energies

Article

Distributed Singular Value Decomposition Method for Fast Data Processing in Recommendation Systems

Krzysztof Przystupa [1,*], Mykola Beshley [2], Olena Hordiichuk-Bublivska [2], Marian Kyryk [2], Halyna Beshley [2], Julia Pyrih [2] and Jarosław Selech [3]

[1] Department of Automation, Lublin University of Technology, 20-618 Lublin, Poland
[2] Department of Telecommunications, Lviv Polytechnic National University, 79013 Lviv, Ukraine; mykola.i.beshlei@lpnu.ua (M.B.); obublivska@gmail.com (O.H.-B.); marian.i.kyryk@lpnu.ua (M.K.); halyna.v.beshlei@lpnu.ua (H.B.); yuliia.v.klymash@lpnu.ua (J.P.)
[3] Institute of Machines and Motor Vehicles, Poznan University of Technology, 60-965 Poznan, Poland; jaroslaw.selech@put.poznan.pl
* Correspondence: k.przystupa@pollub.pl

Abstract: The problem of analyzing a big amount of user data to determine their preferences and, based on these data, to provide recommendations on new products is important. Depending on the correctness and timeliness of the recommendations, significant profits or losses can be obtained. The task of analyzing data on users of services of companies is carried out in special recommendation systems. However, with a large number of users, the data for processing become very big, which causes complexity in the work of recommendation systems. For efficient data analysis in commercial systems, the Singular Value Decomposition (SVD) method can perform intelligent analysis of information. With a large amount of processed information we proposed to use distributed systems. This approach allows reducing time of data processing and recommendations to users. For the experimental study, we implemented the distributed SVD method using Message Passing Interface, Hadoop and Spark technologies and obtained the results of reducing the time of data processing when using distributed systems compared to non-distributed ones.

Keywords: big data (BD); message passing interface (MPI); distributed systems (DS); singular value decomposition (SVD); hadoop; spark

check for updates

Citation: Przystupa, K.; Beshley, M.; Hordiichuk-Bublivska, O.; Kyryk, M.; Beshley, H.; Pyrih, J.; Selech, J. Distributed Singular Value Decomposition Method for Fast Data Processing in Recommendation Systems. *Energies* **2021**, *14*, 2284. https://doi.org/10.3390/en14082284

Academic Editor: Silvio Simani

Received: 10 March 2021
Accepted: 14 April 2021
Published: 19 April 2021

1. Introduction

For the needs of large companies engaged in trade, services and other related activities, an important problem is the constant collection of information about the characteristics of customers and their preferences for various goods or services.

Small companies can collect information without using complex computing resources. This can be done with simple computing devices that do not require much complexity in servicing [1]. However, for large companies there is a need to analyze the data of many users. Regional, national or international companies work with the data of millions of people [2].

In this case, it is advisable to apply other approaches to the organization of the computer system. Even with increasing the capacity of the computing device that processes user data, the risk of failure due to heavy workload increases.

For effective data analysis, it is necessary to use sufficiently complex algorithms that are able to process information about the user and reject the insignificant. For example, if the user of the online store ordered all the things from the autumn collection, there is no need to store data about each favorite thing separately, in order to then conclude that the user of the store likes this style of things [3]. This is impractical in terms of memory usage and computing power of user data processing systems [4].

Large companies, which provide certain services to customers, use special recommendation systems to form recommendations to users in the form of advertising for new products or services based on previous preferences [5–7].

In such tables on one axis there aredata about the user, on the other axis there aredata about goods, and at the intersection there are user ratings for a certain product. This can be a certain average value, which is determined by certain actions of the user on a product: photo preferences in social networks, an Internet search, a purchase, etc. On the basis of such tables, you can see the visual statistics of different products and services and find certain regularities with regard to user behavior.

In the simplest case, when the number of users is small, it is possible to manually collect data on their actions and understand the relevance of certain products. It is also possible to immerse users, such as students, musicians, seniors and the like, in certain features. Based on this, we can assume that if a certain product is popular among many members of the group, it can be offered to a user who has not yet givena single evaluation. Detailed process of the recommendation system is presented in Figure 1 [8,9].

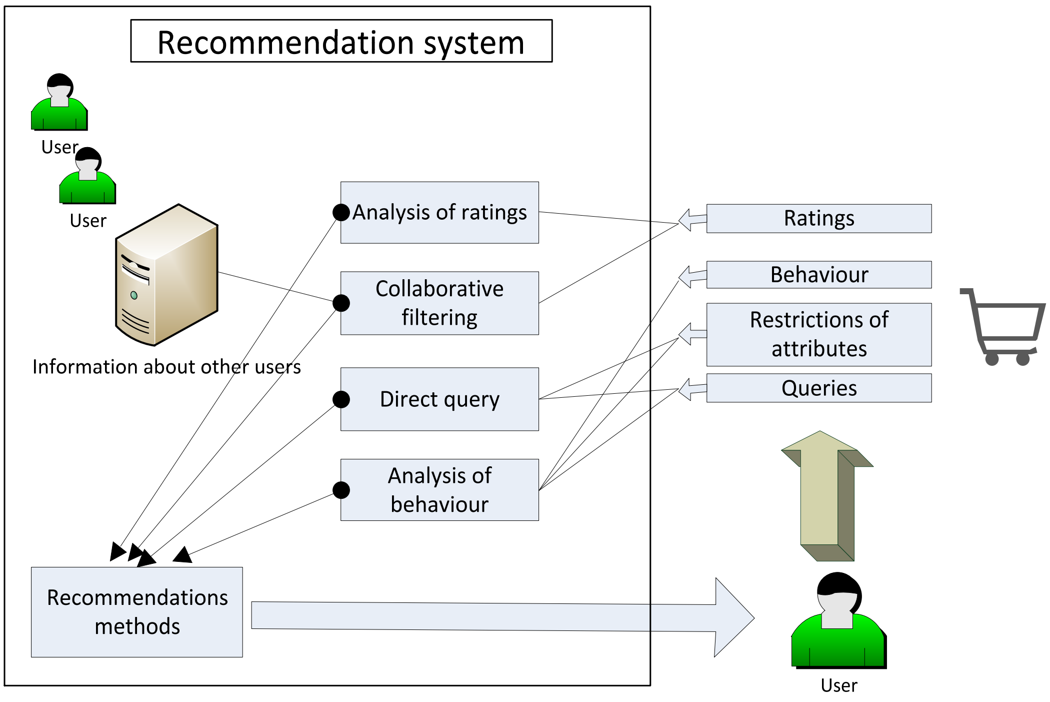

Figure 1. Recommendation systems.

Thus, recommendation systems are able to learn from the information obtained and fill in the blanks in the preferences table. However, for systems with millions of users, special information analysis methods must be used. One of them is the Singular Value Decomposition (SVD) method [10], which, unlike other methods, is easier to use and provides less data processing time.

However, more and more commercial companies are now switching to online mode. This puts even more strain on referral systems. If the operation of the data processing algorithm is supported only by the node server, then sooner or later it will not be able to process real-time data, and there may be delays in providing recommendations. It is also possible for the computing device to fail and, if it alone processes all the data, the system goes completely out of order.

It is proposed to use distributed data processing systems to improve efficiency. In them, all the data that require processing aredivided into smaller parts and processed simultaneously on different computing devices. In this case, data processing time is reduced and the risk of system failures is reduced.

For the operation of distributed systems, it is advisable to use different technologies of distributed data processing. The simplest is MPI (Message Passing Interface) technology. In it, the data aresimply distributed by the central node at the part level and sent to the working nodes for processing [11]. This is communicated between nodes in the system using a special messaging protocol. As the amount of data in the system increases, service messages acquire significant volumes, which has a negative effect on computing performance. Hadoop technology is more efficient, in which there are no such service messages, the data aresimply sent to the working devices, written to its non-volatile memory, processed and sent to the central node. After processing, the data aredeleted from the device memory [12–14]. It is the process of writing and deleting data from non-volatile memory that negatively affects the computation time, especially with large amounts of data. Spark technology solves this problem because the data arenot written to non-volatile memory, but to Random Access Memory (RAM), which speeds up the computation process [15]. It is also worth noting a concept such as distributed agents [16,17]. These are intelligent devices in a distributed system that are able to exchange messages with each other and determine the optimal load on each node.

With the increasing volume of data, the speed of providing users with quality recommendations among large volumes of information has become a serious problem. With the spread of mobile services usage the speed of response to the request also has become an important indicator of quality user services. Nowadays, in recommendation systems that use the Internet as a communication bus, there is a technical problem with the guaranteed quality of services by the criteria of response time of the service. Since the total response time for the provided service consists of the processing time of requests on application servers and network latency introduced by switches and routers, mobile gateways, the task of delivering the requested service response time requires proper management of both communication and computing resources [18–21].

The remainder of the paper is organized as follows. In Section 2, we present existing work related to different aspects of our approach. Then we propose our Singular Value Decomposition method used to reduce the dimensionality of the data in recommendation systems in Section 3. We report the experimental validation in Section 4. Section 5 discusses the results and how they can be interpreted in future studies and finally conclude the paper in Section 6.

2. Related Works

In this part, we illustrate the research status of the development of recommendation systems for data processing. Nowadays humanity is completely absorbed by the information space: goods, films, books, articles, news, etc., and it is difficult to determine which product is more suitable for a particular person. Accordingly, there is a need for technologies that can quickly process large amounts of data and allocate only theinformation that is useful for a particular customer. Recommendations systems are one of the most powerful technologies for solving such problems. Data processing of large volumes is an urgent problem. Because of this, many scientific papers contain research on this topic. Handri, K.E. et al. [22] proposedhybrid recommendation systems based on Spark Cloud. In this article, the author presented a new parallel algorithm in a distributed recommendation system based on the Apache Spark platform. The main idea of this approach was to implement a multi-criteria approach to support decision-making and query dominance using matrix factorization.

Zheng-Yi Chai et al. [23] proposed the algorithm based on SVD and Multi-Objective Immune Algorithm (MOIA), or SVD-MOIA. In this research MOIA was used to optimize the conflicting goals of accuracy and diversity. However, the authors only modified the algorithm in terms of determining the optimal number of parameters to provide recommendations. The question of applying the obtained method to large amounts of data remains unresolved. Therefore, in our research we modified the SVD method to be able to ensure the effective operation of the recommendation algorithms, even with increasing data.Y. Ji et al. [24] proposed to regularize the method of singular decomposition based on the selection of parameters. In paper [25] the adaptation of recommendation systems for the needs of data processing on publications wasconsidered by authors T.Achakulvisut et al.

Li G. et al. [26] proposed a combination of the Singular Decomposition Method and statistical methods for the operation of the travel guide system, but theydidnot present an adaptation of the SVD method for use in distributed systems.Guo X. et al. [27] proposed a method of combining element attribute information with a historical rating matrix to predict potential user preferences. This method combines attribute and temporal information in a decomposition matrix model. The described method is an effective way to solve the problem of cold start of new elements. Methods of a data processing system that analyzes data about movies based on client-oriented algorithms werediscussedV. Chen et al. in paper [28].

Ferreira, D et al. [29] proposed to use an autocoder based on the Singular Value Decomposition algorithm. W. Hong-Xia et al. [30] comparedthe methods of classical collaborative filtering and the method of singular decomposition.

The methods of data processing in polymer systems by means of an algorithm of singular decomposition have been discussed. Methods of data processing in polymer systems using a Singular Value Decomposition algorithm have beengiven. I. Koprinarov and his colleagues proposedto improve the performance of recommendation systems by identifying hidden relationships between elements [31].

