The apriori method in the collection of significant values of pharmaceutical data

Nataliya Maslova[1], Olena Liubymenko[1], Olha Polovynka[2] and Yaroslaw Dorogyy[1]

[1] *Donetsk National Technical University, str. Potebni, 56, Lutsk, 43018, Ukraine*
[2] *National Technical University "Kharkov Polytechnic Institute", ul. Kyrpychyova, 2, Kharkov, 61002, Ukraine 1*

Abstract

The paper demonstrates the relevance and necessity of seeking efficient algorithms for large-scale data applications. A review of existing approaches and methods for analyzing unstructured data, often found in pharmaceutical data and medical research results, has been conducted, providing a description of the characteristics of such data. It is indicated that Data Mining methods, including the Apriori algorithm, are applicable for discovering hidden patterns in large volumes of data. Modern implementation tools of the Apriori algorithm for parallel and distributed processing on platforms such as MPI, OpenMP, Hadoop, OpenCL, Spark, and Apache Flink are discussed. The specifics of implementing the Apriori algorithm on each platform are described. Theoretical recommendations are provided, and the results of applying the platforms in terms of the complexity (required workload) and time to obtain results are forecasted. A computational experiment is conducted, which overall confirms the theoretical conclusions presented in the concluding sections of the paper and outlines avenues for further research, envisaging deeper experimentation with further application of parallel and distributed versions of association search algorithms in various fields, including medical and pharmacological domains.

Keywords

Apriori parallel algorithm , Big Data, pharmaceutical logistics

1. Introduction

The proliferation of sources and volumes of information, its diversity, and distributed storage have led to the necessity of reviewing not only data collection and storage technologies, but also methods of analysis. Data Mining allows for the generation of new ideas and understandings of the nature and interrelationships of objects or phenomena. Typical data analysis tasks include classification, clustering, association rule mining, anomaly detection, and others.

In this context, the Apriori method serves as an important tool for analyzing big data. The Apriori method has long been recognized as a valuable tool for intelligent data analysis, especially for discovering association rules in large datasets. However, the constantly increasing volume and complexity of data, as well as the emergence of new data processing platforms, pose significant challenges for traditional analysis methods. Apriori is used to discover association rules among data elements, enabling the identification of interesting relationships in large datasets that may not be visible using conventional analysis methods. This approach facilitates drawing conclusions and making decisions based on the analysis of large volumes of data with significant efficiency and reliability.

One of the main problems associated with the constant growth of data is the complexity of processing and analyzing large datasets due to the limited performance of traditional methods. Various methodological approaches exist to address this problem, including parallel computation, distributed data processing, and the use of specialized algorithms. These approaches enable the efficient distribution of tasks and optimization of data processing for large volumes.

The paper discusses approaches and methods for the analysis of unstructured data. It demonstrates how Data Mining methods and the Apriori algorithm can be used to uncover hidden relationships in large unstructured datasets, particularly in pharmaceutical logistics. A comparative analysis of the characteristics of Apriori algorithm variants used for association rule mining is conducted, and a

CMIS-2024: Seventh International Workshop on Computer Modeling and Intelligent Systems, May 3, 2024, Zaporizhzhia, Ukraine

nataliia.maslova@donntu.edu.ua (N. Maslova); olena.liubymenko@donntu.edu.ua (O. Liubymenko); olga.polovinka1@gmail.com (O. Polovynka); yaroslav.dorohyi@donntu.edu.ua (Y. Dorogyy)

0000-0002-9078-0973 (N. Maslova); 0000-0002-5935-6891 (O. Liubymenko); 0000-0002-0575-4587 (O. Polovynka); 0000-0003-3848-9852 (Y. Dorogyy)

© 2024 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

CEUR-WS.org/Vol-3702/paper39.pdf

modification of the Apriori algorithm for processing unstructured distributed data using parallel methods is proposed. Results of application on various platforms, both existing and proposed variants of association rule mining algorithms, are presented.

In this context, the scientific novelty of the research lies in organizing the analysis of information platforms, with an emphasis on parallel and distributed processing of unstructured data, where the implementation of deep analysis algorithms, including the Apriori algorithm, will be most efficient in terms of speed and technological implementation during the processing of medical and pharmaceutical data.

1.1. Problems of Analyzing Large Volumes of Data, the Purpose of the Study

The concept of «Big Data» encompasses datasets characterized by large volumes, high accumulation rates, and diversity, known as volume, velocity, and variety.

Analytical publications on big data processing [1-3] reveal key trends, contributions, and research developments in this field. Processing large volumes of data requires powerful computational resources, efficient algorithms to ensure analysis speed and effectiveness, and appropriate information storage and processing systems. There is a significant number of algorithms and methods for intelligent (deep) analysis, but each task proves to be unique, so research on developing methods for solving big data processing tasks continues to be relevant.

Special importance lies in algorithms for processing unstructured distributed data, which accumulate in various fields such as finance, logistics, telecommunications, medicine, etc. [4]. Data analysis in these areas needs to be conducted in real-time, placing additional demands on processing speed and system responsiveness. Additionally, increasing data volume may raise risks of privacy and security breaches, hence organizing an adequate level of data protection is crucial.

Neural networks, decision trees, fuzzy models, regression, and clustering analysis methods are widely used for big data analysis. However, these methods are typically used for processing structured data. Processing unstructured data, such as logistic data from pharmaceutical networks or medical research data related to drug selection and interaction analysis, is not possible with similar methods [5].

