common device to express the syntactic and semantic relationships between clausal constituents" [1]. Some scientists distinguish synthetic (traditional) and analytical cases [2-3]. Regardless of the way of expression, the case is a meaningful category, and its expression is a secondary point. In this study we focus only on the synthetic form of expressing syntactic and semantic relations in sentences. The subsequent analysis of case markers in the Vakh-Khanty dialect is based on the concept of syntactic and semantic cases [4: 325, 5, 6: 122].

The category of case in the Khanty language has always aroused interest among researchers because the number of constituents that form the nominal declension system varies considerably among the group of dialects integrated by a geographical principle. In the Northern dialects of Khanty, for example, the case system includes three markers, but the system constituents may be different. For example, the Kazym dialect possesses the nominative, lative and locative case markers, while in the Priural dialect these markers are for the nominative, locative and translative cases. In the Southern dialects, which have now lost their vitality, researchers distinguish from five to six case markers. In the Eastern dialects, the nominal declension system may include from seven to eleven case markers. The diverging views on the number of case markers in Eastern Khanty reflect different linguists' opinions about the status of some morphemes encoding case semantics, therefore, the same marker in grammatical essays can be termed as a case marker, and as a particle, and as a postposition [7-16].

Similar divergence of opinions is found with regard to the terms used to distinguish some of the identified constituents of the case category. Since most of the case markers are multifunctional, researches face a challenging task of either selecting the most suitable term for a multifunctional case marker focusing on one of its predominant functions, or of using a dual term that reflects multiple functions of a single case marker. Thus, in his Vakh Khanty grammatical essay, N. I. Tereshkin argues that there are markers of such cases that he personally termed as *"napravitel'no-tselevoi"* (rus. lit. *"directional-target"*), *"tvoritel'no-ob'ektnyi"* (rus. lit. *"instrumental-objective"*), *"tvoritel'no-sovmestnyi"* (rus. lit. *"instrumental-comitative"*), *"otlozhitel'no-prodol'nyi"* (rus. lit. *"depositional-transitive"*) [13: 42–55].

It can be concluded that the constituents of the case category in the Vakh Khanty dialect, as they are presented in the scientific literature, are open to multiple interpretations resulting in ambiguous conclusions about representatives of individual cases. This fact complicates the process of digitizing the field data with the tools of the LingvoDoc platform [17].

The purpose of this paper is to clarify the functional and semantic aspects of the Vakh Khanty case markers drawing on the recently obtained field material. This objective is achieved by means of the tools available on the LingvoDoc platform.

While processing a text corpus of Vakh Khanty for posting on the LingvoDoc platform we faced such issues as aspectual, temporal markers and predicative markers, comparative constructions, words that have not yet been accepted by dictionaries. Certain issues we encountered concerned the Vakh Khanty case markers, namely, their terminology, number and functions. Digital processing of the data was grounded in the findings presented in the grammar by N. I. Tereshkin, a student of V. Steinitz. The researcher collected materials for his monograph during his scientific expeditions along the rivers Vakh, Vasyugan, Tromyegan and Salym that he made in the second half of the 20th century. His Khanty students at the Institute of the Peoples of the North in Leningrad [13: 5] also

contributed greatly with their speech samples. His monograph "Essays on dialects of the Khanty language" (1961) describes Vakh Khanty reliably and quite fully, and is, therefore, regarded as a starting point in the process of annotating the corpus. However, the issues faced while processing the field data remain unresolved. We also discovered that some grammatical phenomena were not described in full or were not mentioned at all in that grammar handbook, for example, the absessive case marker and some case functions, which will be discussed below. We also incorporated the works by foreign specialists of Vakh Khanty J. Gulya [9] and L. Honti [10] but in most cases the issues could not be solved and the task became even more complicated, since the scientists interpreted, labelled and described case functions differently.