A. Al-Sabaawi et al. [32] offered the new model by integrating three sources of information: user-point matrix, explicit and implicit relationships. The key strategy of this research is to use the Multistage Resource Allocation (MSRA) method to identify hidden relationships in the social information. First, explicit social information is used to determine similarities between each pair of users. Second, for each pair of users who are not friends, the MSRA method is used to determine the probability of their relationships. If the probability exceeds a threshold value, a new relationship is established. All sources are then included in the Singular Value Decomposition (SVD) method to calculate the missing forecast values.

The authors of [33] offered a new model of recommendations based on the time correlation coefficient and an improved cuckoo search K-tool (CSK-tool) called TCCF (the novel collaborative filtering algorithm based on CSK-means). The clustering method can cluster similar users for further fast and accurate recommendations. H. Luo et al. [34] proposed a system of drug repositioning guidelines (MDRs) for predicting new drug readings by integrating relevant data sources and evidence-based information on drugs and diseases. The authors used a fast Singular Value Threshold (SVT) algorithm to fill the adjunct matrix between drugs and diseases with predicted scores for unknown drug pairs. Full experimental results showed that DRRS (drug repositioning recommendation system) increases the accuracy of predictions compared to other modern approaches. In addition, case studies on several selected drugs once again demonstrated the practical utility of the presented method. In work [35,36], the authors provided a combination of random algorithms and matrix decomposition in recommended distributed systems.

In work [37], modifications of the SVD algorithm were considered J. Wu et al. in order to provide not only exact, but also various recommendations to users.

Despite the considerable amount of related work, itdoes not fully describe the work of the Singular Value Decomposition method in distributed systems and application of distributed data processing technologies. These issues are discussed in thearticle below.

In the articles presented here, methods of improving the operation of the SVD algorithm were described. In particular, the principles of improving the accuracy of recommendations and anticipation of user preferences wereconsidered.

However, no article studied the modification of the SVD algorithm to work in distributed systems. In this article westudied of the dependence of the algorithm on the architecture of a distributed system.

In particular, it was proposed to distribute certain operations of the same type between different computing resources. To do this, a number of operations have been identified that can be performed in a distributed manner. To use the modified algorithm with maximum efficiency, the algorithm is modeled by changing the following parameters: the number of operations performed in a distributed manner, the number of computing resources and the amount of data received at the input of the algorithm.

The proposed approach allowed to determine the optimal number of operations to be performed in a distributedmanner for different architectures of distributed systems and data volume. It is determined that the greatest efficiency of the proposed modified algorithm shows a large number of processors in the system.

3. Research and Modeling of the Method of Collaborative Data Filtering

In order to process data more effectively and make recommendations to users, it is not enough to use a model that simply takes into account that a certain user liked a certain subject. It is necessary to pass to the more exact description in which the general tendencies for certain categories of users to be interested in any subjects are considered.

It also eliminates the problem of too much data. Instead of using a large number of product evaluation vectors to evaluate user preferences, a much more compact preference vector is used.

This section will discuss the Singular Value Decomposition method used to reduce the dimensionality of the data. It is used in problems of finding matrices of hidden characteristics of system users and objects of interest [10].

In a singular decomposition, the rating matrix R ($m \times n$) can be decomposed into the product of three matrices: U ($m \times k$), Σ ($k \times k$) and V ($k \times n$) (k varies depending on the degree of detail of the schedule). Matrices U and V are responsible for the hidden characteristics of users and objects (age, region of residence, price, etc.).

3.1. The Method of Singular Value Decomposition

The method of Singular Value Decomposition (SVD) is that the matrix $A \in M(n, m)$, whose dimension is $d = \mathit{rank}(M) \leq \mathit{min}(n, m)$, can be represented as the product of lower order matrices:

$$A = U \cdot D \cdot V^T \tag{1}$$

where U and V are matrices $U \in M(n, d)$, $V \in M(n, d)$.

These matrices contain orthogonal columns, which are eigenvectors (for nonzero values) for $A \cdot A^T$ and $A^T \cdot A$ matrix.

It should also be borne in mind that

$$U^T \cdot U = V^T \cdot V = I \tag{2}$$

where A is the identity matrix in which the elements located on the main diagonal are equal to one, and all the others are zero.

D is a matrix of size (d, d):

$$D = \begin{matrix} a_1 & 0 & 0 \\ 0 & \ldots & 0 \\ 0 & 0 & a_d \end{matrix} \tag{3}$$

is a diagonal matrix that contains positive elements along the main diagonal. These items are sorted in descending order:

$$a_1 \geq a_2 \geq \ldots \geq a_d \geq 0 \tag{4}$$

SVD is an extension of the original data in the coordinate system, where the covariance matrix is diagonal.

The operation of the Singular Value Decomposition method is shown in Figure 2.

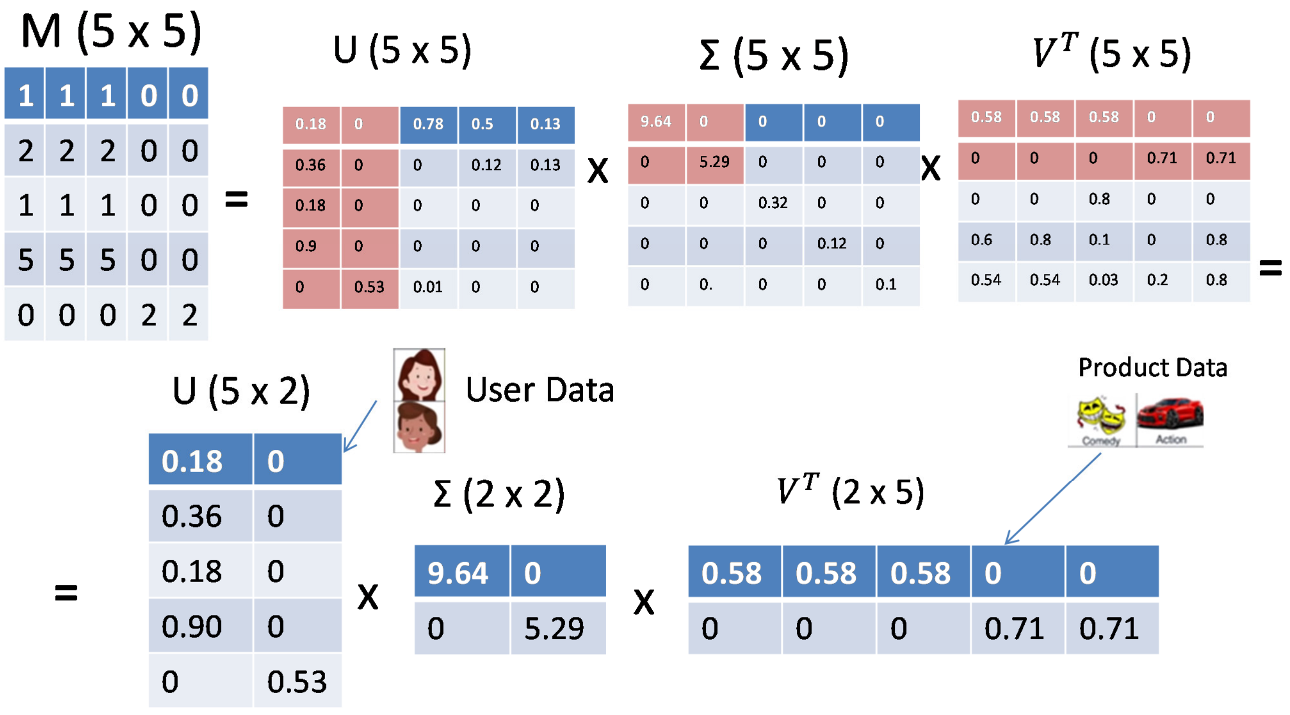

Figure 2. Singular Value Decomposition (SVD).

The initial matrix M is divided into the product of orthogonal matrices U and V and the diagonal matrix Σ.

You can select a certain k maximum number of elements of the diagonal of the matrix Σ and discard other rows and columns. In the matrices U and V, respectively, we also leave only k rows. After multiplying the new submatrices, we obtain a data matrix that repeats with high accuracy the characteristics of the initial M, but has a smaller dimension, because it does not take into account redundant and unnecessary information.

The work of the SVD method consists of finding eigenvalues and eigenvectors (denoted $A \cdot A^T$ and $A^T \cdot A$) [22]. Eigenvectors $A^T \cdot A$ correspond to the columns of the matrix V. Eigenvectors $A \cdot A^T$ correspond to the columns of the matrix U.

The elements of the matrix S correspond to the square roots of the eigenvalues $A \cdot A^T$ or vectors or $A^T \cdot A$.

The elements of the matrix S are arranged in descending order. They always belong to real numbers.

(1) Application of the Singular Value Decomposition in Recommendation Systems

To apply a singular decomposition in recommendation systems, we first search for the user that most closely resembles the current one, using a matrix U that has a reduced dimension. For such a search special algorithms of finding of similarity between elements are applied.

The necessary assessment is then provided. To do this, we use a matrix of elements. It searches for items that are most similar to those previously rated by this user. The estimate of this element can be used for prediction.

Rows and columns of matrices are called vectors. The row of the matrix U corresponds to the users of the system, the column of the matrix Vcontains the elements that are evaluated.If user vectors are similar, we can assume that users have similar preferences. In the method of Singular Value Decomposition, the most attention is paid to finding similar users or elements. To calculate the similarity, such methods are used as: finding the Euclidean distance between the elements, calculating the Pearson correlation coefficient, etc.

(2) Standard Method of Singular Value Decomposition

In order to implement the standard method of singular decomposition of matrices, the following steps should be performed:

1 For the initial matrix A, its QR–schedule is searched;
2 We present the matrix of recommendations RT in the form of the product of three matrices: $L^T \cdot D_2 \cdot R^T$. It is worth noting that the matrix D_2 is a two-diagonal matrix;
3 We representthe matrix D_2 as a product of three matrices: $L^T \cdot D_1 \cdot R^T$ (matrix D_1 has a diagonal appearance).

(3) Finding a QR–schedule

To find the decomposition of the input matrix, its representation is performed as the product of the matrix Q, which is orthogonal, and the matrix R (which is upper triangular).

These operations can be performed using the Givens rotation method.

Consecutive transformation of all elements located under the main diagonal of a matrix to zero is carried out. To do this, we multiply the matrix by a certain rotation matrix (denoted as G):

$$A = G_k \cdot G_{k-1} \cdot \ldots \cdot G_1 \cdot R, \tag{5}$$

$$Q = G_k \cdot G_{k-1} \cdot \ldots \cdot G_1, \tag{6}$$

$$A = Q \cdot R. \tag{7}$$

(4) Bringing the RT Matrix to a Two-Diagonal Form

The matrix R is upper triangular, respectively, R^T is lower triangular. There is a gradual zeroing of the elements under the main diagonal (first, the lower columns of the matrix R^T). This can be done by multiplying R^T on the left matrix of turns of Givens [37].

It should be noted that in each iteration of the method, the element of the matrix $r(i,j)$ may be nonzero. In cases of fairness of the condition $j \neq (i+1)$ we must multiply the matrix R^T on the right matrix of rotation of Givens on the right that the given element was equal to zero.