The choice of processing methods and algorithms determines the research results and effectiveness. The most applicable methods for processing both structured and unstructured large-scale data are Data Mining methods.

Research Object:

Processes of intelligent analysis of unstructured large-scale data.

Research Subject:

Algorithms and methods of intelligent data analysis in solving tasks of discovering hidden patterns.

The research objective is to select platforms for the effective implementation of deep analysis algorithms (Apriori algorithm), considering the possibility of parallel and distributed data processing for large-scale data. Pharmaceutical data applicable in medical research and pharmaceutical logistics data are chosen as input data for conducting experiments.

2. Apriori Method as a Tool for Big Data Analysis

To process large arrays of unstructured data and address the mentioned tasks, it is advisable to use methods for mining associative rules. Based on identified dependencies, rule bases understandable to experts in applied fields can be synthesized.

The Apriori method in Data Mining is an algorithm used for extracting associative rules from a database.

The main goal of the Apriori method is to identify relationships between different elements in sets of structured and unstructured data.

The main Apriori algorithm has several stages.

Candidate generation: Initial itemsets are formed with one item. On the second iteration, two-item sets are analyzed. On the third iteration, three-item sets, and so on, until the algorithm provides informative results (rules are formed).

Trimming non-informative values in the Apriori method is done by determining significance indicators, which may vary depending on the research area and the required reliability of the results. These indicators are support and confidence.

Support measures how often a set of items appears in the dataset.

Candidate filtering: Itemsets that do not meet certain support criteria are removed.

Association rule creation: The remaining itemsets are used to create association rules describing relationships between elements.

A detailed description of the basic algorithm can be found, for example, in [6].

Among the main drawbacks of this algorithm, the authors note the need for repeated scanning of the original dataset and the complexity of determining threshold values for each indicator. Therefore, one of the main directions for improving the efficiency of the Apriori algorithm is the possibility of parallelizing the processing of large datasets, which can significantly reduce scanning time.

The Apriori method is widely used in areas such as recommendation systems, purchase analysis (including pharmaceuticals), advertising strategies, education [7], and other fields where the discovery of relationships between different elements in datasets is important. In medical modeling, the Apriori method is used for automated analysis of medical data in the process of providing medical care, in researching the interaction of drugs taken by the patient [8-10]. Modern data analysis methods are used to examine the patient's basic information, medical history, physical condition, and other important textual characteristics. Additionally, correlations are sought between the factors that caused the illness, its severity degree with the results of the patient's examination. This is done to achieve automatic medical clinical care and establish an accurate diagnosis. In this work, the method is applied to pharmaceutical logistics tasks, critical factors in the implementation of management methods after the end of the life cycle in the pharmaceutical care process described in [11].

We will not dwell on the description and analysis of basic and modified algorithms for mining associative rules. They are sufficiently detailed in the literature and works of the authors [11]. We will provide a brief overview of the application area emphasized in the article preview and proceed to the direct analysis of modern and most promising technologies and implementations in terms of speed and reliability.

3. Methodological Approaches to Addressing Big Data Processing Issues in Pharmaceuticals

Pharmaceutical logistics involves managing the movement of medical supplies and related financial, personnel, and information flows to effectively distribute and reduce overall costs in the supply, production, and sale of pharmaceuticals to achieve the required quality and meet consumer demands.

In the context of pharmaceutical logistics, big data may include information on production, supply, storage, and distribution of medical supplies, clinical trials, electronic medical records of patients, information on pharmaceuticals and their interactions, as well as data on patients' previous prescriptions and treatments.

The use of pharmacy network data is envisaged as input to obtain analytical dependencies necessary to solve pharmaceutical logistics tasks. They are characterized by large volumes of information, a distributed structure, and the availability of specialized basic data collection and processing packages.

Materials [12-14] describe the features of pharmaceutical logistics, list twenty main principles characterizing it, and confirm the complexity and multidimensionality of pharmaceutical data processing tasks. This data can be classified as Big Data - falling into the category of processing large volumes of structured and unstructured data.

In the article [15], the relevance and necessity of searching for effective Data Mining algorithms, including parallel ones, applied to large volumes of pharmaceutical data, are proven. Examples of existing processing packages are provided, their main capabilities and shortcomings are analyzed.

It is noted that the main problems in processing pharmaceutical data are:

Distribution: collecting information from different sources and the need to store it in a specialized repository;

Significant volume (productivity): analyzing information volumes measured in terabytes can take an unacceptable amount of time for analysis and require significant computational power.

Modern technologies supporting IDA algorithms and methods include:

- Data analysis software programs (such as STATISTICA, SAS, SPSS, STATGRAPHICS, etc.);
- Artificial neural networks (BrainMaker, NeuroShell, OWL);

- Case-based reasoning systems (tools), for example, KATE tools, Pattern Recognition Workbench;
- Decision Trees methods (KnowledgeSeeker, See5/C5.0, Clementine, SIPINA, IDIS, etc.);
- Evolutionary programming software tools (NeuroShell);
- Genetic algorithms (GeneHunter);
- Limited enumeration methods (used, for example, in the WizWhy system from WizSoft);
- Data visualization systems in multidimensional space (DataMiner 3D).

Despite this variety, [16] asserts that the main problems of processing most types of data can be solved through parallel (Parallel Data Mining) and/or distributed (Distributed Data Mining) Data Mining algorithms.