All these have prompted us to write this article in which we aim to verify the constituents of the Vakh Khanty nominal case system. Standardization of the case marker labels is believed to be an important stage in developing a reliable and viable Vakh Khanty parser in the nearest future, and it will allow case morphemes to be glossed universally, in a standardized way. Besides, it will help errors to be excluded that, given the human factor, are inevitable in the manual processing of a large volume of data. It will also help achieve a completely different speed of text processing.

Research materials and methods

The study was conducted on the field data that was collected through the communication with speakers of the Vakh dialect by one of the authors in the village of Korliki (KhMAO-Yugra) in 2019. Language data was collected using traditional field research methods (survey, interview, experiment, observation, recording and description using various technical means and programs). Part of the collected material is digitized and presented on the LingvoDoc platform, namely: a fairy tale "A mouse's song", a text about deer, questionnaire "Case category" P. I, P. II, questionnaire "Tense, Mood, Aspect" P. I, II, III, IV" [AFM]. Some data that was also used in the analysis has not yet been digitally processed, but will soon appear on the same platform, these include: a fairy tale "Two birds - sister and brother" and a text about deer [AFM].

The analysis of the obtained language data on the Vakh dialect was carried out using descriptive-comparative and structural research methods, as well as corpus analysis methods (diversified search, compilation of contexts and concordances, counting, analysis of the text meta-marking).

2. The cases of the Vakh Khanty dialect

In this study we use the following terminology for case markers: the abessive, ablative, allative, distributive, comitative, comparative, lative, locative, nominative, oblique, and translative cases. Two of the above case markers, namely, the nominative and locative, are involved in grammatical (syntactic) relations. The remaining markers, including the locative in one of its functions, specialize in the function of marking semantic relations. These are the latter that are the object of study in this article.

The case markers under analysis are divided into three subgroups in accordance with the conceptual domain their semantics belongs to. The first subgroup includes case markers that express spatial relations, namely, markers of the ablative, allative, lative and locative cases. The second subgroup includes markers of the absessive and comitative cases, which encode the joint or disjunctive nature of the designated action. The third subgroup includes distributive, objective and translative cases markers, which are used to indicate adverbial or 'indirect' relations.

Let us now analyze the spectrum of meanings encoded by each marker.

The spatial case markers from the first group, namely the ablative, allative, lative, and locative markers, encode the meanings of direction of movement, static position, and position in time. Most of the case morphemes – the ablative, lative and locative markers – belonging to the analyzed group are multifunctional. The only case marker from this group – the allative marker – acts as a monofunctional one.

According to A. L. Malchukov's concept [18], spatial cases participate in the representation of basic semantic roles, namely, the semantic roles of Place, Path, Source (a starting point of movement), Goal (the end point of movement). The field material of Vakh Khanty dialect does not support a clear division of spatial cases according to these semantic roles. In this dialect of Khanty, each role can be expressed by one or two spatial cases. Thus, to represent the role of Place, the locative case is used, to implement the role of Path, the ablative case is used, which can also be used to represent the role of Source. The lative and allative cases are focused on representing the role of the Goal. Let us now dwell separately on each marker from the group of spatial cases.

2.1 Cases of spatial meaning

The ablative case marker ***-oɣ/-öɣ*** encodes the meaning of a starting point of some movement directed away from something or someone. It also marks movement or an action, and a concept or an event derived from an object. Let us illustrate these meanings with examples (1–6):

(1) *lüɣ-ən* *män-t* *kǒr-äm-**öɣ*** *pŏr-ɣas*
3SG-LOC 1SG-ACC foot-POSS.1SG-**ABL** bite-PST2.SBJ.3SG
'She bit me in the foot/leg.'

(2) *lüɣ-ən* *män-t* *puɣləm-am-**oɣ*** *öɣmilə-s*
3SG-LOC 1SG-ACC cheek-POSS.1SG-**ABL** kiss-PST1.SBJ.3SG
'He kissed me on the cheek.'

(3) *lin* *paj-**oɣ*** *rɻt* *wer-s-äɣən*
3DU aspen-**ABL** boat make-PST1-SBJ.3DU
'They made a boat out of aspen.'