Therefore, only the elements of the matrix thatare located on the main diagonal and directly above it are not zero:

$$\begin{matrix} x & 0 & 0 \\ x & x & 0 \\ x & x & 0 \end{matrix} \Longrightarrow \begin{matrix} x & Z & 0 \\ 0 & x & Z \\ 0 & 0 & x \end{matrix}. \tag{8}$$

The result of such operations is a matrix D_2 (which is characterized by dual diagonality):

$$D_2 = L_1 \cdot R^T \cdot R_1. \tag{9}$$

(5) Bringing the Matrix D_2 to a Diagonal Form

To make it diagonal, we will gradually assign a zero value to elements that are not on the main matrix diagonal (as long as at least one element is larger than the set value).

When operations are performed with elements below the main diagonal, the matrix is multiplied by the left rotation matrix. When processing the elements located above the main diagonal, multiplication by the right rotation matrix is performed.

After iterative processing of the matrix D_2 it is gradually reduced to a diagonal form. Then the resulting diagonal matrix D_1:

$$D_1 = L_2 \cdot D_2 \cdot R_2. \tag{10}$$

As a result, we get:

$$R^T = L_1^T \cdot D_2 \cdot R_1^T, \tag{11}$$

$$D_2 = L_2^T \cdot D_1 \cdot R_2^T, \tag{12}$$

$$R_2^T = L_1^T \cdot L_2^T \cdot D_1 \cdot R_2^T \cdot R_1^T, \tag{13}$$

$$R = R_1 \cdot R_2 \cdot D_1^T \cdot L_1, \tag{14}$$

$$R = R_1 \cdot R_2 \cdot D_1 \cdot L_1 \cdot L_2, \tag{15}$$

$$A = Q \cdot R_1 \cdot R_2 \cdot D_1 \cdot L_2 \cdot L_1, \tag{16}$$

$$U = Q \cdot R_1 \cdot R_2, \tag{17}$$

$$\sum = D_1, \tag{18}$$

$$V = L_2 \cdot L_1, \tag{19}$$

$$A = U \cdot \sum V^T. \tag{20}$$

3.2. QR Matrix Schedule

QR–schedule is a conversion of an input matrix with dimension $(n \times m)$ into the product of matrices: Q, dimension $(n \times n)$, which is orthogonal, and matrix $R(n \times m)$, which is upper triangular.

There are many methods of obtaining a QR–schedule. Among the most common is the method of rotation (it is also called the Givens method).

The method of rotation gradually brings the matrix, which was at the entrance, to the upper triangular shape. The method works as follows: at each stage for the matrix A_h a certain rotation matrix is calculated Q_h, to make the item a_{ij} from the matrix $A_{h+1} (A_{h+1} = Q_h \cdot A_h)$ equal to zero.

The Givens rotation is called the transformation (linear) of the vector plane. This transformation corresponds to the rotation of the two coordinate axes in the plane by a certain angle.

To reset a certain element of the matrix, it is necessary to calculate the rotation matrix corresponding to the initial matrix, then these two matrices are multiplied.

The rotation matrix is as follows:

$$G(i, k, \varphi) = \begin{array}{ccccccc} 1 & & & \cdots & & & \cdots \\ & \ddots & & \cdots & & 0 & \\ & & c & \cdots & s & & \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ & & -s & \cdots & c & & \\ & 0 & & \cdots & & \ddots & \\ \cdots & & & \cdots & & & 1 \end{array}, \tag{21}$$

where $s = \sin(\varphi)$, $c = \cos(\varphi)$, φ is the angle of rotation of the matrix.

For example, find the QR–schedule of a matrix of this kind:

$$A = \begin{array}{ccc} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{array}. \tag{22}$$

We use the rotation method for this.

We will process the elements of the matrix gradually. First, we select the left column, and the elements in it are calculated from top to bottom. Then we work with the second column on the left, and so on.

At the first stage, we calculate the rotation matrix of the element c_1. Rotation matrix G_{c1}:

$$G_{c_1} = \begin{array}{ccc} 1 & 0 & 0 \\ 0 & c & s \\ 0 & -s & c \end{array}. \tag{23}$$

It is not necessary to explicitly calculate the angle φ, but simply determine it by the following formulas:

$$\gamma = \sqrt{b_1^2 + c_2^2}, \quad (24)$$

$$c = \frac{b_1}{\gamma}, \quad (25)$$

$$s = \frac{c_1}{\gamma}. \quad (26)$$

After the multiplication of matrices G_{c_1} and only the lines corresponding to the indexes of the rows where they are located are changed b_1 and c_1.

$$A_{C_1} = G_{c_1} \cdot A = \begin{matrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ 0 & c_2 & c_3 \end{matrix}. \quad (27)$$

Let us perform the same transformations for the elements b_1 and c_2 by analogy:

$$A_{C_2} = G_{c_2} \cdot \left(G_{C_2}\left(G_{C_1} \cdot A\right)\right), \quad (28)$$

$$R = A_{c_2} = \begin{matrix} a_1 & a_2 & a_3 \\ 0 & b_2 & b_3 \\ 0 & 0 & c_3 \end{matrix}, \quad (29)$$

$$Q = G_{c2} \cdot \left(G_{b_1} \cdot G_{c_1}\right). \quad (30)$$

After the calculations, the matrices Q and R are obtained. The matrix Q is orthogonal, and R is triangular. Then:

$$A = Q \cdot R. \quad (31)$$

3.3. *Application of Eigenmatrix Values for Singular Decomposition*

To calculate the Singular Value Decomposition, it is necessary to find the eigenvalues and eigenvectors of the matrices that can be decomposed.

By definition, the eigenvector of the matrix A is called a vector k such that:

$$A \cdot k = l \cdot k \quad (32)$$

where l is the eigenvalue for A.

According to the theory of the Singular Value Decomposition method, $A = U \cdot S \cdot V^T$. Note that U and V have orthogonality properties:

$$U^T \cdot U = V^T \cdot V = I \quad (33)$$

where I is the identity matrix in which the elements located on the main diagonal are equal to one, and all the others are zero.

This is an easy conclusion to draw:

$$A^T = U^T \cdot S \cdot V. \quad (34)$$

From here:

$$A \cdot A^T = U \cdot S \cdot V^T \cdot V \cdot S \cdot U^T = U \cdot S \cdot S \cdot U^T = U \cdot S^2 \cdot U^T, \quad (35)$$

$$A \cdot A^T = V \cdot S \cdot U^T \cdot U \cdot S \cdot V^T = V \cdot S \cdot S \cdot V^T = V \cdot S^2 \cdot V^T, \quad (36)$$

$$A \cdot A^T \cdot U = U \cdot S^2 \cdot U^T \cdot U = U \cdot S^2, \quad (37)$$

$$A \cdot A^T \cdot V = V \cdot S^2 \cdot V^T \cdot V = V \cdot S^2. \quad (38)$$

After analyzing the last two formulas, it becomes obvious that the eigenvectors for the matrix $A{\cdot}A^T$ are columns from the matrix U, eigenvectors for the matrix. Note that eigenvalues for matrices $A{\cdot}A^T$ and $A^T{\cdot}A$ there are numbers $S = diag(l_1, l_2, \ldots, l_n)$.

To find the matrices V and U, we calculate the matrix of the form $A{\cdot}A^T$ or $A^T{\cdot}A$. Then we calculate the second matrix, using:

$$\begin{cases} V = \left(S^{-1}{\cdot}\left(U^T\right){\cdot}A^T\right), \\ U = A{\cdot}V{\cdot}S^{-1}. \end{cases} \tag{39}$$

The matrix S is diagonal:

$$S^{-1} = \left(l_1^{-1}, l_2^{-1}, \ldots, l_n^{-1}\right). \tag{40}$$

For example, we describe the operation of the method for finding the eigenvalues and vectors of the initial matrix.

The basic principle of the QR method is to gradually bring to the maximum similarity of the initial matrix (A) with a similar matrix An. This method is used for this purpose. Since the initial and approximate matrices are very similar, they have the same eigenvalues.

Provided that the initial matrix has real and different eigenvalues, the QR method leads to the resulting matrix, on the diagonal there are eigenvalues, and the matrix itself is upper triangular.

However, the eigenvalues of the initial matrix can also be complex numbers. Then the result of the method will be a block upper triangular matrix of the result. On the diagonal of such a matrix are placed blocks (first and second orders). The first-order blocks have their own real numbers. In the blocks of the second are complex numbers.

It should be noted that complex eigenvalues do not exist in symmetric matrices, so it is convenient to use them in such methods.

The QRmethod is performed in the following stages:

1. The input of the method receives the matrix A;
2. The matrix R_n (upper triangular) and Q_n (diagonal) arecalculated. The product of these matrices corresponds to the initial matrix $A_n = Q_n{\cdot}R_n$;
3. We calculated $A_{n+1} = Q_n{\cdot}R_n$;
4. In the event that Ai is not upper triangular, the method returns to the second stage. If the matrix is triangularwe assume that the diagonals of the matrix are its own values. Also, the matrix formed by multiplying the matrices Q of each iteration of the method contains the eigenvectors of the initial matrix.

3.4. Application of the Modified Method for Carrying Out Singular Value Decomposition

With increasing data dimension, there is a need to process data not on a single computing device, but in a distributed computing system. In such systems, it is possible to modify the Singular Value Decomposition method, in which the same type of operations, such as adding data or multiplying matrices, are divided into smaller parts and processed simultaneously by different devices. Thanks to this approach, it is possible to speed up the running time of the method and, accordingly, to give recommendations to users faster. More detailed implementation of the data processing model using distributed systems is discussed in the next section.

4. Experimental Study of Data Arrays Processing in a Distributed System

To conduct an experimental study of the effectiveness of the proposed method of distribution, Singular Value Decomposition, it was proposed to create a software model in Java.

Three technologies were used to implement distributed computing:

- MPI (Message Passing Interface)—between the nodes that perform data calculations and the main node are exchangeda large number of service messages. As the amount

of data that needs to be processed increases, the number of service messages can take up a significant portion of computing resources, which leads to a deterioration in system performance;
- Hadoop does not use a significant number of service messages. All data received for calculation are written to the permanent memory of the device and, after processing, are deleted. The time of recording and deleting data in this technology slows down;
- Spark technology is similar to Hadoop, but uses RAM (random access memory) instead of persistent memory. It is especially advisable to use this technology when processing large amounts of data.

4.1. Calculation of Singular Value Decomposition in Systems Using MPI

The standard method for searching for a singular matrix decomposition described in the previous section was implemented. During the experiment, the dependences of the program execution time on the method of computing and data dimension were investigated.

To create a testing environment we used software tools and built-in methods to simulate the operation of distributed systems based on MPI, Hadoop and Spark technologies.

The number of computational processes in the distributed system varied from 1 to 16 to determine the benefits of using the proposed modified algorithm. The load distribution in the simulated system was carried out by a software control process, the data were distributed for processing equally amongall available processes. This took into account the possible minor delays in the processing of individual computational processes of their tasks. Therefore, when individual processors were busy, the data weresent to another free one.

This approach significantly increasedthe efficiency of the algorithm, especially when we needed to process Big Data. The reliability of information processing wasalso increased because the load wasdistributed amongseveral processors, and problems or unavailability of one of them didnot stop the whole algorithm.

Distributed algorithm of recommendations. The SVD algorithm is widely used in information systems for data processing. Given that in our study we proposed the use of distributed data processing systems, it was decided to modify the existing SVD algorithm and divide it into two parts: operations that couldbe performed in parallel by different computing devices; and operations that weretoo difficult to perform in parallel, so it was decided to use traditional sequential data processing.

