Algorithms for Placing Files in Tiered Storage Using Kohonen Map*

Tatiana M. Tatarnikova[1][0000-0002-6419-0072], Ekaterina D. Poymanova [2][0000-0002-7903-2480]

[1] Russian State Hydrometeorological University, 79, Voronelsraya st., 192007 St. Petersburg, Russia
tm-tatarn@yandex.ru
[2] Saint-Petersburg State University of Aerospace Instrumentation, 67, Bolshaya Morskaia str., 190000 St. Petersburg, Russia
e.d.poymanova@gmail.com

Abstract. The data storage task is not limited only to the allocation of volume for data placement. It needs a new storage resource management models with tiered storage and proactive migration of information. Present storage systems are still one-tier. The storage administrator decides about data migration to other media. The decision to migrate data is determined by the time that has passed since the last access to the data. However, many metrics can be taken into account, such as the rate at which the requested data is provided, the cost of data loss, the period through which information is transferred to another storage tier. The paper proposes a sequence of algorithms for distributing files in tiered storage. For the first, the algorithm of vertical placement files across storage tiers, next horizontal placement of files on physical and logical media, and then migration data algorithm. The result of algorithms applying is visualized in the form of a matrix, the size of which corresponds to the number of storage tiers and the number of physical or logical media. All storage resource management algorithms are based on the analysis of stored file metadata. The representation of the storage system in the form of a matrix allows using the Kohonen neural network tool to arrange files by levels and sections of a specific storage system level. Using Kohonen neural network allows you to move from sequential execution of algorithms to placement in one-step.

Keywords: Tier Data Storage, Data Storage System, Metadata, Efficient Data Placement, File Metadata, Data Migration, Clustering, Kohonen Neural Network.

1 Introduction

Data storage is a necessity for both an enterprise, a corporation, state structures, and a person. For enterprises and the corporate sector, the need to store large amounts of data is determined by existing business processes, in the public sector by the transition to interdepartmental electronic document management and the creation of departmental analytical resources. In addition, users who upload their photos to the Internet, videos and actively exchange multimedia content in social networks create a powerful data stream.

The engineering solution for the implementation of storage infrastructure is data storage systems. The data storage system mainly is segregate into a computing subsystem complex, for example, in a data center [1].

The data storage system is an architectural solution for connecting external data storage devices of different physical nature [2].

The main task in the design of storage infrastructure is the effective management of storage resources, mainly capacitive. Its solution is complicated by the following circumstances:

- storage heterogeneity: storage devices in a storage system can be different in physical nature (for example, magnetic, optical, solid-state) and in architecture (direct or network access storage systems) [3];
- different data storage requirements: critical transactional systems such as billing, processing, ERP, etc., require highly reliable and productive storage systems; analytical systems - high productivity and low cost per storage unit; for work with files – functionality and low cost of storage [4];
- the lack of efficient multi-level data storage algorithms with different storage requirements.

The paper describes the problem of organizing tiered data storage, which assumes using different storage technologies for different files depending on the guaranteed storage time.

Storage engineering solutions analysis allows distinguishing three tiers of the data storage architecture:

- RAID (Redundant Array of Independent Disks);
- automated libraries;
- long-term storage media.

Each storage tier involves its own storage technologies [1].

Despite the fact, that modern data storage systems (DSS) are multi-tier associations [2], the decision on the choice of the storage tier lies with the administrator. This decision is based on a single metric - the time elapsed since the last access to the information [3].

The purpose of the research is to develop a tiered data storage model and data distribution algorithms for storage systems that will partially solve the listed issues of data storage.

2 Tiered structured

It is proposed to distribute data files using the file metadata analysis.

The distribution process includes the following principles:

- the storage tier selection depending on data storage time;
- the storage local volume selection depending on the file size and the length of the logical data block;
- the principle of data migration across tiers depending on the frequency of data file access.

Before writing to a specific storage tier, it is necessary to analyze the data in order to select the optimal file system (FS) for the RAID tier or the type of archive media, which will allow saving storage space [5]. Thus, the data storage gets a matrix structure, containing data with certain characteristics in each cell (Fig. 1).