4. Tools for Implementing Parallel Versions of the Apriori Algorithm

The main modern instrumental tools for implementing the Apriori algorithm for parallel and distributed processing are platforms listed below.

MPI (Message Passing Interface, 1990): This is a standard for working with parallel programming in large computing clusters. Additionally, the mentioned platform MPI4py is a Python wrapper for the MPI (Message Passing Interface) library and was developed in 2008. This library allows developers to utilize MPI capabilities for parallel programming in the Python language.

OpenMP (Open Multi-Processing, 1997): An open standard for shared-memory parallel programming that provides directives for parallelizing code at the process level.

Hadoop (2006): A framework for processing and analyzing large volumes of data in distributed computing environments.

OpenCL (Open Computing Language, 2008): An open standard for parallel programming on heterogeneous systems such as GPUs, CPUs, and others.

Spark (2010): A framework for parallel data processing working in memory, enabling a wide range of data analysis tasks.

Apache Flink (2014): A distributed system for processing streaming data and batch operations, providing high-speed data processing.

The platforms are arranged by the years of active market entry and allow programmers to effectively utilize parallel computing resources for fast and efficient resolution of various tasks.

Let's consider the peculiarities of implementing the Apriori algorithm on some of the listed platforms.

4.1. MPI platform

The implementation of the Apriori algorithm in a specified environment is most thoroughly described in the work [17]. It discusses modern methods for mining associative rules and discovering knowledge in large volumes of data. Based on a conducted comparative analysis and taking into account the specificity of the subject area, the use of the Apriori algorithm is justified, for which criteria and methods of parallelization during the implementation of the associative rule search procedure are considered.

The results obtained from the research were tested in the form of a parallel implementation of the QuickSort algorithm.

As the hardware base, a parallel cluster of the university with the characteristics listed in Table 1 was used.

Table 1
Technical characteristics of the cluster

Fat servers	Thin servers	Network devices
- Dell Power Edge R710; - 16 cores; - 24 GB RAM; - HDD: 2 x 140 GB - 6 x 1000 GB	- Intel® Xeon®; CPU X5560@2.8 GHz - 16 cores; - 16 GB RAM; - HDD – no;	- Dell Power Connect 5448; - InfiniBand Voltaire Grid Director

- 2 IB ports; - 4 LAN ports	- 1 IB ports; - 2 LAN ports; - 1 IPMI port.

The general scheme of the cluster is shown in Figure 1.

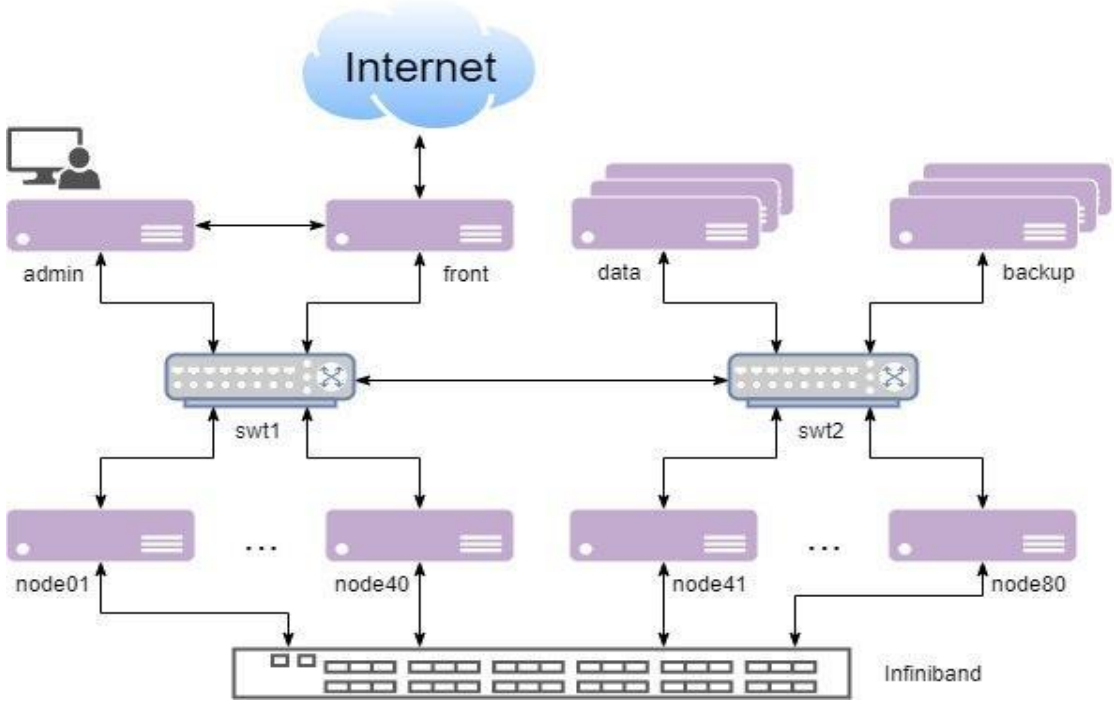

Figure 1: Cluster diagram

The MPI implementation allows for distributing data among cluster nodes to perform separate processing tasks on each. MPI does not provide specific tools for distributing processes. It's the developer's responsibility to handle tasks such as data distribution, processing, execution, and final result collection. Considering these capabilities and aiming to expedite result acquisition, the algorithm was divided into functions executed sequentially:

$$F = F1 + F2 + F3 + F4 + F5 + F6, \quad (1)$$

where the first function (F1) - is reading part of the data from the file;

the second function (F2) - distributes data between processes;

function F3 - parallel scanning and calculation of "itemsets" - a set of elements that are often found in this part of the sample;

function F4 - for data exchange between nodes (initiating data transmission operations between nodes);

function F5 - for result aggregation: Each node receives partial results from other nodes and combines them to create a common list of frequently occurring itemsets.