(4) nüŋ paḳi wer-ɣäs-ən tŏnəɣ-**oɣ**
2SG doll make-PST2-SBJ.2SG birch.bark-**ABL**
'You made dolls out of birch bark?'

(5) *min* *wor* *ont-**oɣ*** *wɻɣəl-s-əmən*
1DU woods inside-**ABL** come.out-PST1- SBJ.1DU
'We came out of the woods.'

The ablative marker also encodes the prolative meaning of movement, expressing the meaning of the path, passage, or roadside "along, through" along which the action unfolds:

(6) *kiriw* *waɣ-**oɣ*** *mən-wəl*
motorboat Vakh-**ABL** sail-NPST.SBJ.3SG
'A motorboat sails along the Vakh river.'

(7) *lin* *ḷök-**öɣ*** *pŏŋl-a* *mən-s-äɣən*
3DU road-**ABL** side-LAT walk-PST1-SBJ.3DU
'They were walking along the side of the road.'

The allative case marker ***-pa/-pä*** is monofunctional and it usually indicates the final point of movement in the Vakh Khanty dialect. According to the field materials, the allative marker is regularly found in the speech samples of the Vakh Khanty, although in some cases it reveals its functional similarity to the marker of the lative case. This is illustrated by the following examples (8–9):

(8) *lüɣ* *lawka-j-**pa*** *jəl-äs* *päni* *n'än'* *wä-s*
3SG store-EP-**ALL** go-PST1.SBJ.3SG and bread take-PST1.SBJ.3SG
'He went to the store and bought bread.'

(9) *məčəɣ* *təɣɻ-l-**pa*** *mənä-s*
farmland place-POSS.3SG-**ALL** go-PST1.SBJ.3SG
'He went to the farmland.'

The marker of **the lative case *-a/-ä*** is multifunctional. One of its functions is to encode the final point of the movement trajectory, i.e. the meaning similar to that of the allative case. Let us illustrate these similar meanings with examples:

(10) *min wor* *ont-oɣ* *lar-**a*** *wi̯ɣəl-s-əmən*
1DU grove inside-ABL swamp-**LAT** come.out-PST1-SBJ.1DU
'We came out of the grove to the swamp.'

(11) *wəlta* *təɣi̯-l-**a*** *mən-il-wəl*
live-INF place-POSS.3SG-**LAT** go-MULT-NPST.SBJ.3SG
'He is going to the farmland.'

Another function is to mark the lative meaning, which is to indicate the purpose to which the action is directed.

(12) *män-nə* *köɣ-**nä**-ti* *jöɣ-ɣäs-i* *mŏ̆ŋkäm-**ä***
1SG-LOC rock-**COM**-PRTC throw-PST2-PASS.3SG snake-**LAT**
'I threw a rock at the snake.'

(13) *lüɣ* *n'əŋiḳjji̯-t-**a-tį*** *litot* *tu-ɣal*
3SG kid-PL-**LAT-PRTC** food take.away-PST3.SBJ.3SG
'She took food away to the kids.'

The lative case is often used to mark nouns that form semantically fused combinations with verbs, for example, to hunt, to matchmake, etc. Thus, one of the most frequently used phrases among speakers of Vakh Khanty 'to hunt' consists of the verb *kəṇčta* 'to search' and the noun *wajəɣ* 'beast' in the lative case. The meaning of 'to matchmake' is conveyed by the verb *ärəɣtätä* 'to promise' and a noun in the lative case. Let us illustrate the use of these phrases with examples (14–16):

(14) *käčəŋ* *ḳotl* *wajɣ-**a*** *kənč̣-i̯l-wəl*
every day beast-**LAT** search-MULT-NPST.SBJ.3SG
'He hunts every day.'

(15) *änä,* *män-nə* *səwəs* *iki-j-**ä*** *ärəɣ-tə-s-uj-əm*
sister 1SG-LOC sevsiki old.man-**LAT** promise-TR-PST1-PASS-1SG
'Sister, I got you married to Sevsiki.'