	FS 1	FS 2	...	FS *n*	
RAID	Volum 1	Volum 2	...	Volum 1	1st Tier
Automated libraries	Data Media 1	Data Media 1	...	Data Media 1	2nd Tier
Long term storage media	Data Media 1	Data Media 1	...	Data Media 1	3rd Tier

Fig. 1. The matrix structure of the data storage

The storage tier selection principle is based on the analysis of organizational metadata containing data type information:

- *ind* (*initial data*) – raw data that is placed on the RAID tier;
- *bck* (*backups*) – backups, archived data that is stored at the automated libraries tier;
- *ngd* (*next-generation data*) – data of unlimited storage that is stored in the long term storage tier.

The storage logic tier selection principle of the storage system is based on the selection of the file system for the RAID tier and the type of storage media on the lower storage tiers:

$$\text{If } f \in (f_i; f_{i+1}], \text{ then } \to a_{i+1} \Leftrightarrow F \to Vol_{i+1}, \tag{1}$$

where f – the size of the saved file F;
f_i, f_{i+1} – size limits of the file F, with which the file system can work;
a_{i+1} – the logical data block size with which the file system operates;
Vol_{i+1} – the number of a RAID volume that is managed by the corresponding file system.

At the lower storage tiers, it is proposed to divide the capacity according to the types of storage media: tape drive, DVD, BD for the automated libraries tier and M-disk, glass disk and DNA – for long-term storage media tier:

$$\text{If } f \in (f_i,\ f_{i+1}], \text{ then } F \to al_{i+1}\ (lt_{i+1}), \tag{2}$$

where al_{i+1} or lt_{i+1} – the type of media at the automated libraries tier (al) or at the long-term storage tier (lt).

The principle of data migration across storage tiers depends on the frequency of data files access:

$$\text{If } \lambda_F \in (\lambda_i, \lambda_{i+1}], \text{ then } F \to l, \tag{3}$$

where λ_F – the frequency of the file F request;
λ_i, λ_{i+1} – limits of the file request frequency;
l – the number of the storage tier to which the file F is migrated.

The migration principle allows overcoming the shortcomings of a subjective selection of the saving files type when implementing the first stage - data recording.

The complex of principles above allows managing storage capacity and making rational use of media. Note that all these principles are based on the file's metadata analysis.

The result of the proposed storage capacity management principle, in general, is the storage matrix of size $m \times n$, where m is the number of storage tiers (M), and n is the number of physical or logical media (N). Elements of the matrix are sets of data files with certain characteristic values: file type (*type*), file size (f) (Fig. 2). During the initial data distribution, the data access frequency (λ) is not taken into account, because of absence of accumulated statistics of data access at this moment.

	N_1	...	N_n
M_1	$type_1, f_1$	...	$type_1, f_n$
M_2	$type_2, f_1$	...	$type_2, f_n$
M_3	$type_3, f_1$	...	$type_3, f_n$

Fig. 2. Storage matrix

Based on the storage technologies analysis, three storage tiers were identified. Accordingly, the matrix will always contain three rows. The number of columns is selected based on the physical implementation of each data storage tier.

A block diagram of aggregate algorithms for placing files in the tiered storage is shown in Fig. 3

The consistent implementation of the presented above principles requires high-energy costs. In this regard, to distribute files among the cells of the matrix (Fig.1), it is proposed to analyze the metadata of the input file stream using Kohonen neural networks. Kohonen trained a neural network that is capable of solving this problem not by the consistent use of principles, but in one-step [6]. It is noteworthy that the neural network, in this case, is used precisely as a principle for distributing data to the cells of the storage matrix.