Thus, some operations that wereof the same type, such as multiplication of matrices andexponentiation, were distributed amongseveral computational processes, which speededup the algorithm. An experimental determination of the number of operations to be parallelized was performed in order to achieve maximum efficiency of the modified algorithm.

Figure 3 shows the algorithm for modifying the SVD method in distributed systems.

According to Figure 3, the algorithm performeda sequential execution of the classical SVD algorithm. However, at each step where it wasnecessary to perform a certain action on the data, the possibility of its slave division for processing by different processors waschecked. When modeling the operation of this algorithm, a 4-processor architecture of a distributed system was used. The simulation results are shown in Figure 4.

According to the results obtained in Figure 4, we concluded that with increasing the number of operations performed distributed, the duration of calculations first decreasedand then increased. This wasbecause distributed computations of small operations take more time than their simple sequential execution. Therefore, for further research, we used a distributed execution of no more than 20 operations.

The proposed algorithm allowedto determine the required degree of modification of the SVD algorithm for different architectures of distributed systems with the highest efficiency.

The implementation of the Singular Value Decomposition method in a distributed system is shown in Figure 5.

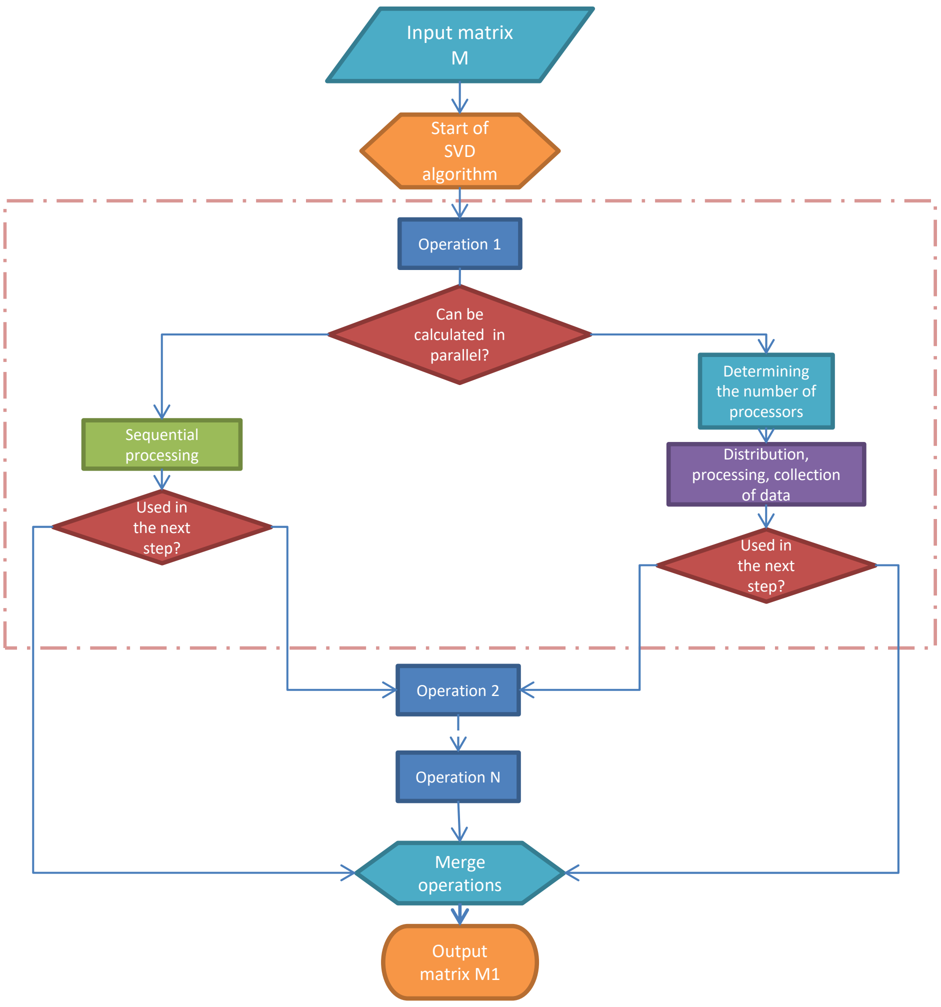

Figure 3. Algorithm for determining the maximum number of distributed operations in the SVD algorithm.

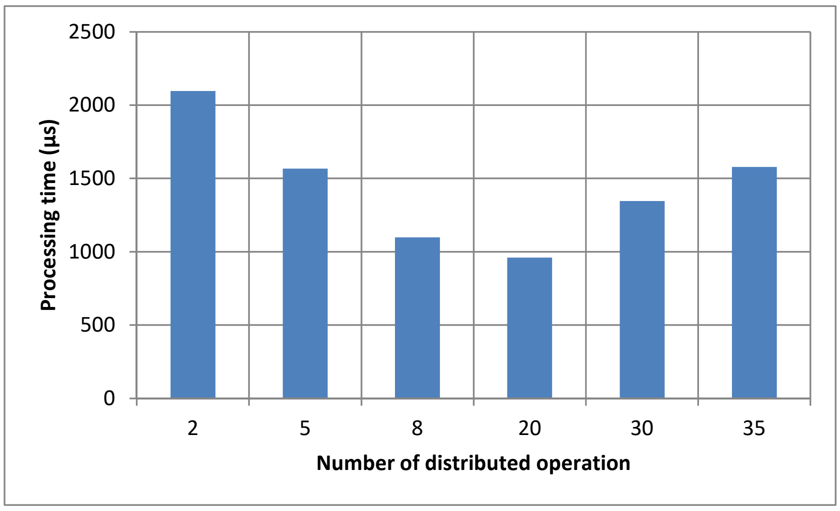

Figure 4. Dependence of SVD processing time on the number of distributed operations.

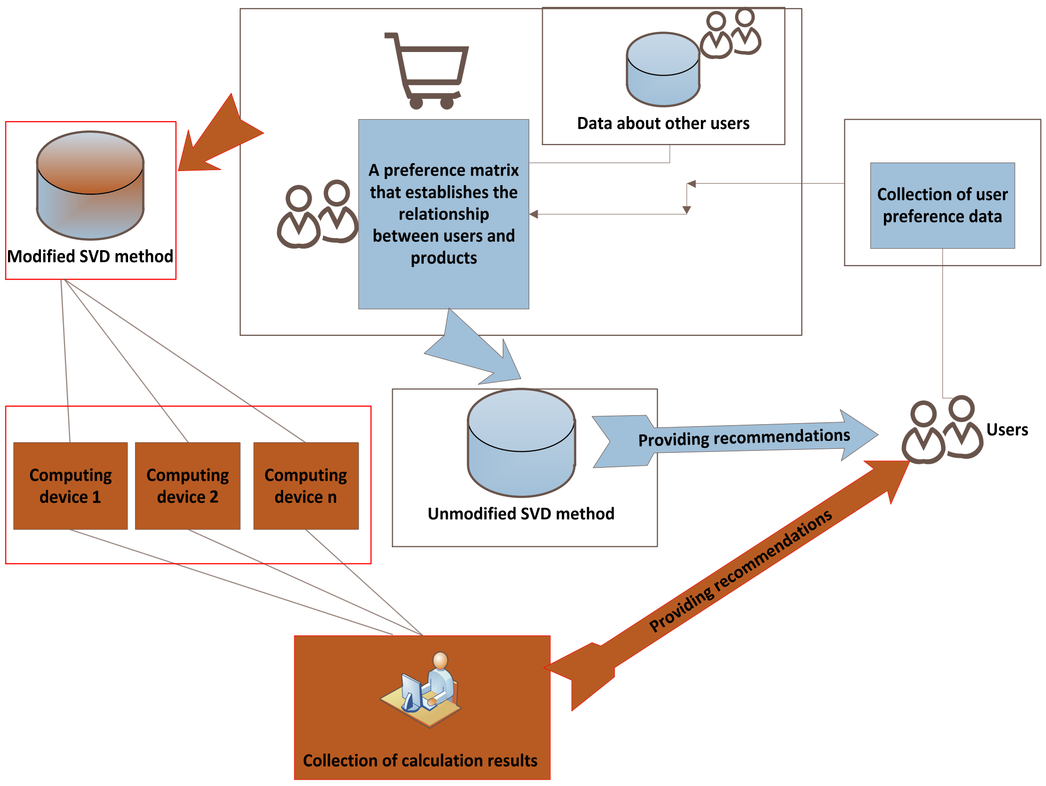
Data about other users
A preference matrix that establishes the relationship between users and products
Collection of user preference data
Modified SVD method
Computing device 1
Computing device 2
Computing device n
Unmodified SVD method
Providing recommendations
Users
Providing recommendations
Collection of calculation results

Figure 5. Implementation of the SVD method in a distributed system.

Figure 5 shows the operation of an unmodified Singular Value Decomposition method, in which all data is processed by one processor, and a modified one, in which the task is distributed to several processors. Distributed processing can increase the efficiency of the computing system. Elements of the modified method are highlighted in red.

Figure 6 and Table 1 show the result of the experiment. The dependence of the execution time of a Singular Value Decomposition of a matrix with a dimension of 120 rows and 120 columns depending on the number of computational processes is given.

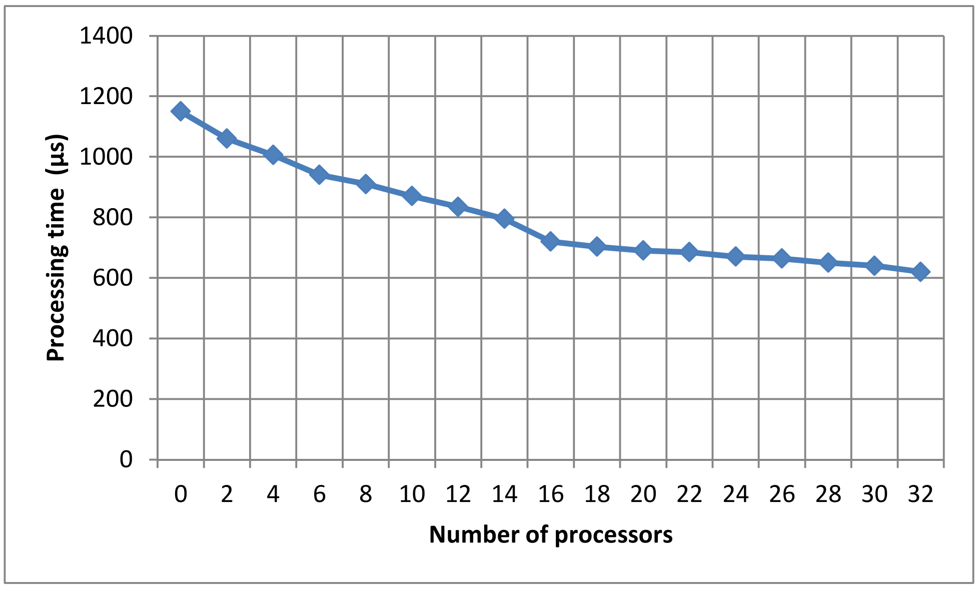

Figure 6. Dependence of SVD processing time on the number of computing processors.

Table 1. Dependence of the execution time of distributed calculations of singular data decomposition depending on the number of computational processes.