F6 - function implementation of the main part of the Apriori algorithm (main function), which controls the processes of data reading, distribution among processes, exchange, and initiation of each process and generation of new candidates.

Steps 2-6 are repeated until reaching the maximum (specified) value of itemsets or another stopping condition. After processing is completed, each node aggregates the results and analyzes them to discover associative rules. Calculations were performed with a fixed support count (20%).

In conclusion, it can be summarized that the implementation of the Apriori algorithm using MPI is possible, although algorithmically it proved to be multi-stage. It is also worth noting that despite the computations being performed quite fast, the data reading and distribution block among processes required significant time.

4.2. OpenMP Platform

OpenMP (Open Multi-Processing) is designed for parallel programming on multi-core architectures. It allows programs to be parallelized by adding compiler directives to regular sequential code. Implementation features of the Apriori algorithm for OpenMP include aspects such as memory-level parallelism, distributed work, mechanisms for synchronization of access to shared data between threads, memory allocation, performance optimizations, testing, and validation.

Memory-level parallelism means that OpenMP can simultaneously process data on multiple processor cores. Distributing work among different threads speeds up processing time because different parts of the data can be processed simultaneously.

OpenMP provides mechanisms for synchronizing access to shared data between threads, which is important for preventing conflicts and race conditions during parallel computation.

The implementation of Apriori for OpenMP requires performance optimization by using optimal parameters and selecting the best algorithms for the specific task.

After implementing the algorithm, testing and validation were conducted on real data to ensure the correctness and efficiency of the algorithm.

OpenMP is a standard for parallel programming on multi-core and multi-processor systems with shared memory. It provides a set of compiler directives and a library of functions that allow programs to be parallelized at the directive level. For comparison with MPI results, C and C++ programming languages were applied during the programming stage.

Table 2 shows the algorithm execution time for parallelizing Apriori on a quad-core processor using OpenMP with a fixed support count (20%) and the same algorithm implemented with MPI technology.

Table 2
Algorithm Execution Time for OpenMP and MPI

Array size	MPI 1	OpenMP 1	MPI 2	OpenMP 2	MPI 4	OpenMP 4
1000000	0.16	0.17	0.13	0.13	1.53	0.32
5000000	1.22	0.51	1.11	0.46	1.63	0.43
10000000	3.564	0.95	3.17	0.86	1.51	0.64
15000000	7.008	1.17	4.40	0.98	1.72	0.41
20000000	11.536	1.31	6.71	1.23	1.63	0.72
40000000	40.571	25.23	27.25	23.15	11.88	5.21
80000000	151.998	27.32	81.49	27.64	21.6	5.34

As demonstrated by the experiment, implementing Apriori for OpenMP can significantly improve data processing speed by leveraging parallel programming and optimizing algorithm performance across multiple processor cores.

4.3. Hadoop Platform.

The platform description is provided in [18], [19], [20]. For instance, [19] examines the most well-known modifications of Apriori algorithms for association rule mining. The research presents the performance results of linear algorithms Apriori and Apriori-TID on datasets of various sizes using computers with standard memory capacities. The feasibility of utilizing the AIS algorithm and its parallel implementation on the Hadoop MapReduce framework is justified. A comparison between the linear implementation of the Apriori-TID algorithm and the parallel implementation of the AIS algorithm is provided. Conclusions are drawn regarding the effectiveness of employing parallel algorithms to address the formulated task.

In general, developing an Apriori program for Hadoop on 12 processors requires the utilization of the Hadoop MapReduce framework and distributed computing. This can be achieved by leveraging the following capabilities of Hadoop and its compatible tools (Table 3).

Table 3
Stages of implementation of the Apriori algorithm on Hadoop

№	Stage	Processing tool	Stage description
1	Distribution of data	Hadoop Distributed File System (HDFS)	the data should be distributed among the Hadoop cluster nodes
2	Fragment processing	Mapper	In Mapper, you need to implement logic for selecting subsets of items from each fragment
3	aggregation of results	Reducer	Reducer combines and processes the results obtained from different pieces of data.
4	Apriori	Apriori Process	finding frequently occurring items and generating candidates for the next level. Each iteration is performed on a separate subset of the data
5	Processing	Parallel Processing	parallel processing of different data fragments at the same time on different nodes of the cluster.
6	Load control	Resource Management	Monitoring the correct use of cluster resources to avoid excessive load on individual nodes.

Thus, it can be concluded that implementing the Apriori algorithm for Hadoop is a rather complex process that requires a deep understanding of the Hadoop platform itself, the operation of associated tools (MapReduce, Apache Spark), and their application in software implementation.

Another aspect to consider is that Hadoop necessitates programming in Java for MapReduce task implementation. For other programming languages, such as C++, one needs to utilize the Hadoop Streaming framework, which does not allow for a direct comparison of results obtained for the Hadoop platform with those from experiments on MPI and OpenMP.

Additionally, input files must be prepared in a format understandable by Hadoop, and to obtain results, an output directory must be created.