The marker of the lative case *-a/-ä* can also be used to encode the recipient:

(16) *pəɣ-ali* *äni-l-**ä*** *t'uti* *jəɣ-iɣən…*
boy-DIM sister-POSS.3SG-**LAT** this say-NST.SBJ.3SG
'The boy says this to his sister.'

As it follows from the description above, the lative case marker demonstrates a greater variety of functional applications. However, the most multifunctional case marker in the Vakh Khanty dialect appears to be **the locative case** marker ***-nə/-nə̈*** which is analyzed in detail in a separate article [15]. A new grammatical function of the locative marker has been identified which was not described previously in that work [15]. In this article, we give only few examples in which the locative case marker functions as a representative of the semantic role of Place and marks the location of an object in space:

(17) *köčəɣ* *satəw-**ən**-əḳi̯*
knife sheath-**LOC**-PRED
'The knife is in sheath.'

(18) *sar-ɣən* *jəŋk-**ə̈n**-ə̈ki-ɣən*
pike-DU water-**LOC**-PRED-3DU
'The pike are in the water.'

2.2 Cases of compatibility or deprivation meaning

The second subgroup of case markers includes morphemes of the abessive and comitative cases, which represent the meanings of compatibility or deprivation correspondingly. **The comitative case** marker ***-na/-nä*** expresses a comitative meaning, i.e. it describes multiple (two or more) participants of the situation fulfilling the same semantic role.

(19) *pəɣ-ali-nə* *öɣ-äli-l-**nä**-ti* *kömən* *jaŋ̣ka-l-əɣən*
boy-DIM-LOC girl-DIM-POSS.3SG-**COM**-PRTC outside play-NPST-SBJ.3DU
'A boy and a girl are playing outside.'

(20) *vasya* *marina-**na**-tį* *mən-ɣäs*
Vasya Marina-**COM**-PRTC go-PST2.SBJ.3SG
'Vasya and Marina have left.'

The comitative case can convey an instrumental meaning by marking the subject which names the instrument and means of action:

(21) *iɣ-nə* *köt-əl-**nä**-ti* *joɣįm-ɣal-į* *wän'əm-əl-öɣ*
bear-LOC paw-POSS.3SG-**COM**-PRTC slap-PST3-PASS.3SG face-POSS.3SG-ABL
'The bear slapped him in the face with his paw.'

(22) *kolya* *rįt* *laijm-**na**-tį,* *suɣəl-**na**-tį,* *kįr-**na**-tį* *wer-ɣäs*
Kolya oblas axe-**COM**-PRTC scraper.axe-**COM**-PRTC side.axe-COM-PRTC make- PST2.SBJ.3SG
'Kolya hollowed out the oblas with an axe, a scaper axe, a side axe.'

(23) *tim-tom* *pəjlaŋ-əl-**na*** *čoɣət* *jerimil-l-ätə*
this-that wing-POSS.3SG-**COM** snow streak-NPST-SBJ.3SG:OBJ.SG
'[It] marks the snow with this or that wing.'

The comitative case also expresses a concomitant object:

(24) *mara* *jŏ-ɣäs* *ul-**na**-tį*
Mara come-PST2.SBJ.3SG berry-**COM**-PRTC
'Mara came with berry.'

In the Vakh Khanty dialect, the comitative marker *-na/-nä* is also used to denote means of transportation, which can be considered the fourth function of the marker.

(25) *iki* *ku-j-t* *wajəɣ* *kənč-il-tä* *weli-t-**nä*** *jəl-ilə-wəl-ət*
old man-PL beast search-MULT-INF reindeer-PL-**COM** go-MULT-NPST-SBJ.3PL
'Old men ride reindeer to hunt.'

(26) *lüɣ rįt-**na*** *mən-ɣäs*
3SG boat-**COM** идти-PST2.SBJ.3SG
'He left by boat.'