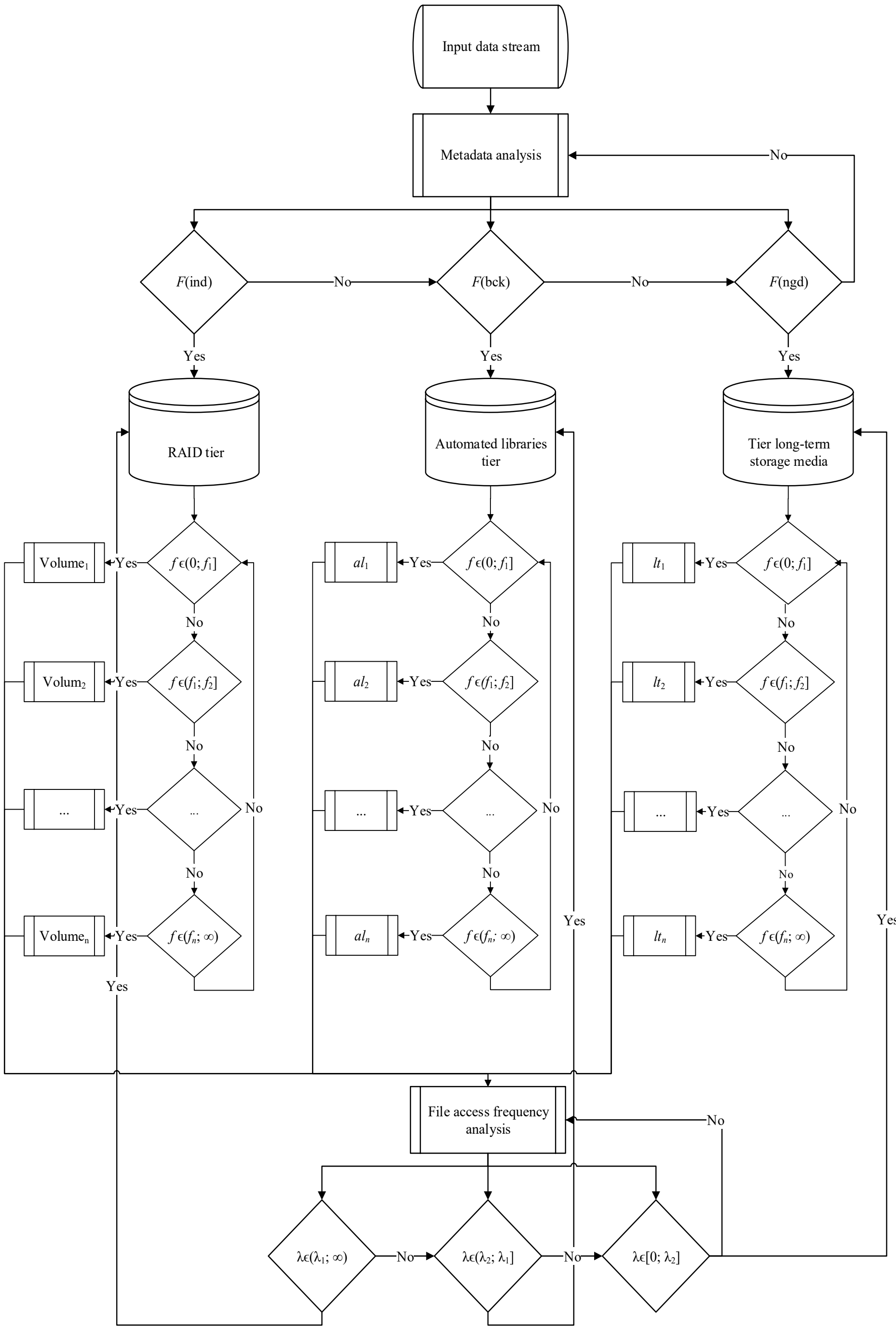

Fig. 3. Algorithm of file allocation in tiered data storage

The result of using a neural network is a topological map in which the input data is classified into groups (clusters). Thus, each cell of the resulting map must correspond to a cell of the storage matrix.

3 Kohonen Network as a Data File Clustering Tool

Kohonen neural network, unlike many other types of neural networks, is trained without a teacher.

The main purpose of the Kohonen network is to solve the problem of cluster analysis (clustering).