4.4. OpenCL Platform

The implementation of the Apriori algorithm for OpenCL utilizes the open standard OpenCL for parallel programming on heterogeneous computing devices such as central processing units (CPUs), graphics processing units (GPUs), and others.

The uniqueness of the Apriori implementation for OpenCL lies in harnessing the parallelism capabilities provided by OpenCL to process large volumes of data. The algorithm is divided into parts that can be computed in parallel and processed on different computing devices simultaneously.

With OpenCL, various computing device resources can be efficiently utilized, allowing for computation distribution between CPUs and GPUs based on their capabilities and tasks. This enables significant improvements in data processing speed and algorithm performance for Apriori.

The general process of implementing Apriori for OpenCL includes the following five steps:

Development of an OpenCL kernel for computing subsets of itemsets and analyzing their frequency.

Implementation of mechanisms for data partitioning and distribution among different computing devices.

Configuration and optimization of computing system parameters for maximum performance.

Testing and validation of the implemented algorithm on real datasets.

4.5. Apache Spark platform

Apache Spark is a distributed data processing system that provides capabilities for parallel processing of large volumes of data. One of its features is the use of the Scala language for algorithm implementation.

For data processing, candidate itemsets generation, Apriori algorithm iterations, and result analysis using Spark, functions should be applied, some of which are listed in Table 4. The availability of functions is a distinctive feature of Spark.

Table 4
Basic Apache Spark functions to implement the Apriory method

	Functions	Action
Data processing functions		
1	map(func)	Enables you to perform data transformation, processing, and cleaning operations in a distributed Spark environment.
2	flatMap(func)	as a result of the operation of the flatMap method, each element can generate zero or more elements in the original RDD.
Functions for building itemsets candidates		
1	Cartesi (anotherRDD)	Returns an RDD that contains all possible pairs (combinations) of elements from the current RDD and another RDD.
2	mapPartitions(func)	Applies the given function func to each section of the RDD and returns a new RDD with the results.
Apriori functions (iterations)		
1	map	Data processing and filtering in RDD
2	flatMap	
3	filter and others	
Analysis of results		
1	reduceByKey(func)	Combines the values for each key in the RDD using the given function func.
2	collect	Gathers all elements from the RDD into a node and returns an array.

4.6. Apache Flink Platform

The features of implementing the Apriori algorithm for Apache Flink include the following aspects:

– Stream processing: Apache Flink supports stream data processing, allowing data to be processed in real-time. This can be particularly useful for analyzing large data streams, such as in large networks or IoT systems.

– Distributed processing: Apache Flink provides distributed data processing capabilities, enabling the utilization of server clusters for fast processing of large volumes of data.

– Integration with other tools: Apache Flink easily integrates with other Apache tools such as Apache Hadoop, Apache Kafka, Apache Hive, and others. This facilitates seamless data exchange between different systems and leverages additional functionalities.

– Programming simplicity: Apache Flink offers a convenient programming interface for implementing complex analytical tasks on large datasets. It supports programming languages such as Java, Scala, and Python, making program development easier for a wide range of developers.

– Performance optimization: Apache Flink incorporates built-in mechanisms for performance optimization, including query optimization and execution plan scheduling. This maximizes cluster resource utilization and reduces data processing time.

– Scalability: Apache Flink scales easily both vertically and horizontally, allowing for the processing of both small and large volumes of data. It supports horizontal scaling, enabling the addition of new nodes to the cluster to enhance its performance.

Despite its many advantages, Apache Flink also has its drawbacks. These include configuration complexity, the prevalence of Apache Spark, and significant resource requirements, which can pose challenges for scalability and performance when processing large datasets. Additionally, the current stage of incomplete and inadequate documentation and debugging issues does not recommend Apache Flink for roles other than research-oriented ones.

5. Recommendations and Results

The execution speed of the Apriori algorithm for the same data volume and with a fixed support count can significantly depend on various factors, such as the characteristics of the program itself, data peculiarities, and the hardware on which it runs. Therefore, there is no definitive answer to the question of which algorithm is "absolutely" fast.

However, some general guidelines can be provided:

OpenCL Technologies: These can provide a speed boost in execution through parallel computations on a large number of GPU cores.

MPI and MPI4py: These can work very efficiently on clusters with a large number of nodes. However, MPI requires complex development and management of communications between cluster nodes.

OpenMP: It provides a simple way to parallelize code for multi-processor systems with multicore processors. It can be effective for execution on a single node with multiple processors.

Hadoop and Spark: These are designed to work with large data volumes on server clusters. They provide distributed computing capabilities but usually demonstrate lower speed compared to optimized solutions for parallel computing on GPUs or multicore systems.

Apache Flink: It is a distributed data processing system for streaming data and batch data. It can be an effective option for real-time data processing or streaming data.

In addition, performance can be improved by optimizing the code, using specialized libraries and frameworks, and adjusting system parameters for maximum productivity. It is also important to consider the specific task and project requirements when choosing a parallel programming platform.

Thus, an analysis of the feasibility of using different platforms for association rule mining algorithms has been conducted, and recommendations have been provided. The results of the descriptive comparison can assist researchers in selecting a data processing algorithm for large volumes of data using the Apriori algorithm or in solving other data mining tasks.

In addition to general recommendations, the authors conducted a series of experiments.

To conduct the experiments, several datasets of different sizes were formed. All datasets were stored in CSV format files. This format is lightweight, contains no metadata, and is easily readable, which is crucial when reading large volumes of data.