(27) *t'ukumənta,* *lüɣ* *mən-ɣäs* *likər-nä-ti*
turn.out 3SG go-PST2 sledge-COM-PRTC
'It turns out that he left on a sledge.'

Field data show that the comitative marker attaches the particle *-(ti)/-(ti)* more often than other case morphemes, see examples (19–22, 24, 27).

Semantically, the comitative marker *-na/-nä* is opposed by the monofunctional **abessive case** marker ***-ləɣ/-läɣ***.

(28) *mä* *ämp-**läɣ*** *kuɣa-l-em*

1SG dog-**ABS** walk-NPST-SBJ.1SG
'I am walking without a dog.'

(29) *läɣ pučkän-**läɣ** jəl-il-ɣäs-ət*
3PL gun-**ABS** go-MULT-PST2-SBJ.3PL
'They went without weapons.'

2.3 Cases of "oblique" meaning

The third subgroup of case morphemes includes markers of the oblique, translative and distributive cases. These markers are grouped on the basis of the circumstantial (adverbial) or "indirect" semantics they convey, similar to the case meanings that are found only in languages with case paradigms including multiple members. The translative and distributive cases are found in other languages of the Finno-Ugric group, for example, the translative case is present in Finnish and Hungarian, but the distributive case is only found in Hungarian. The oblique case is a unique feature of the eastern dialects of the Khanty language.

The marker of **the oblique case *-(t)ə/-(t)ӓ*** conveys the meanings that are typically encoded by the instrumental case in Russian, however, it still differs in its ability to mark only the inanimate object of a transitive verb in the subject or object conjugation, active or passive. In the Russian literary translation, we use the accusative case (i.e., "Give me water"), although a literal translation implies the instrumental case "by water" (see examples 30-33):

(30) *män-t äčiɣ jäŋk-**ӓ** mäj-itäɣ*
1SG-ACC cold water-**OBL** give-IMP.SBJ.PL.
'Give me some cold water.'

(31) *lüɣ-ӓn män-t pučkän-**ӓ** mäj-entil-wəl*
3SG-LOC 1SG-ACC rifle-**OBL** give-MULT-NPST.SBJ.3SG
'He gives me a rifle.'

(32) *käčəŋ ḳu män-nä ḳul-**ə** mə-s-i*
every person 1SG-LOC fish-**OBL** give-PST1-PASS.3SG
'I gave everyone a fish.'

(33) *käčəŋ ḳu män-nä läŋki-**tə** pit-s-i*
every person 1SG-LOC squirrel -**OBL** help-PST1-PASS.3SG
'Everyone got a squirrel.'

The marker of **the translative case *-ɣ/-əɣ/-äɣ*** is used to indicate a change in form or state. Typically, a noun with a translative marker is used with the verbs *jəta* "to become" and *sälimtätä* "to transform". This is illustrated by an example:

(34) *lüɣ jol-ta ḳu-j-**ɣ** jə-ɣäs*
3SG shamanize-INF man-**TRL** become-PST2.SBJ.3SG
'He became a shaman.'

The distributive case marker *-(ə)təl/-(ä)täl* is not identified by all linguists. As a case marker *-(ə)təl/-(ä)täl* is designated only in the Khanty reader by L. Honti [10]. The case marker encodes the meaning "everyone"; it is translated into Russian as "per everyone, per capita" and is not used with all words. Below are examples from the latest field materials that confirm the existence of this marker in the case paradigm of Vakh Khanty (see examples 35-36).

(35) *ejnäm ej n'än' təɣt-ali-**təl** li-s-ät*
все один хлеб кусок-DIM-**DISTR** есть-PST1-SBJ.3PL
'Everyone ate a piece of bread.'

(35) *käčəŋ ḳu kä rɨt-**ətəl** wə-ɣäs*
each person two boat-**DISTR** take-PST2.SBJ.3SG

‘Each took two boats.’