The Kohonen network includes two layers: input and output. Each neuron of the input layer is connected to each neuron of the output layer and all connections have a certain weight, which is corrected in the learning process. The output layer is also called the topological map layer. Neurons of a topological map are scattered in a two-dimensional field (Fig. 4) [6].

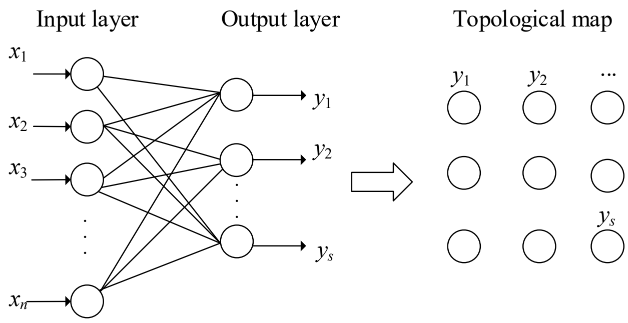

Fig. 4. Kohonen network architecture

Kohonen maps have a set of input elements, the number of which coincides with the dimension of the input vectors, and a set of output elements, each of which corresponds to one cluster (group).

The essence of the Kohonen network is as follows. When inputting some vector **X** to the input, the network must determine to which of the clusters this vector is closest. As the proximity criterion can be chosen the metric of the square of the Euclidean distance

$$d_{ij} = \sum_{k=0}^{n} \left(x_{ik} - y_{jk} \right)^2, \tag{4}$$

where d_{ij} is the squared distance between point **X** and cluster **Y**. The coordinates of point **X** are $x_{i1}, x_{i2}, \ldots, x_{in}$, and the coordinates of the cluster center Y are $y_{j1}, y_{j2}, \ldots, y_{jn}$.

Vector **X** is a point in n-dimensional space, where n is the number of vector coordinates that are fed to the input of the neuron network. Calculating the distance between this point and the centers of different clusters allows you to determine the cluster, the distance to which will be minimal. Then this cluster and the corresponding output neuron is declared the winner.

The cluster center is defined as a point whose coordinates in n-dimensional space are the weights of all connections that arrive at a given output neuron from the input neurons.

Kohonen maps solve the clustering problem as follows: by submitting a vector to the network input, one winning cluster will be obtained, which determines the membership of the input vector to this cluster.

Kohonen network learning algorithm is recurrent. Any training example is processed in this sequence [6]:

- take the winner neuron, that is, the one that is closer to the entered example;
- perform correction of the winning neuron so that it becomes as close as possible to the entered example, finding the square of the Euclidean distance from the center of the winning neuron to the learning example.

When calculating the center of the winning neuron, uses the learning rate factor. This coefficient gradually decreases in such a way that, at each new stage, the correction of the weights became more and more close to the given one. As a result, the location of the center will be established in some position that best reflects the examples for which this neuron is the winner.

Because of such a recurrent learning procedure, the neural network will be organized so that neurons located close to each other in the space of the input layer will be located close to each other and on a topological map.

The completion of the Kohonen map is based on the concept of a neighborhood.

The neighborhood is characterized by a radius *R*. It is a group of neurons that surround the winner-neuron (Fig. 5). Similarly, the learning rate, the radius of the neighborhood decreases with the learning time in such a way that a significant number of neurons are located in the neighborhood first (perhaps almost everything located on the topological map). Then, in the final stages, the neighborhood tends to zero – it includes only the winner-neuron itself (Fig. 5).

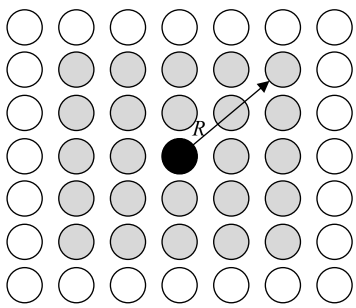

Fig. 5. Cluster neighborhood

If changes in weights become insignificant, then the training ends.

After the network has been trained to solve the clustering, we use it as a visualization tool in the form of a Kohonen map [7].