Number of Processors	Processing Time (µs)
1	1132
2	1060
4	1006
8	910
16	720
32	620

From Table 1 and Figure 6 we concluded that increasing the number of processes reducedthe processing time calculations because the data weredivided into smaller parts that wereprocessed faster by the processor.

Figure 7 and Table 2 present the dependence of the processing time of calculations on the size of the array of calculated data during processing in a dual-processor system.

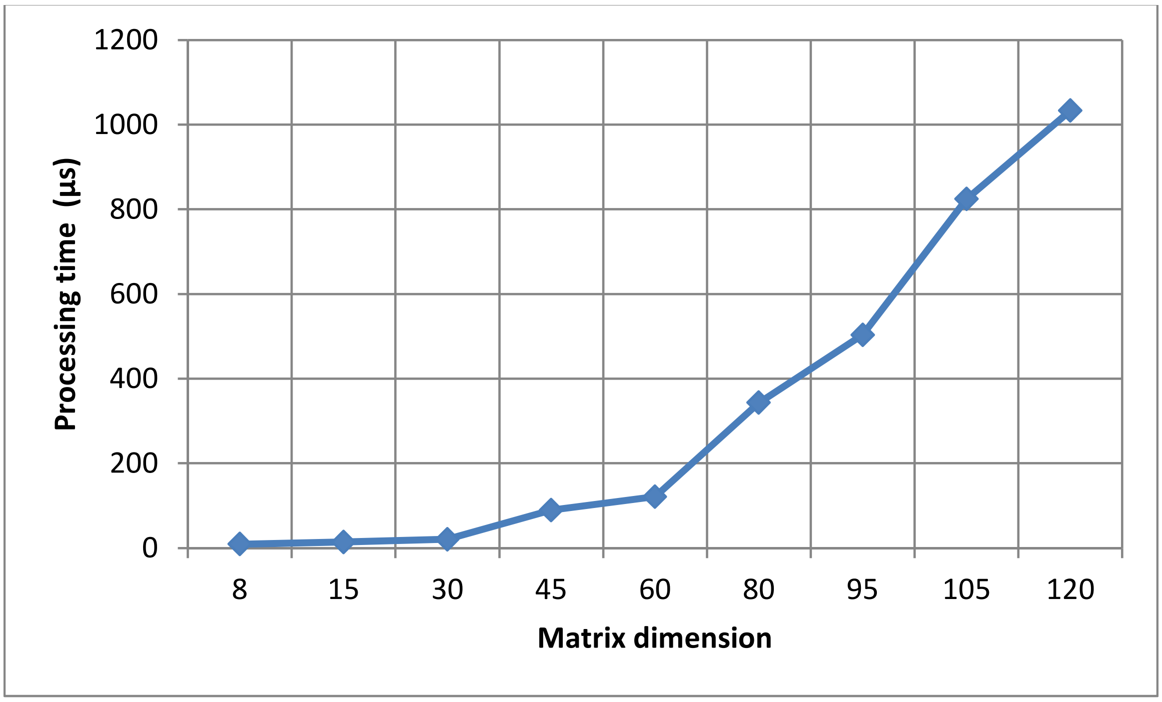

Figure 7. Dependence of processing time of number of processors calculations on dimension of data in two-processor system.

Table 2. Dependence of processing time on the size of the array of calculated data

Matrix Dimension	Processing Time (µs)
8	9
15	14
30	20
45	89
60	121
80	343
95	503
105	824
120	1033

From Table 2 and Figure 7 we concluded that with increasing the dimension of the data the processing time increased. To perform calculations more efficiently it wasnecessary to increase the number of processors, as shown in Figure 6.

From the results of experiments we concluded that the execution time of the software algorithm increasedwith decreasing the number of computing devices. For a single-processor system, which is therefore not distributed, the computation time wasthe longest.

4.2. Comparison of MPIApplication Efficiency for Distributed Computation for Different Data Volumes

We conducted experiments to calculate the QR–schedule of the matrix for matrices of different sizes, using different numbers of process devices in a distributed system.

1. We generateda matrix of size 20 rows by 20 columns.

   This amount of data was considered small.
   Each element of the matrix wasof type double (8 bytes). Then the total amount was:

$$C = 8{\cdot}20{\cdot}20 = 3200 \text{ Byte} = 3.2 \text{ Kbyte} \tag{41}$$

   The number of processors varied sequentially from 1 to 32.
   The results of the experiment appear in Figure 8.

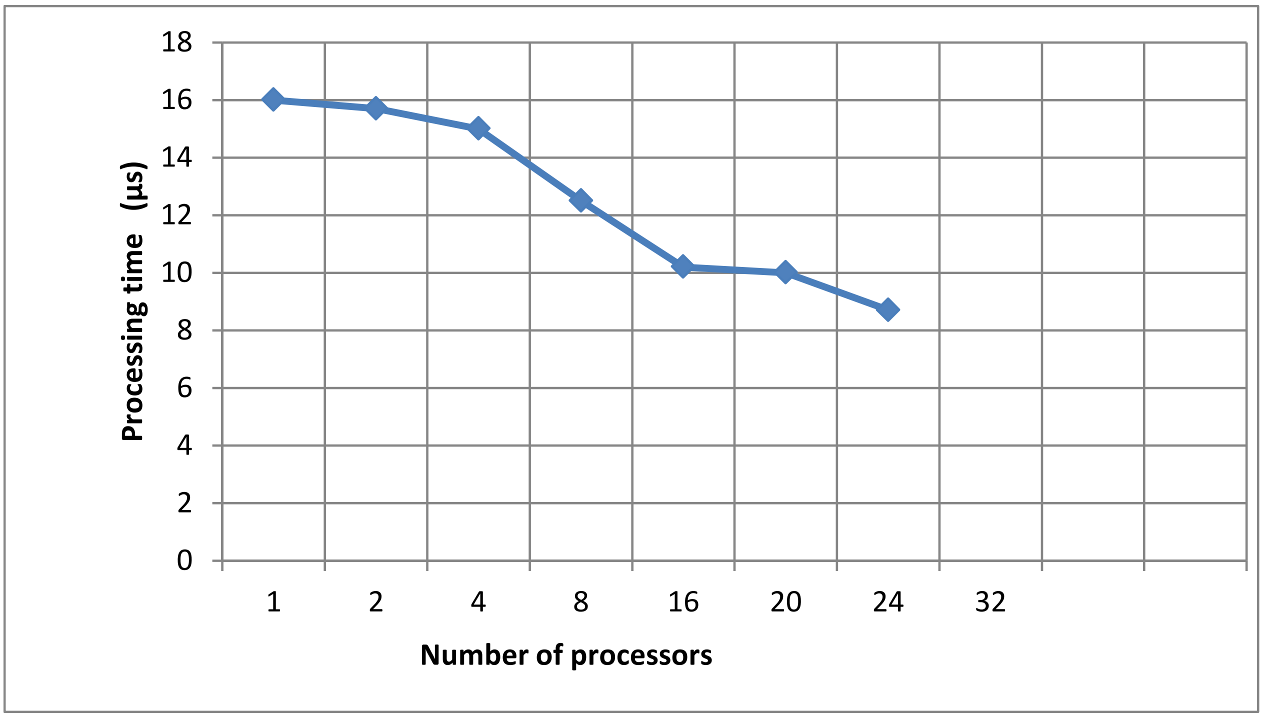

Figure 8. Dependence of SVD processing time on the number of computing processors at 20 × 20 matrix size.

From the graph we concluded that as the number of processors increased, the processing time decreased.

2. We increased the amount of data. To calculate, we took a matrix of rank 200 × 200. The amount of information:

$$8{\cdot}200{\cdot}200 = 320,000 \text{ Byte} = 320 \text{ Kbyte}. \tag{42}$$

The result is shown in Figure 9. We concluded that the processing time increased significantly. However, as the number of processors increased, the time was reduced.

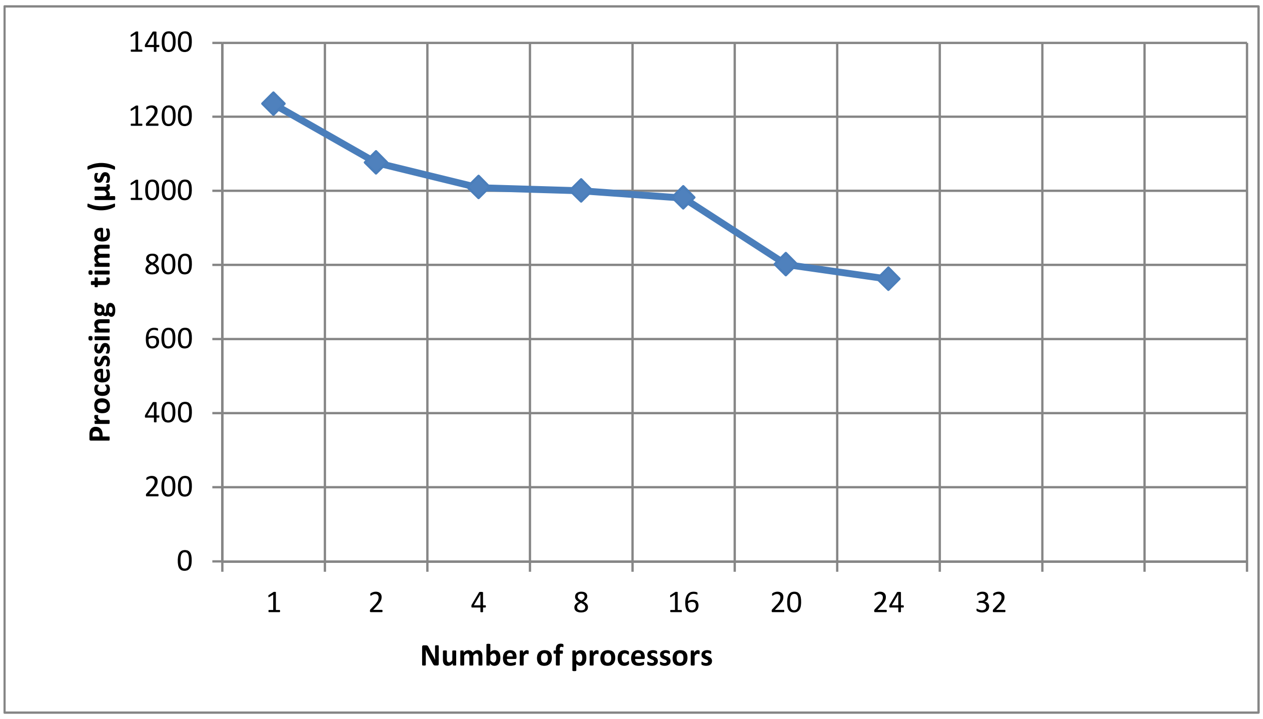

Figure 9. Dependence of SVD processing time on the number of computing processors at a matrix size of 200 × 200.

3. To calculate, we took a matrix of rank 1500 × 1500. The amount of information: 8 × 1500 × 1500 = 18,000,000 bytes = 18 MB (Figure 10).