(36) *ejnäm* *ej* *ḳul-**ətəl*** *li-s-ät*
все один рыба-**DISTR** есть-PST1-SBJ.3PL
‘Everyone ate one fish.’

2.4 The Status of the element niŋit and the summarry of the nominal case markers of modern Vakh Khanty in a table

The comparative element *niŋit*, which is singled out by J. Guya as an independent case marker [9], does not confirm its status as a case marker drawing on the results of the analysis of field data. It is used to denote the comparison of one object with another and functions as an autonomous auxiliary element - a postposition, which is confirmed by the violation of the vowel harmony of the element *niŋit* with a preceding word. N. I. Tereshkin also defines the element *niŋit* as a postposition, for example (37):

(37) *körl'iki* *larjyak* *niŋit* *jəm-iki*
Korliki Larjyak compared.to good-PRED
‘Korliki is better than Larjyak.’

Table 1 shows markers of all cases of the nominal declension system, used to mark semantic relations in the present-day Vakh Khanty dialect.

Table 1. A set of markers of the semantic cases in Vakh Khanty based on the latest field data

	Semantic groups of modern Vakh Khanty maarkers	
№	***Case***	***Case marker***
Group 1. Cases of spatial meaning		
1	Ablative (ABL)	oɣ/öɣ
2	Allative (ALL)	(a)pa/(ä)pä
3	Lative (LAT)	a/ä
4	Locative (LOC)	nə/nə̈
Group 2. Cases of compatibility or deprivation meaning		
5	Abessive (ABS)	ləɣ/ləɣ
6	Comitative (COM)	na/nä
Group 3. Cases “oblique” meaning		
7	Distributive (DISTR)	-(ə)təl/-(ə̈)tə̈l
8	Oblique (OBL)	(t)ə/(t)ə̈
9	Translative (TRS)	ɣ/əɣ/ə̈ɣ (< ḳə/kə̈)

Table 1 shows a summary of the unified data on case markers of Vakh Khanty, compiled on the basis of this study using materials from the latest expeditions of the current century. The table lists 9 case markers in unified terminological notations. We have proposed a one-component case terminology, where next to each case term there is also a variant of its designation for glossing in order to standardize glossing of texts in the Vakh dialect. Unification of notations is necessary for universal markup of texts, so that when using the tools of the LingvoDoc platform, an entire sample of combinations with case markers available in the corpus is selected.

3. Conclusion

Field data of the present-day Vakh Khanty dialect indicates that its nominal declension system includes ten case forms, nine of which specialize in encoding semantic relations. The analysis of the functional features of case markers was carried out drawing on a digitized corpus of linguistic data

of the Vakh Khanty dialect, available on the LingvoDoc platform. The corpus contains the latest language data on this dialect, collected in the village of Korliki, Nizhnevartovsk region, Khanty-Mansi Autonomous Okrug in 2019. The study results made it possible to confirm the stability of the system of nominal declension of the Vakh Khanty dialect, to clarify the functional uniqueness of each case marker and to identify the grounds for their preferential terminological designation with the following labels: 1) ablative, allative, lative, locative; 2) abessive, comitative; 3) distributive, oblique and translative. The comparative element *niŋit* did not confirm its status of the case marker. This study proposes a unified terminology of case markers for the development of parsers and markup of text corpora in Vakh Khanty.

In ensuing studies analyses of the analytical ways of expressing syntactic relations between objects will be conducted, namely, constructions with postpositions and postpositional nouns which are used to express spatial and temporal meanings along with synthetic cases.