The experiments were conducted under identical conditions using a multicore computer with variations on 2, 4, and 8 cores.

For comparison, the parallel processing and parallelization algorithm described in the authors' works were applied, as well as implementations of data analysis algorithms from popular open-source data analysis libraries, namely RapidMiner and WEKA, and with the Apache Spark MLLib MapReduce-based parallel computing system.

Weka is a free library and software for machine learning and data analysis developed at the University of Waikato in New Zealand. Weka provides a wide range of machine learning algorithms for classification, clustering, regression, association rules, visualization, and other data analysis tasks. The Apriori algorithm in Weka is available through the user interface and through the programming interface. Weka is primarily designed for use on a single server or workstation and does not have built-in support for distributed or parallel computing using GPUs, MPI, or Hadoop. However, in combination with other tools and technologies, the Weka library can be used as a node in a general computation algorithm.

RapidMiner is a powerful data analysis platform for machine learning, model development, and visualization that allows users to effectively analyze and use data for decision making. RapidMiner can effectively work with data of various volumes, processing both small data volumes (e.g., a few megabytes or gigabytes) and large data volumes (several gigabytes or terabytes). The Apriori algorithm in RapidMiner is available as one of the modules for data analysis. Although RapidMiner is not directly associated with MPI (Message Passing Interface) or Hadoop, it can be integrated with them for processing large data volumes and distributed computing.

Apache Spark MLlib (Machine Learning Library) is a machine learning library that is part of Apache Spark, an open-source framework for parallel and distributed data processing. MLlib provides a wide range of machine learning algorithms and data analysis algorithms that can be used to solve various tasks, such as classification, regression, clustering, recommendations, image processing, and more. These algorithms are optimized for execution on distributed clusters using Apache Spark, allowing

large volumes of data to be processed quickly and efficiently. The Apriori algorithm is not included in the standard Apache Spark but is implemented using the Python programming language.

Apache Spark is designed for efficient processing of large volumes of data in distributed environments. It is integrated with Hadoop and can use it as the primary data store and platform for distributed computing, although its library can also be used separately from Hadoop. At the core of this integration is a common distributed file system mechanism (HDFS - Hadoop Distributed File System) that allows Apache Spark to work with data stored in Hadoop.

The results of the experiments for various data analysis algorithms are presented in Table 5.

Table 5
Execution Time of Association Rule Mining Algorithms

Cores / Sample a	Apache Spark MLLib				DXelopes				Rapid Miner	Weka
	1	2	4	8	1	2	4	8	1	1
1000000	0,156	0,149	0,119	0,096	0,108	0,092	0,066	0,051	0,1144	0,0663
2000000	0,295	0,280	0,192	0,111	0,201	0,135	0,084	0,074	0,2145	0,1248
3000000	0,454	0,376	0,241	0,133	0,278	0,168	0,110	0,092	0,3289	0,1911
4000000	0,565	0,413	0,301	0,174	0,370	0,211	0,127	0,113	0,4225	0,247
5000000	0,618	0,485	0,350	0,237	0,415	0,301	0,190	0,131	0,5161	0,3016
6000000	0,823	0,686	0,397	0,299	0,548	0,319	0,176	0,143	0,5967	0,3484
7000000	0,935	0,761	0,475	0,360	0,632	0,344	0,194	0,155	0,6773	0,3952
8000000	1,119	0,913	0,585	0,417	0,706	0,403	0,219	0,170	0,8112	0,4732
9000000	1,240	0,999	0,692	0,464	0,800	0,446	0,235	0,198	0,8983	0,5239
10000000	1,445	1,089	0,778	0,522	0,849	0,595	0,370	0,239	1,0192	0,5954

Graphical Interpretation of Results (calculation time for the Apriori algorithm on different platforms and data volumes) is provided in Figure 2.