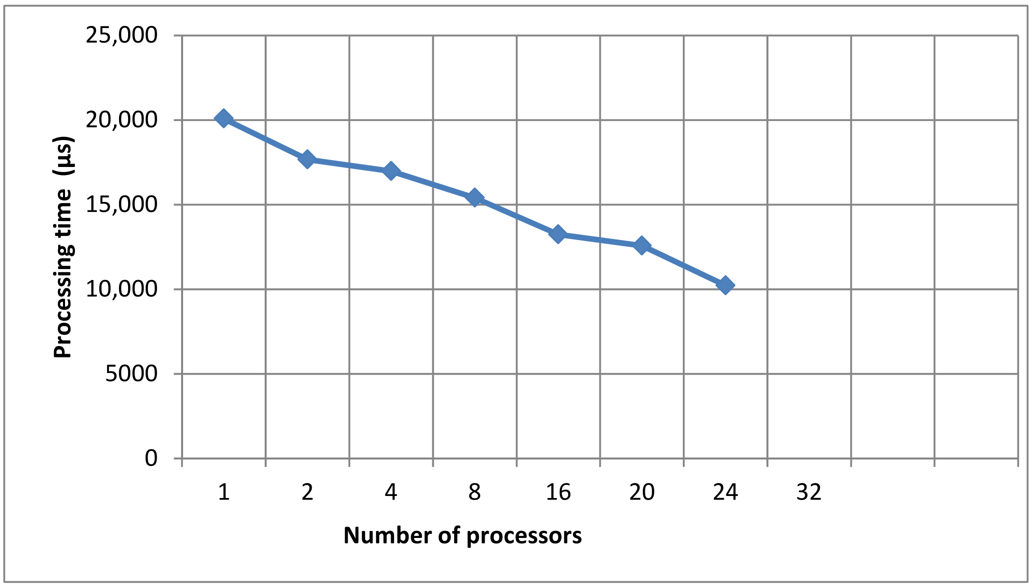

Figure 10. Dependence of SVD processing time on the number of computing processors at a matrix size of 1500 × 1500.

The graph shows that the computation time decreasedwith increasing the number of processors.

General results for the experiments are shown in Table 3.

Table 3. Dependence of processing time on the number of computational processes and the size of the matrix.

Number of Processors	**Matrix Dimension (n × n), n**		
	1500	200	20
	Processing Time (µs)		
1	20,286	1564	18.1
2	20,076	1234	16
4	17,654	1076	15.7
8	16,976	1009	15
16	15,400	1000	12.5
20	13,234	981	10.2
24	12,567	801	10
32	10,213	762	8.7

From the results of the experiments we concluded that with increasing data volumes, the computation time wasconstantly increasing. In order to increase the productivity of the system, it wasnecessary to carry out distributed information processing. For example, computing a two-dimensional data array represented as a rank 20 matrix on a conventional unallocated processor tookmore than 20 ms. When using a distributed system consisting of 32 processors, processing time washalved. Similar figures couldbe observed for other volumes of input data. Thus, for a matrix size of 200 × 200, the processing time wasreduced from 1564 µs on one processor to 762 µs on 32. For the largest data array of rank 1500, it wasalso possible to halve the processing time by using a distributed system with many computing devices. From the research we concluded that for the processing of large data sets it wasnecessary to use distributed computing systems, with a significant number of processors that provided the required performance of the computing system.

5. Dependence of Data Processing Efficiency on Hidden Factors

To form recommendations, it is often not enough to calculate the Singular Value Decomposition of the preference matrix. It is necessary to take into account certain initial conditions, in case the system does not yet have enough information to form a result. Adjustment coefficients are also added to the result of the work in order to take into account possible changes in preferences and to ensure the possibility of automatic training of the system of forming recommendations.

A different number of singular numbers were selected for the modified distributed algorithm. As a result, we found a relationship between computational speed and accuracy of the algorithm.

Modeling of the modified algorithm for additional reduction of time of giving of recommendations was carried out. To achieve that we changed the number of singular numbers. The results of research showed the possibility to improve the efficiency of the SVD algorithm if it wasnot necessary to take into account such indicators as, for example, the user's affiliation to a particular profession, place of residence or other factorthat increasedthe precision of personal recommendations.

The dependence of the processing time of calculations in the system (using MPI) of the matrix of rank 120 on the control coefficients isshown in Figure 11.

The predictive regularization factor usedadditional information about the user to adjust the provision of recommendations after a certain time;the initial learning factor introducedcertain initial values to provide recommendations;the learning speed ratio allowedus to calculate data faster by considering more parameters that characterize the user.

Figure 11 shows that with increasing the values of the control coefficients of the formation of recommendations, which should increase the reliability of the results, the processing time increased.

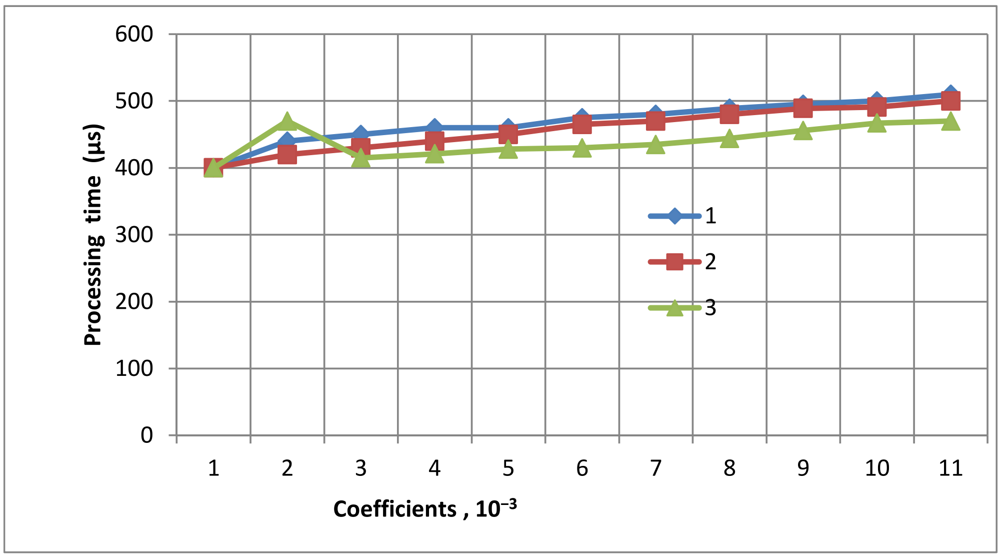

Figure 11. The dependence of the processing time of the calculations of the matrix of rank 120 on the coefficient of regularization of predictions (curve 1), the coefficient of initial learning (curve 2) and the coefficient of learning speed (curve 3).

Figure 11 shows that although the values of the coefficients wereinsignificant (from 0.001 to 0.01), the duration of the calculations increased from 0.4 to 0.5 ms. This result may increase with more data, so fewer factors need to be considered to improve computational speed.

Next, we considered the processing of singular numbers of data matrices. In the matrix Σ, which is an integral part of the Singular Value Decomposition of the matrix M, on the main diagonal are the singular numbers of the latter in descending order of value. To reduce the amount of computational data, it wassufficient to consider only the k largest singular numbers, and discard the rest. We couldalso take into account k the first rows of matrices U and V. When multiplying we obtained a matrix M1, which repeatedthe parameters of the initial M, but smaller in volume.

The dependence of the processing time of calculations of the matrix of rank 200 on the number of singular numbers isshown in Figure 12.

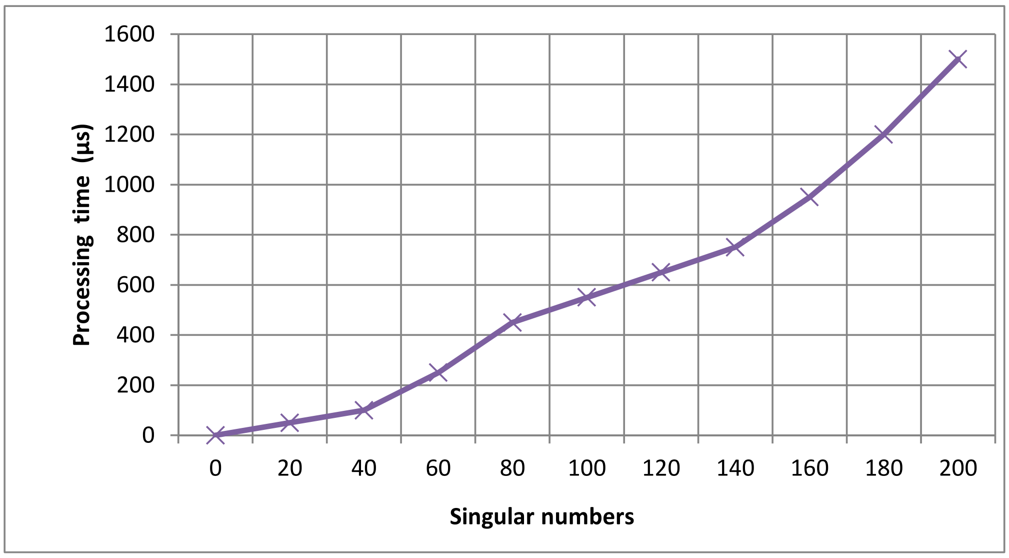

Figure 12. The dependence of the processing time of calculations of the data matrix of rank 200 on the number of singular numbers.

From the graph we concluded that the data processing time of the calculations decreasedwith increasing the number of rejected singular numbers.

Figure 12 shows that when the number of singular numbers was 20, the duration of the calculations was 0.05 ms. When the number increasedto 200, the computation time increasedto 1.5 ms.

The results showed that when using many singular numbers, the duration of the calculations increases, but this leads to an increase in the accuracy of the algorithm.However, for the effective operation of the method for the formation of recommendations, it is advisable to get rid of as few as possible with values close to zero.

It should be noted that when using a distributed SVD algorithm wecoulduse a larger number of singular numbers for the same duration of calculations (Figure 13).

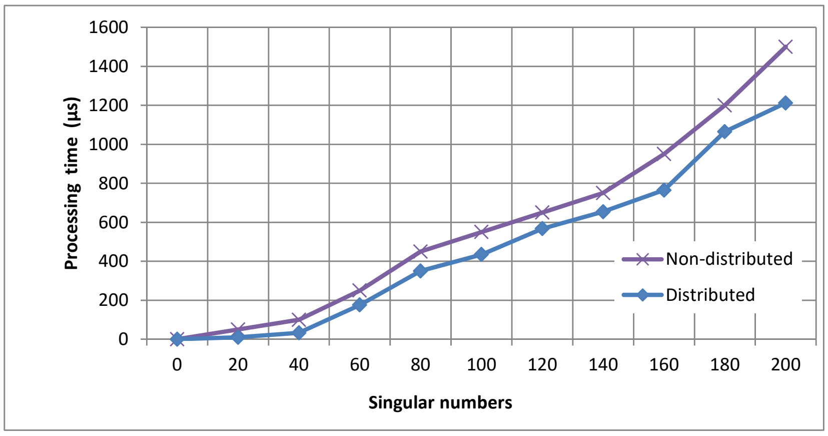

Figure 13. The dependence of the processing time of calculations of the data matrix of rank 200 on the number of singular numbers in distributed and non-distributed systems.

Figure 13 shows the durations of calculating the same number of singular numbers for the unmodified SVD algorithm and one modified using MPItechnology.

The unmodified algorithm completed the work taking into account the maximum number of additional parameters for 1.5 ms. The modified algorithmprocessedthis amount of data in approximately 1.2 ms.

Therefore, the proposed modification of the SVD algorithm operating in distributed systems can be used as an effective way to increase the speed and accuracy of calculations.

Data Processing Using Hadoop and Spark Technology

This model of distributed computing of the modified method of SVD can be used to determine the feasibility of using a complex computer system to solve problems.