4. List of abbreviations

ABL – ablative
ABS – abessive
ACC – accusative
ALL – allative
COM – comitative
DIM – diminutive
DISTR – distributive
DU – dual
EP – epenthetic
IMP – imperative
INF – infinitive
LAT – lative
LOC – locative
MULT – multiplicative
NPST – non-past
NST – suffixless
OBJ – objective
OBL – oblique
PASS – passive
PL – plural
POSS – possessive
PRED – predicative
PRTC – particle
PST1 past in *-s*
PST2 – past in *-ɣas/-ɣäs*
PST3 – past in *-ɣal/-ɣäl*
SBJ – subject
SG – singular
TRL – translative

5. Field materials of the authors

AFM – Field materials of the authors. Polevye materialy avtora. Jekspedicija v p. Korliki HMAO – Jugry, leto 2019 g. [Author's field materials. Expedition to Khanty-Mansi Autonomous Okrug-Yugra. Summer 2019”] (speakers: T. F. Katkaleva, b. 1944, E. A. Kunina, b. 1939, Zh. A. Khokhlyankina, b. 1967).

References

[1]. O.A. Iggesen, Number of cases. WALS Online. 2013. Eds. by Dryer Matthew S., Hasplmath Martin. Available at: https://doi.org/10.5281/zenodo.7385533 (Accessed April, 2024).

[2]. C. J. Fillmore, The case for case. Universals in linguistic theory. Eds. by E. Bach, R. T. Harms. New York, NY: Holt, Rinehart and Winston, 1968. Pp. 1–88.

[3]. A. A. Zaliznyak, On the understanding of the term “case” in linguistic descriptions. Problemy grammaticheskogo modelirovanija [Problems of grammatical modeling]. Ed. by A. A. Zaliznyak. Moscow: Nauka Publ., 1973. Pp. 53–87. (А.А. Зализняк. О понимании термина «падеж» в лингвистическом описании. Проблемы грамматического моделирования. М.: Наука, 1973. С. 53-87)

[4]. I.A. Melchuk, Course of general morphology. Vol. 2. Moscow; Vienna: Languages of Russian culture, Vienna Slavic Almanac, Progress Publishing Group, 1998. 544 p. (И.А. Мельчук, Курс общей