The Kohonen network learning algorithm has the following sequence of steps:

Step 1. For the input vector **X**, calculate the coordinates of the neurons of the output layer, calculate d_{ij} from **X** to each of the neurons of the network.

Step 2. Find the minimum d_{ij} from the obtained values, determine the winner neuron.

Step 3. For the winning neuron, as well as for those neurons that are in a neighborhood with radius R, perform the adjustment of the weights:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta(t)\left(x_i - w_{ij}(t)\right), \tag{5}$$

where $w_{ij}(t+1)$ is the weight at step $(t + 1)$;

$w_{ij}(t)$ – weight at step t;

Δ – learning rate;

x_i – coordinate of the input vector.

Step 4. Update the values of the learning rate and radius R of the neighborhood.

Step 5. Continue learning until the condition for stopping learning is fulfilled.

Learning stops when weights become insignificant.

The choice of this method for solving the file allocation is due to the peculiarities of the algorithm that implements the Kohonen network:

1. Uncontrolled learning is used, in which the learning rule of a neuron is based on information about its location;
2. There are no reference values of the training set;
3. The result of the algorithm is a topological map, in which the input data are classified into groups (clusters).

In the case of the proposed distribution of files in the data storage cells, it is obvious that the selected file characteristics define the feature value vectors. Thus, each cell of the resulting map must correspond to an element of the matrix representing the storage system [8].

4 Experiment Description and Analysis of Results

The experiment involved 5,000 files with different characteristics: the type of file, its estimated storage time, file size, frequency of accessing data.

The stream is generated based on the analysis of 43,000 files taken from an experimental file server [9], [10]. The file size does not exceed 1 GB, the frequency of accessing data is a random variable in the interval [0–300] requests per hour. The frequency of accessing data was considered in absolute units - the number of requests per hour.

The data files distribution was implemented in accordance with the structure of the storage matrix of size 3×3.

Normalization of file types was adopted based on the considerations of obtaining disjoint classes: the file type *ind* corresponds to the value 1, *bck* – 2, *ngd* – 3.

Normalization of the file size is done in decimal order: if the file size is from 0 to

999 B, we assign the value of the attribute 1; from 1000 to 999,999B – 2; from 1,000,000 to 999,999,999B – 3.

The results of the experiment show that the Kohonen neural network apparatus can be a principle for solving the problem of distributing files with different characteristics and storage time requirements. The main difficulty is the choice of classification parameters and their normalization. An example of the Kohonen map in 3D, which is built as a result of the experiment, is shown in Fig. 6.

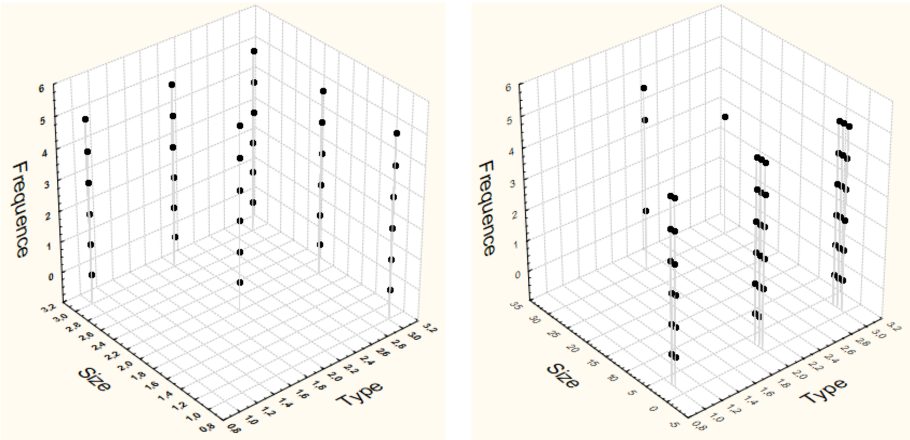

Fig. 6. The results of the experiment file distribution depending on the type, size, and frequency of requests (the Kohonen maps in 3D)

The decision of the necessity of data files migration to another storage tier is based on analyzing the value of the files accessing frequency. The storage administrator should determine the limit values of the request frequency, upon which the files are migrated (Fig. 7).