The input of the model provides the parameters of the amount of data to be processed, and the output is the processing time for different numbers of computing devices. For companies that depend on a quick response to changing user preferences, a few seconds are crucial to making recommendations. For example, a site user searches an online store for a specific product, specifying specifications and a reasonable price. The recommendation system, which analyzes its needs in 3 s and provides a similar and interesting proposal, increases the likelihood of selecting the right product. Instead, if it takes 20–30 s to form recommendations, the user, without finding many interesting suggestions, can simply leave the site without making a purchase. Such situations happen quite often, and a significant part of the profits of sales companies depends on them. This is especially important now that many stores are online.

Figure 14 shows the dependence of the processing time of calculation of data matrix size 1500 for different technologies of distributed data processing depending on the number of processors.

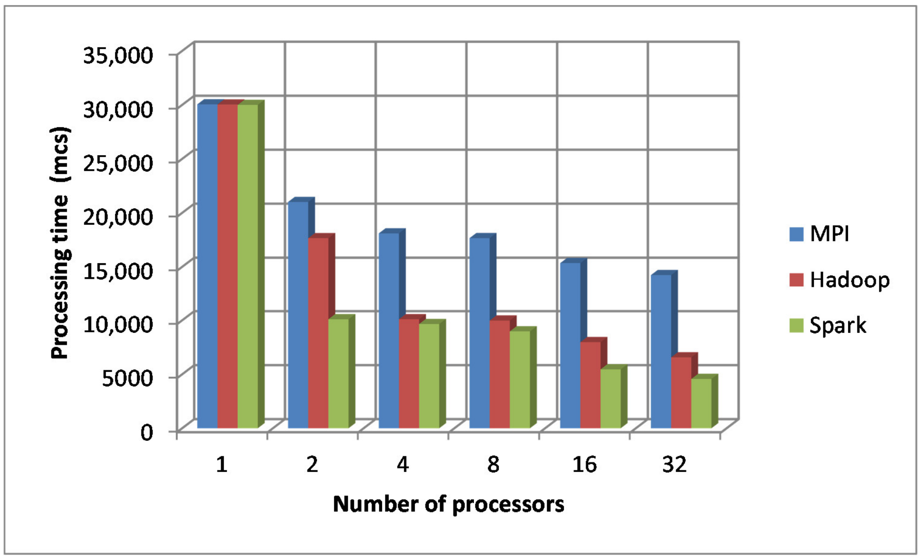

Figure 14. Processing time of SVD in the size 1500 for various technologies of distributed data processing versus number of processors.

From Figure 14 we concluded that with increasing the requirements for computing resources, it wasadvisable to use a distributed system with more processors. Spark technology wasthe most time-efficient.

When using one computational process, the computation time for all technologies was 30 ms. However, when the number of processors doubled, the MPItechnology calculateda similar amount of data in 20 ms, Hadoop in about 17 ms, and Spark in 10 ms.

When using 32 processors, we sawthat MPI, Hadoop and Spark technologies performed calculations for 13, 6 and 4 ms, respectively. This provedthat Spark technology performedthe calculation of large amounts of data the fastest of all the technologies considered in the study.

Figure 15 shows the dependence of the processing time of the calculation of recommendations to users depending on the dimension of the input data for distributed systems and distributed using different data processing technologies:

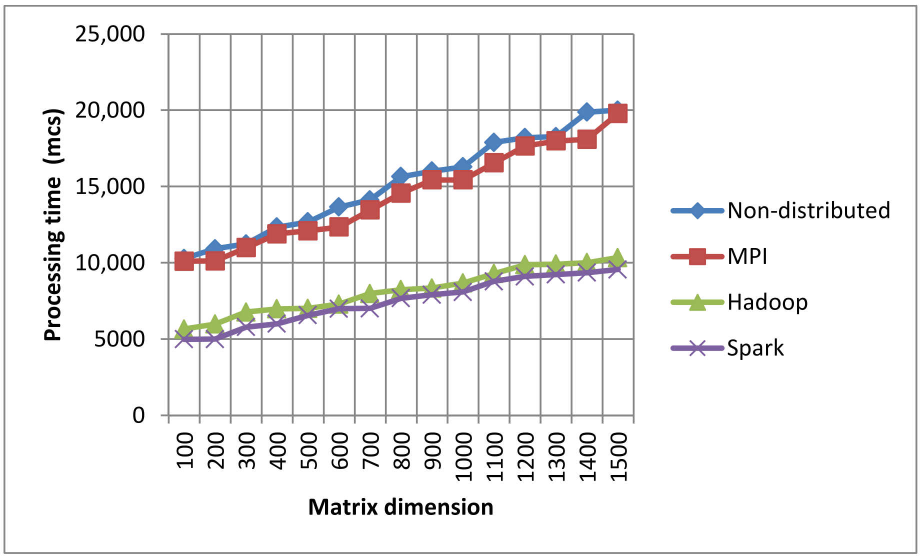

Figure 15. Dependence of the processing time of recommendations to users on the dimension of input data for unmodified SVD method (1); modified method based on MPI (Message Passing Interface) technology (2) modified method based on Hadoop technology (3); modified method based on Spark technology (4).

In Figure 15 we see that the SVD method, which workedwithout modification, showed aworse performance, compared to the modified method, which workedin distributed systems. To improve the efficiency of calculations, depending on the requirements for the processing time of calculations, it wasadvisable to use Hadoop and Spark technologies.

According to Figure 15 we concluded that the Hadoop and Spark technologies weremost effective when used in distributed systems with large amounts of data. For these technologies, the calculation time of the data matrix of rank 1500 didnot exceed 10 ms. Instead, for MPI technology, the computation time was 20 ms.

The results showed that Spark demonstratedthe greatest efficiency among the studied technologies of distributed computing. The advantage over Hadoop in the duration of calculations wasan average of 0.5 ms. For recommendation systems this determines the duration of processing a piece of user data. There are often many such fragments, so if the system needs to provide recommendations quickly for calculations it is suggested to choose Spark technology.

Figure 16 shows a relationship similar to that shown in Figure 15, but by analyzing the data obtained in the previous graphs, the duration of the calculations is for larger amounts of data. Thus, it is clear that it is necessary to use distributed computing systems to process large amounts of data.

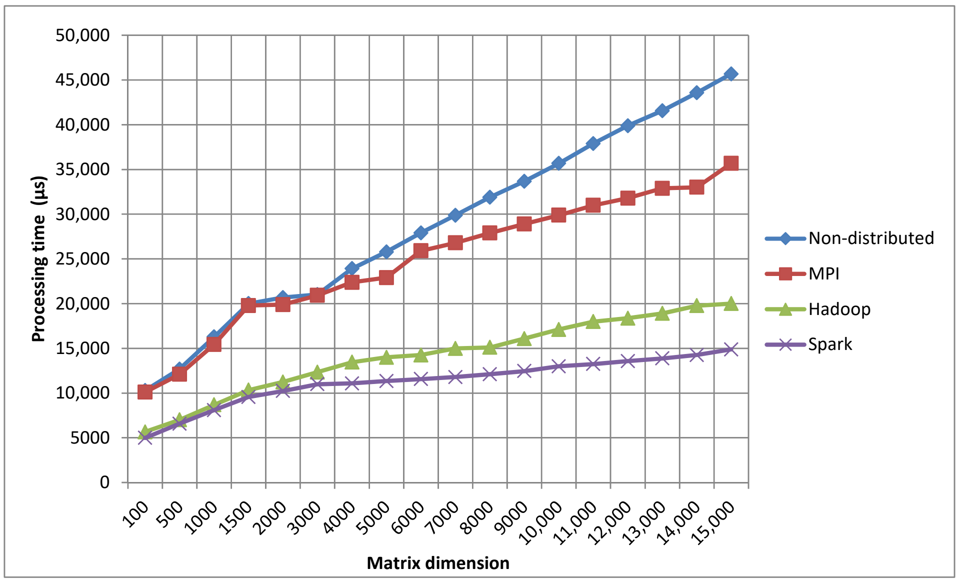

Figure 16. Dependence of the processing time of recommendations to users for large volumes on the dimensionality of the input data for the unmodified SVD method (1); a modified method based on MPI technology (2); a modified method based on Hadoop technology (3); a modified method based on Spark technology (4).

According to Figure 16, it was concluded that among the distributed technologies, MPI technology processeddata for the longest time (from 10 to 35 ms, depending on the amount of data). The data matrix with a maximum size of 15,000 was processedby Hadoop in 20 ms, Spark in 15 ms.

Figure 16 shows the relationship that allows us to conclude that to process large amounts of data it wasadvisable to use Spark technology. MPI and Hadoop technologies also hadadvantages over a conventional unallocated system. This dependence allowedto choose the optimal method of data processing on the basisof the required processing time and data volume. If it is economically profitable for a large company to spend resources on a powerful computing system and quickly process data, then the best choice of technology is Spark. Compared to an undistributed system, a system using Spark technology can calculate data 2–3 times faster.

The study used a volume of data that could not be processed on the equipment used in the experiment without modifying the data processing method and using distributed computing. Therefore, we can say that for this study the work was done with Big Data. The results obtained can be used to continue research on more powerful equipment and improve the efficiency of Big Data calculations.

From the simulation results we can also conclude not only about the comparison of unallocated and distributed systems using the algorithm SVD but also about the analysis of work in different architectures of distributed systems. The number of computational processes has changed in the modeling process. The results showed that the modified algorithm gives the most advantages in distributed systems with a large number of computing devices and a significant load on them. In addition to changing the number of processors we used various technologies: MPI, Hadoop, Spark. This allowed us to create several different distributed SVD algorithms with different results. Using the obtained research results it waspossible to predict the operation of the SVD algorithm in different distributed systems with different amounts of processed data. This also allowedto choose a distributed implementation of the algorithm that gave the best result for certain requirements and conditions in a particular system.

6. Discussion

In this work, a study was conducted on the processing of large volumes of data in reference systems. For this purpose, a modified method Singular Value Decomposition was used, which was proposed for use in distributed processing systems.

MPI, Hadoop and Spark technologies were used for the work method in distributed systems. Based on the obtained results, a model was built that allowedto determine the expediency of using distributed systems and the method of unambiguous decomposition was modified depending on the tasks being solved.

This method can be improved in the future by applying machine learning techniques, which will allow the proposed method to learn itself and give more accurate recommendations to users. This topic is also promising for further research and experimentation with new distributed data processing technologies and computing platforms in order to find optimal ways to improve the efficiency of processing large volumes of data.

7. Conclusions

The growth of data requires the introduction of new approaches to organizing calculations. In this regard, distributed and parallel processing technologies are widely used.

The paper consideredthe use of distributed infocommunication systems in the processing of large amounts of data. The work of recommendation systems of information analysis wasinvestigated. An overview of approaches to reduce computing using collaborative data filtering methods wasreviewed.

The introduction of methods for optimizing data sets in distributed systems wasinvestigated. An example wasthe method of SVD. The novelty of this approach wasthat it allowedusto organize information and discard redundant data, which led to faster and more efficient processing. System user information collected from various sources wassorted and analyzed. Based on that, recommendations wereformed on the services of the system that may be of interest to its customers. Because the data obtained from the recommendation systems werelarge, distributed computing was used to process them to ensure the required performance.

The operation of the SVD in the processing of large arrays of information wassimulated. The data size and number of nodes of the distributed system were changed using the MPI protocol to obtain optimal values of productivity. It wasdetermined that when the number of computing devices increased from 1 to 32, the computation time washalved for different amounts of data. The singular decomposition of the matrix of rank 1500 in the distributed 32–x processor system wasperformed in approximately 10 ms and in the unallocatedin 20 ms.