морфологии. Т. 2. Москва – Вена: Языки русской культуры. Венский славистический альманах. Из-во: Прогресс, 1998. 544 с.)
[5]. V. A. Plungyan, Studies in Grammar, is. 2: Grammarization of spatial meanings in the languages of the world. M.: Russian dictionaries Publ., 2002. Pp. 57–99. (В. А. Плунгян, Исследования по теории грамматики, вып. 2: Грамматикализация пространственных наречий в языках мира. М.: Русские словари, 2002. С. 57–99).
[6]. V.A. Plungyan, General morphology. Introduction to the issue. Moscow: Editorial URSS Publ., 2003. 384 p. (В. А. Плунгян. Общая морфология. Введение в проблематику. Москва: Эдиториал УРСС, 2003. 384 с.)
[7]. M. Chepregi, Surgut dialect of the Khanty language. Khanty-Mansiysk, Printed World of Khanty-Mansiysk, 2017. 275 p. (М. Чепреги, Сургутский диалект хантыйского языка. Ханты-Мансийск: ООО «Печатный мир г. Ханты-Мансийск», 2017. 275 с.)
[8]. A.Yu. Filchenko, A Grammar of Eastern Khanty. Houston, Texas, 2007. 615 p. Available at: https://theswissbay.ch/pdf/Books/Linguistics/Mega%20linguistics%20pack/Uralic/Khanty%2C%20A%20Grammar%20of%20%28Filchenko%29.pdf (Accessed April, 2024)
[9]. J. Gulya, Eastern Ostyak Chrestomathy. Bloomington: The Hague, 1966. 207 p.
[10]. L. Honti, Chrestomathia Ostiacica. Budapest: Tankonyvkiado, 1986. 432 p.
[11]. K. F. Karjalainen, Grammatikalische Aufzeichnungen aus Ostjakischen Mundarten. Bearbeitet und herausgegeben von Edith Vertes. Helsenki: Suomalais-Ugrilaisen Seura, 1964. 234 p.
[12]. D. F. Mymrina, Case category in the Khanty language (comparative aspect). Tomsk, 2006. 145 p. (Д. Ф. Мымрина, Категория падежа в хантыйском языке (сопоставительный аспект). Автореферат диссертации на соискание ученой степени к.ф.н. Томск, 2006)
[13]. N. I. Tereshkin, Essays on dialects of the Khanty language (The Vakh dialect), L.: Nauka, 1961. (Н.И. Терёшкин, Очерки диалектов хантыйского языка (ваховский диалект). Л.: Наука, 1961.)
[14]. V. Vorobeva, I. Novitskaya, Inflectional morphology of nouns in Eastern Khanty (Vakh, Vasyugan, Surgut, Salym). Ural-Altaic Studies, vol. 38, 2020, pp. 33-70.
[15]. V.V. Vorobeva, I.V. Novitskaya, Z.V. Fedorinova, Syntactic cases of nouns in Vakh Khanty (based on the latest field data of LingvoDok / Kazan Science, vol 9, 2023, pp. 157-159. (В.В. Воробьева, И.В. Новицкая, З.В. Федоринова, Синтаксические падежи имени существительного в вахавском хантыйском языке (на материале современных полевых данных платформы LingvoDok. Казанская наука, том 9, 2023, стр. 157-159).
[16]. V.V., Vorobeva, I.V. Novitskaya, Functional characteristics of the nominal case markers in Vakh Khanty (on the basis of the recent field language data of the LingvoDoc platform). Trudy ISP RAN/Proc. ISP RAS, vol. 35, issue 4, 2023. pp. 165-176 (В.В Воробьева, И.В. Новицкая, Функциональные особенности падежных показателей в ваховском хантыйском языке (на материале базы современных полевых данных на платформе ЛингвоДок). Труды ИСП РАН, том 35, вып. 4, 2023 г., стр. 165–176).
[17]. Vakh Khanty Corpus. Lingvodok Platform. Available at: http:// https://lingvodoc.ispras.ru/dictionary/3568/147/perspective/3568/151/view. (Accessed 24 April, 2024). (Корпус ваховского диалекта хантыйского языка. Доступен на: http:// https://lingvodoc.ispras.ru/dictionary/3568/147/perspective/3568/151/view. (Accessed 24 April, 2024)
[18]. A. L. Malchukov, Syntax of the Even language: structural, semantic, communicative aspects. St. Petersburg, Nauka, 2008. 425 p. (Мальчуков А. Л. Синтаксис эвенского языка: структурные, семантические, коммуникативные аспекты. СПб, Наука, 2008. 425 с.)

Информация об авторах / Information about authors

Виктория Владимировна ВОРОБЬЕВА – кандидат филологических наук, доцент отделения иностранных языков Школы общественных наук Национального Исследовательского Томского политехнического университета с 1999 года. В настоящее время также работает старшим научным сотрудником в лаборатории Лингвистические платформы Института системного программирования РАН. Область научных интересов: сравнительно-типологическое языкознание, уральская языковая семья, корпусная лингвистика, документация и цифровая обработка исчезающих языков.

Victoria Vladimirovna VOROBEVA – Cand. Sci. (Philology), Associate Professor of Department of Foreign Languages, School of Social Sciences of National Research Tomsk Polytechnic University since 1999. At the present time she is a researcher at the Linguistic Platforms Laboratory of Information Systems Department as well. Research interests: comparative typological linguistics, Uralic family of languages, corpus linguistics, documentation and digital processing of endangered languages.

Ирина Владимировна НОВИЦКАЯ – доктор филологических наук, доцент, профессор кафедры английской филологии Национального исследовательского Томского государственного университета с 1997 года. Область научных интересов: морфология и семантика древнегерманских языков, типологическое языкознание, сравнительно-сопоставительные исследования, морфосинтаксис обско-угорских языков, прикладная лингвистика.

Irina Vladimirovna NOVITSKAYA – Dr. Sci. (Philology), professor at the Department of English Philology, at National Research Tomsk State University. Research interests: morphology and semantics of the Old Germanic languages, typological studies, comparative studies, morpho-syntactical properties of Khanty lexical units, applied linguistics.