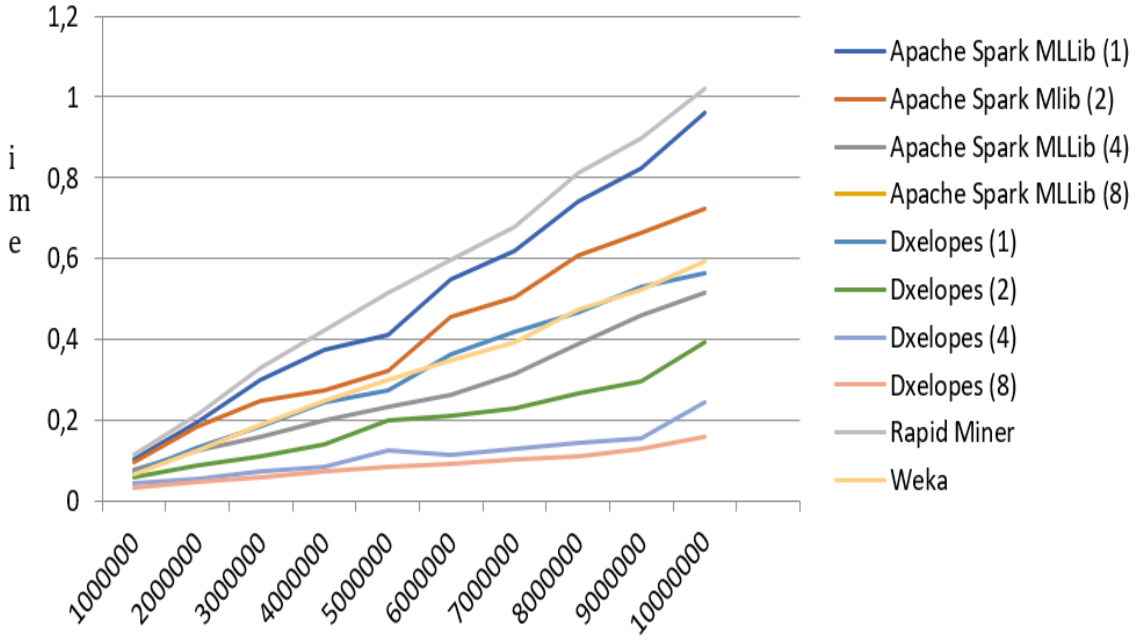

Figure2: Graphical Interpretation of Results

Thus, for algorithms and platforms without GPUs, the experiment revealed that the most efficient algorithm was the parallel implementation using the Weka platform, but it does not provide parallel processing capabilities. Additionally, its execution was preceded by significant preparatory work in data preparation for calculations. Therefore, the Weka library at this stage may be recommended for linear implementations. Slightly lower performance results were provided by the Apache Spark MLLib library, but the further prospect is the full utilization of Apache Spark's capabilities and its integration with Hadoop.

It should be noted that during the experiments, a self-developed algorithm (QuickSort algorithms) and the DXelopes algorithm (Kharkov Polytechnic Institute) were applied. Therefore, the further perspective of the research may be the integration of machine learning methods with the Apriori algorithm. For example, by applying the Apriori method, frequent item sets will be identified in the dataset and used to detect associations between different elements. Machine learning methods will allow modeling complex dependencies and predicting results based on input data and classification, regression models, as well as elements of cluster analysis for various prediction or classification tasks.

6. Conclusions

As a result of the study, findings have been obtained that both generalize and advance the previous research conducted by the authors. The focus of the work has been on selecting platforms for the efficient implementation of deep analysis algorithms (specifically the Apriori algorithm), considering the possibility of parallel and distributed data processing of large volumes.

Pharmaceutical data applicable to medical research and pharmaceutical logistics data were chosen as input data for the experiments. The characteristics of such data, notably their distributed nature and substantial volume, were highlighted. The following actions were undertaken in the study:

The concept of "big data" and the challenges associated with their processing were examined.

The necessity of applying modern algorithms to analyze unstructured data, such as the results of medical research and pharmaceutical data, including pharmaceutical logistics data, was confirmed.

The Apriori method was discussed as a tool for analyzing data of significant volumes, and the study applied the author's QuickSort program.

The instrumental tools for implementing the Apriori algorithm for parallel and distributed processing (MPI, OpenMP, Hadoop, OpenCL, Spark, and Apache Flink platforms) were analyzed.

The peculiarities of applying these platforms in constructing algorithms for association rule mining were described, recommendations were provided, and the results of applying each of the mentioned platforms were forecasted in terms of complexity (required workload) and time to obtain results.

Computational experiments were conducted for cases of using linear and four multi-processor platforms in parallel implementations, confirming the analytical conclusions presented in the final sections of the work.

Paths for further research were outlined, envisioning a deeper experiment in the future application of parallel and distributed versions of association rule mining algorithms in various fields, including the medical-pharmacological sphere, with further integration of the Apriori algorithm with machine learning methods.

It's worth noting that the scientific novelty of the research lies in the conducted and integrated analysis of the characteristics of various information platforms designed for parallel and distributed processing of unstructured data. It has been demonstrated which of these platforms the implementation of deep analysis algorithms, including the Apriori algorithm, will be most efficient in terms of speed and technological feasibility in the processing

The Apriori method has long been recognized as a valuable tool in data mining, particularly for uncovering associative rules in large datasets. However, the ever-growing volume and complexity of data pose significant challenges to traditional methods of analysis. In this context, the scientific novelty of this study lies in the exploration of effective methods and algorithms for implementing the Apriori algorithm specifically tailored to handle large volumes of pharmaceutical data, with a focus on parallel and distributed processing.

The study delves into the methodological approaches necessary for processing unstructured data in pharmaceutical logistics, highlighting the importance of finding efficient Data Mining algorithms, particularly those capable of parallel and distributed processing. This aligns with the broader challenges faced in big data processing across various sectors, emphasizing the need for specialized approaches to handle the unique characteristics of pharmaceutical data.

The research systematically evaluates various platforms for implementing the Apriori algorithm, considering factors such as parallelism, memory-level processing, performance optimization, and scalability. This comprehensive analysis extends beyond theoretical considerations to practical implementations, providing insights into the feasibility and effectiveness of each platform in real-world scenarios.

Furthermore, the study conducts experiments to empirically validate the performance of different implementations of the Apriori algorithm on various platforms and datasets. By comparing results across different platforms and data volumes, the research offers valuable insights into the relative strengths and limitations of each approach, guiding researchers and practitioners in selecting the most suitable methods for their specific requirements.

Overall, the study contributes to the advancement of data mining techniques in pharmaceutical logistics by offering novel insights into the implementation of the Apriori algorithm for processing large volumes of data. By addressing the challenges posed by big data in this domain and proposing practical solutions, the research enhances our understanding of how Data Mining methods can be effectively applied to extract valuable insights and improve decision-making processes in pharmaceutical logistics.

Acknowledgements

Thanks to Professor O. Dmitrieva for guiding the first stages of the work and scientific contribution to research.