Simulation of SVD in systems using MPImessaging protocol and MapReduce software model wasperformed. The results showed that for small computations it is more efficient to use MPI, and when processing more than 10 MB of information it is advisable to use MapReduce due to simple implementation and faster operation.

We also compared the efficiency of Hadoop and Spark technologies, which indicated the feasibility of using the latter when it is necessary to quickly process large amounts of data. In particular, the use of a modified SVD method together with Hadoop and Spark technologies allowed to process data 2–3 times faster than when using an unmodified method in an unallocated system.

In the field of distributed systems, new research and improvements will always be relevant. This work can serve as a basis for further research in the field of processing large data sets using distributed or parallel computations.

Author Contributions: All authors contributed to the study conception and design. Methodology, K.P., M.B., O.H.-B., M.K., H.B., J.P. and J.S. Formal analysis and investigation, K.P., M.B., O.H.-B., M.K., H.B. and J.P. Writing of the original draft preparation, K.P., M.B., O.H.-B., M.K. and H.B. Review and editing, K.P., M.B., O.H.-B., and M.K. Funding acquisition, K.P. and J.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was financed in the framework of the project Lublin University of Technology—Regional Excellence Initiative, funded by the Polish Ministry of Science and Higher Education (contract no. 030/RID/2018/19).This research was supported by the project No. 0120U102201, “Development the methods and unified software-hardware means for the deployment of the energy efficient intent-based multi-purpose information and communication networks”, and by the project No.0120U100674, “Development of the novel decentralized mobile network based on blockchain-architecture and artificial intelligence for 5G/6G development in Ukraine”. This work was financed in the framework of the project Poznan University of Technology—contract no. 0613/SBAD/4677.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ortega, F.; González-Prieto, A. Recommender systems and collaborative filtering. *Appl. Sci.* **2020**, *10*, 7050. [CrossRef]
2. Shafqat, W.; Byun, Y.-C. Enabling “Untact” Culture via Online Product Recommendations: An Optimized Graph-CNN based Approach. *Appl. Sci.* **2020**, *10*, 5445. [CrossRef]
3. Son, Y.; Choi, Y. Improving Matrix Factorization Based Expert Recommendation for Manuscript Editing Services by Refining User Opinions with Binary Ratings. *Appl. Sci.* **2020**, *10*, 3395. [CrossRef]
4. Zhang, D.; Liu, L.; Wei, Q.; Yang, Y.; Yang, P.; Liu, Q. Neighborhood aggregation collaborative filtering based on knowledge graph. *Appl. Sci.* **2020**, *10*, 3818. [CrossRef]
5. Al-Ghuribi, S.M.; Noah, S.A.M. Multi-Criteria Review-Based Recommender System–The State of the Art. *IEEE Access* **2019**, *7*, 169446–169468. [CrossRef]
6. Yang, X.; Dong, M.; Chen, X.; Ota, K. Recommender System-Based Diffusion Inferring for Open Social Networks. *IEEE Trans. Comput. Soc. Syst.* **2020**, *7*, 24–34. [CrossRef]
7. Xiong, F.; Wang, X.; Pan, S.; Yang, H.; Wang, H.; Zhang, C. Social Recommendation with Evolutionary Opinion Dynamics. *IEEE Trans. Syst. Man, Cybern. Syst.* **2018**, *50*, 3804–3816. [CrossRef]
8. Nouh, R.M.; Lee, H.-H.; Lee, W.-J.; Lee, J.-D. A Smart Recommender Based on Hybrid Learning Methods for Personal Well-Being Services. *Sensors* **2019**, *19*, 431. [CrossRef]
9. Rabiu, I.; Salim, N.; Da’U, A.; Osman, A. Recommender System Based on Temporal Models: A Systematic Review. *Appl. Sci.* **2020**, *10*, 2204. [CrossRef]
10. Wang, Z.; Wu, H.; Jiang, Z.; Ju, P.; Yang, J.; Zhou, Z.; Chen, X. Singular value decomposition-based load indexes for load profiles clustering. *IET Gener. Transm. Distrib.* **2020**, *14*, 4164–4172. [CrossRef]
11. Hunold, S.; Carpen-Amarie, A. Reproducible MPI Benchmarking is Still Not as Easy as You Think. *IEEE Trans. Parallel Distrib. Syst.* **2016**, *27*, 3617–3630. [CrossRef]
12. Khan, M.; Jin, Y.; Li, M.; Xiang, Y.; Jiang, C. Hadoop Performance Modeling for Job Estimation and Resource Provisioning. *IEEE Trans. Parallel Distrib. Syst.* **2015**, *27*, 441–454. [CrossRef]

13. Yeromenko, V.; Kochan, O. The conditional least squares method for thermocouples error modeling. In Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS); Institute of Electrical and Electronics Engineers (IEEE), Berlin, Germany, 12–14 September 2013; Volume 1, pp. 157–162.
14. Sridharan, K.; Komarasamy, G.; Raja, S.D.M. Hadoop framework for efficient sentiment classification using trees. *IET Netw.* **2020**, *9*, 223–228. [CrossRef]
15. Hu, Z.; Li, D.; Guo, D. Balance resource allocation for spark jobs based on prediction of the optimal resource. *Tsinghua Sci. Technol.* **2020**, *25*, 487–497. [CrossRef]
16. Iannino, V.; Mocci, C.; Vannocci, M.; Colla, V.; Caputo, A.; Ferraris, F. An Event-Driven Agent-Based Simulation Model for Industrial Processes. *Appl. Sci.* **2020**, *10*, 434. [CrossRef]
17. Zhao, T.; Ding, Z. Distributed Agent Consensus-Based Optimal Resource Management for Microgrids. *IEEE Trans. Sustain. Energy* **2017**, *9*, 443–452. [CrossRef]
18. Beshley, M.; Kryvinska, N.; Seliuchenko, M.; Beshley, H.; Shakshuki, E.M.; Yasar, A.-U.-H. End-to-End QoS "Smart Queue" Management Algorithms and Traffic Prioritization Mechanisms for Narrow-Band Internet of Things Services in 4G/5G Networks. *Sensors* **2020**, *20*, 2324. [CrossRef] [PubMed]
19. Klymash, M.; Beshley, M.; Stryhaluk, B. System for increasing quality of service of multimedia data in convergent networks. In Proceedings of the 2014 First International Scientific-Practical Conference Problems of Infocommunications Science and Technology, Kharkiv, Ukraine, 14–17 October 2014; pp. 63–66.
20. Romanchuk, V.; Beshley, M.; Polishuk, A.; Seliuchenko, M. Method for processing multiservice traffic in network node based on adaptive management of buffer resource. In Proceedings of the 2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), Slavske, Ukraine, 20–24 February 2018; pp. 1118–1122.
21. Jun, S.; Przystupa, K.; Beshley, M.; Kochan, O.; Beshley, H.; Klymash, M.; Wang, J.; Pieniak, D. A Cost-Efficient Software Based Router and Traffic Generator for Simulation and Testing of IP Network. *Electronics* **2019**, *9*, 40. [CrossRef]
22. Handri, K.E.; Idrissi, A. Parallelization of Topk Algorithm through a New Hybrid Recommendation System for Big Data in Spark Cloud Computing Framework. *IEEE Syst. J.* **2020**. [CrossRef]
23. Chai, Z.; Li, Y.; Han, Y.; Zhu, S. Recommendation system based on singular value decomposition and multi-objective immune optimization. *IEEE Access* **2019**, *7*, 6060–6071. [CrossRef]
24. Ji, Y.; Hong, W.; Shangguan, Y.; Wang, H.; Ma, J. Regularized singular value decomposition in news recommendation system. In Proceedings of the 2016 11th International Conference on Computer Science & Education (ICCSE); Institute of Electrical and Electronics Engineers (IEEE), Nagoya, Japan, 23–25 August 2016; pp. 621–626.
25. Achakulvisut, T.; Acuna, D.E.; Ruangrong, T.; Körding, K.P. Science Concierge: A Fast Content-Based Recommendation System for Scientific Publications. *PLoS ONE* **2016**, *11*, e0158423. [CrossRef]
26. Li, G.; Hua, J.; Yuan, T.; Wu, J.; Jiang, Z.; Zhang, H.; Li, T. Novel Recommendation System for Tourist Spots Based on Hierarchical Sampling Statistics and SVD++. *Math. Probl. Eng.* **2019**, *2019*, 1–15. [CrossRef]
27. Guo, X.; Yin, S.-C.; Zhang, Y.-W.; Li, W.; He, Q. Cold Start Recommendation Based on Attribute-Fused Singular Value Decomposition. *IEEE Access* **2019**, *7*, 11349–11359. [CrossRef]
28. Chen, V.; Tang, T. Incorporating singular value decomposition in user-based collaborative filtering technique for a movie recommendation system: A comparative study. In *PRAI '19: Proceedings of the 2019 the International Conference on Pattern Recognition and Artificial Intelligence*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 12–15.
29. Ferreira, D.; Silva, S.; Abelha, A.; Machado, J. Recommendation System Using Autoencoders. *Appl. Sci.* **2020**, *10*, 5510. [CrossRef]
30. Hong-Xia, W. An Improved Collaborative Filtering Recommendation Algorithm. In Proceedings of the 2019 IEEE 4th International Conference on Big Data Analytics (ICBDA); Institute of Electrical and Electronics Engineers (IEEE), Suzhou, China, 15–18 March 2019; pp. 431–435.
31. Koprinarov, I.N.; Hitchcock, A.P.; McCrory, C.T.; Childs, R.F. Quantitative Mapping of Structured Polymeric Systems Using Singular Value Decomposition Analysis of Soft X-ray Images. *J. Phys. Chem. B* **2002**, *106*, 5358–5364. [CrossRef]
32. Al-Sabaawi, A.M.A.; Karacan, H.; Yenice, Y.E. Exploiting implicit social relationships via dimension reduction to improve recommendation system performance. *PLoS ONE* **2020**, *15*, e0231457. [CrossRef] [PubMed]
33. Cui, Z.; Xu, X.; Xue, F.; Cai, X.; Cao, Y.; Zhang, W.; Chen, J. Personalized Recommendation System Based on Collaborative Filtering for IoT Scenarios. *IEEE Trans. Serv. Comput.* **2020**, *13*, 685–695. [CrossRef]
34. Luo, H.; Li, M.; Wang, S.; Liu, Q.; Li, Y.; Wang, J. Computational drug repositioning using low-rank matrix approximation and randomized algorithms. *Bioinformatics* **2018**, *34*, 1904–1912. [CrossRef]
35. Chen, H.; Zhao, J.; Luo, Q.; Hou, Y. Distributed randomized singular value decomposition using count sketch. In Proceedings of the 2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC); Institute of Electrical and Electronics Engineers (IEEE), Shenzhen, China, 15–17 December 2017; pp. 187–191.
36. Sun, S.; Przystupa, K.; Wei, M.; Yu, H.; Ye, Z.; Kochan, O. Fast bearing fault diagnosis of rolling element using Lévy Moth-Flame optimization algorithm and Naive Bayes. *Ekspolatacja Niezawodn. Maint. Reliab.* **2020**, *22*, 730–740.
37. Wu, J. Simple technique to determine the Givens-rotation matrix in the two-source ICA problem for skewed sources. *Electron. Lett.* **2016**, *52*, 613–615. [CrossRef]