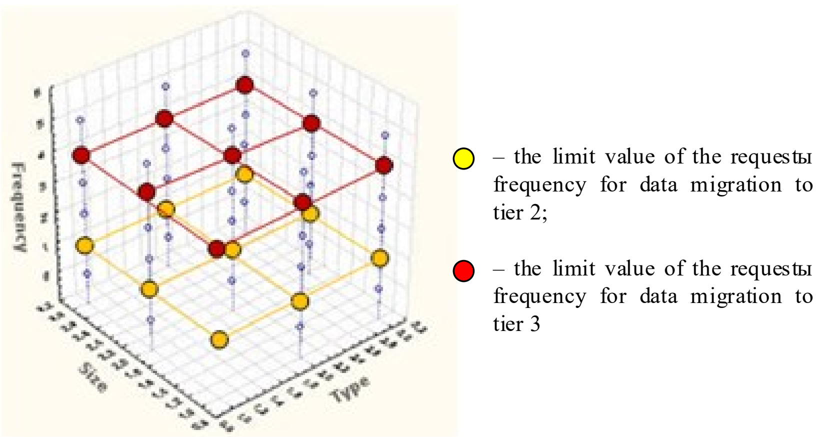

Fig. 7. An example of visualization of the limit values of the files access frequency

The results of the experiment showed that the Kohonen neural network can be a tool for solving the problem of placing files.

5 Conclusion

The paper suggests a tiered data storage model. The file distribution in data storage system is implemented in accordance with the consistent use of principles for vertical, horizontal placement and migration of data.

The initial vertical and horizontal distribution of files in the tiered data storage system is formalized in the form of a matrix. Such a presentation allows using the Kohonen neural network apparatus, whose main purpose is to solve the clustering problem. In the problem of the distributing file, clusters correspond to cells of the storage matrix. Before solving the clustering problem, metadata that sets the characteristics of the saved files were normalized.

Using the Kohonen neural network allows us to abandon the consistent implementation of distribution algorithms, and solve the problem of file distribution in one step.

References

1. Information Storage and Management. 2nd Edition. New Jersey: John Wiley & Sons Inc., (2016).
2. Mesnier M., Ganger G., Riedel E. Object-Based Storage. IEEE Communications Magazine, vol. 41, no. 8, 84-90 (2003).
3. Farley M. Building Storage Networks. Osborne: McGraw-Hall, (2001).
4. Sovetov B. Ya., Tatarnikova T. M., Poymanova E. D. Organization of multi-level data storage. Informatsionno-Upravliaiushchie Sistemy, no. 2, 74-81 (2019). DOI: 10.31799/1684-8853-2019-2-68-75
5. Bogatyrev V.A, Bogatyrev A.V., Bogatyrev S.V., Polyakov V. I. Redundant service of request copies by a sequence of groups of reserved nodes. In CEUR Workshop Proceedings: Workshop International Scientific and Technical Conference "Fundamental and practical aspects of computer technology&quot, FPACT 2018, 5-7 Nov. 2018, Saint Petersburg, IET - 2019, pp. 135-140.
6. Kohonen T. Self-Organizing Maps. New York, (2001).
7. Stacey M., Salvatore J., Jorgensen A. Visual Intelligence: Microsoft Tools and Techniques for Visualizing Data. New Jersey, John Wiley & Sons, (2013).
8. Morville P., Callender J. Search Patterns: Design for Discovery. O’Reilly, (2010).
9. Poymanova E.D., Tatarnikova T.M. Models and Methods for Studying Network Traffic. In Wave Electronics and its Application in Information and Telecommunication Systems, WECONF 2018, pp. 1-5, (2019). DOI: 10.1109/WECONF.2018.8604470
10. Tatarnikova T. M., Volskiy A.V. Estimation of probabilistic-temporal characteristics of network nodes with traffic differentiation. Informatsionno-Upravliaiushchie Sistemy, no. 3, 54-60 (2018). DOI: 10.31799/1684-8853-2019-2-68-75.