References

[1] M. M. Medeiros, N. C. Hoppen and A. G. Maçad, Data science for business: benefits, challenges and opportunities. The Bottom Line: Managing Library Finances, ahead-of-print. (2020). DOI: 10.1108/BL-12-2019-0132

[2] M.Juliardi, I. A Malik, Bibliometric Analysis of Data Science: Trends, Contributions, and Research Developments. West Science Interdisciplinary Studies, 1(07) (2023) 365-375. DOI:10.58812/wsis.v1i07.81.

[3] L. P.Favero, P. Belfiore, Data Science for Business and Decision Making. Elsevier Science. (2019). DOI https://doi.org/10.1016/C2016-0-01101-4

[4] I. Dorohyi. Critical IT Infrastructure Resource Distribution Algorithm. Proceedings of the 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021, 2 (2021): 632–639.

[5] J.Luengo, D.García-Gil, S.Ramírez-Gallego, S.García, F. Herrera, Big Data Preprocessing: Enabling Smart Data. Springer International Publishing. (2021). ISBN 978-3-030-39107-2.

[6] Akobir Shahida, Apriori is a scalable algorithm for searching for associative rules, 2020. https://loginom.ru/blog/apriori

[7] N.Maslova, O. Polovynka Associative Methods as a Tool to Improve the Quality of Knowledge Control. Proceedings of the 4th International Conference on Computational Linguistics and Intelligent Systems {(COLINS} 2020). Volume {I:} Main Conference. Lviv, 2020. pp. 778-787. https://dblp.uni-trier.de/rec/bibtex/conf/colins/MaslovaP20

[8] K.Gulzar, M. A.Memon, S. M.Mohsin,, S.Aslam, A Akber,., M. A.Nadeem, "An Efficient Healthcare Data Mining Approach Using Apriori Algorithm: A Case Study of Eye Disorders in Young Adults," Information 2023, 14(4), (2023). DOI: 10.3390/info14040203

[9] X. Li, B.Tan, J. ZhengXu, J.X. Xiao, Y. Liu. The Intervention of Data Mining in the Allocation Efficiency of Multiple Intelligent Devices in Intelligent Pharmacy. Comput Intell Neurosci, (2022). DOI: 10.1155/2022/5371575

[10] M. L. Faquetti, A. M. D. L. Torre, Th.Burkard, G. Obozinski, Andrea M. Burden, Identification of polypharmacy patterns in new-users of metformin using the Apriori algorithm: A novel framework for investigating concomitant drug utilization through association rule mining, Pharmacoepidemiology and Drug Safety, 2022, Vol. 32, Issue 3, pp. 366-381.

[11] O.L Polovynka, D.V. Nikulin, O.A. Dmitrieva, Study of the speed of operation of a priori algorithms on large volumes of data. Science and production. Series: Information technologies, Mariupol, 2020, 22, pp. 246-253. http://sap.pstu.edu/article/view/211383,

[12] E. A. R. de Campos, I. C. de Paula, C. Sc. ten Caten, K. P. Tsagarakis, J. L.D.Ribeiro, Logistics performance: critical factors in the implementation of end of life management practices in the pharmaceutical care process, Environmental Science and Pollution Research, 2023, Vol. 30, pp. 29206–29228. https://doi.org/10.1007/s11356-022-24035-z

[13] B. Sarwar, Supply chain management in the pharmaceutical industry (2020). https://enhelion.com/blogs/2020/12/30/supply-chain-management-in-pharmaceutical-industry/
[14] O. O. Molodid, V. V. Shemena, General principles of organizing logistics business processes in a pharmaceutical company, Electronic journal "Effective economy", №6, 2019. doi: 10.32702/2307-2105-2019.6.62,
[15] H.Jahani, , R.Jain, D.Ivanov. Data science and big data analytics: a systematic review of methodologies used in the supply chain and logistics research. Ann Oper Res (2023). https://doi.org/10.1007/s10479-023-05390-7
[16] A. Koschmider, M. Aleknonytė-Resch, Fr. Fonger, Christian Imenkamp, Ar. Lepsien, K. Apaydin, M. Harms, D. Janssen, D. Langhammer, T.Ziolkowski, Y.Zisgen, Process Mining for Unstructured Data: Challenges and Research Directions Cornell University, (30 Nov 2023), 2023. https://doi.org/10.48550/arXiv.2401.13677
[17] O.A. Dmitrieva, O.L. Polovynka, Algorithmic support for parallel methods of searching for associative rules, Collection of scientific works of the Donetsk National Technical University. Series: "Computer technology and automation", Pokrovsk, 2018, Is. 1 (31). pp. 62-69, https://science.donntu.edu.ua/ota-arhiv/31/025_dmitrieva.pdf
[18] The Apache Software Foundation. Apache Hadoop, (28 July, 2020), 2020. URL: https://blogs.apache.org/hadoop/entry/announce-apache-hadoop-3-3
[19] O.Polovynka, O. Dmitrieva, Research of the efficiency of the apriori group algorithms on different database sizes, SWorldJournal, Issue 6 / Part 1, p. 71-78, https://www.sworldjournal.com/index.php/swj/article/view/swj06-01-012
[20] Yange, T. S., Gambo, I. P., Ikono, R., & Soriyan, H. A. A Multi-Nodal Implementation of Apriori Algorithm for Big Data Analytics using MapReduce Framework. International Journal of Applied Information Systems (IJAIS), 2020, 12(31). DOI: 10.5120/ijais2020451868