УДК 004.07
doi:10.31799/1684-8853-2019-2-68-75

Научные статьи
Articles

Организация многоуровневого хранения данных

Б. Я. Советов[а], *доктор техн. наук, профессор, orcid.org/0000-0003-3116-8810*
Т. М. Татарникова[б], *доктор техн. наук, доцент, orcid.org/0000-0002-6419-0072, tm-tatarn@yandex.ru*
Е. Д. Пойманова[б], *старший преподаватель, orcid.org/0000-0002-7903-2480*
[а]*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ», Профессора Попова ул., 5, Санкт-Петербург, 197376, РФ*
[б]*Санкт-Петербургский государственный университет аэрокосмического приборостроения, Б. Морская ул., 67, Санкт-Петербург, 190000, РФ*

Введение: *в условиях увеличивающегося объема данных практически все организации нуждаются в формировании развитой инфраструктуры их хранения. Инфраструктура реализуется с применением технологий многоуровневого хранения — проектированием систем хранения данных, включающих устройства хранения, разнородные по физической природе и архитектуре доступа. К файлам также предъявляются требования по соблюдению сроков гарантированного хранения, в том числе на законодательном уровне. Отсутствие алгоритмов эффективной организации хранения данных с разными требованиями к хранению актуализирует необходимость новых моделей управления ресурсами системы хранения данных.* ***Цель:*** *разработка новых моделей и алгоритмов управления ресурсами систем хранения данных с учетом многоуровневого принципа хранения данных.* ***Результаты:*** *предложен алгоритм многоуровневого хранения данных, позволяющий выполнять распределение файлов в системе хранения данных в соответствии с последовательным применением механизмов вертикального, горизонтального и динамического размещения файлов. Алгоритмы размещения и миграции данных по уровням иерархии системы хранения данных позволяют применять анализ метаданных, что дает возможность реализовать проактивное управление ресурсами систем хранения данных с учетом сроков гарантированного хранения. Представлена модель хранения данных в виде матрицы, размер которой соответствует количеству вертикальных и горизонтальных уровней системы хранения данных. Рекомендовано проводить распределение файлов по ячейкам матрицы с использованием аппарата нейронных сетей Кохонена. Эффект применения сети Кохонена заключается в возможности анализа метаданных и частоты обращения к файлам одним этапом. Это позволяет отказаться от последовательного выполнения механизмов размещения. Результаты эксперимента показали действенность аппарата сетей Кохонена при решении задачи размещения файлов в многоуровневой системе хранения данных.* ***Практическая значимость:*** *предложенные модель и алгоритмы могут найти применение при проектировании систем хранения данных с учетом многоуровневого принципа хранения данных.*

Ключевые слова *— система хранения данных, многоуровневое хранение данных, время хранения файла, метаданные, миграция данных, нейронная сеть Кохонена.*

Для цитирования: Советов Б. Я., Татарникова Т. М., Пойманова Е. Д. Организация многоуровневого хранения данных. *Информационно-управляющие системы*, 2019, № 2, с. 68–75. doi:10.31799/1684-8853-2019-2-68-75
For citation: Sovetov B. Ya., Tatarnikova T. M., Poymanova E. D. Organization of multi-level data storage. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2019, no. 2, pp. 68–75 (In Russian). doi:10.31799/1684-8853-2019-2-68-75

Введение

Необходимость в хранении данных можно встретить на уровне отдельного предприятия, корпорации, государственных структур [1].

Для предприятий и корпоративного сектора потребность хранить большой объем данных определяется действующими бизнес-процессами, в государственном секторе — переходом к межведомственному электронному документообороту и созданием ведомственных аналитических ресурсов. Не менее мощный поток данных создают и обычные пользователи, которые выкладывают в Интернет свои фотографии, видеоролики и активно обмениваются мультимедийным контентом в социальных сетях [2].

Инженерным решением реализации инфраструктуры хранения являются системы хранения данных (СХД), которые в основном выделяют в отдельную подсистему вычислительного комплекса, например, центра обработки данных [3].

Система хранения данных — это архитектурное решение для подключения внешних устройств хранения данных разной физической природы [4].

Основной задачей при проектировании инфраструктуры хранения данных является эффективное управление ресурсами СХД, в основном емкостными. Ее решение осложняется наличием следующих обстоятельств:

— разнородностью СХД: устройства хранения в СХД могут быть разными по физической природе (например, магнитные, оптические, твердотельные) и по архитектуре (СХД прямого или сетевого доступа) [5, 6];

— разными требованиями к хранению данных: критически важным транзакционным системам типа биллинговые, процессинговые, ERP и т. п. требуются высоконадежные и производительные

СХД; аналитическим системам — высокая производительность и низкая стоимость в расчете на единицу хранения; для работы с файлами — функциональность и низкая стоимость хранения [7];

— отсутствием алгоритмов эффективной многоуровневой организации хранения данных с разными требованиями к хранению. Современные СХД по-прежнему остаются одномерными, поскольку перемещение данных между уровнями СХД определяется одной метрикой — временем, прошедшим с момента последнего обращения к информации [8, 9].

В статье приводится предложенный авторами алгоритм распределения данных в СХД с многоуровневой организацией, учитывающий типы файлов данных, нормативные сроки их хранения и частоту обращения к ним.

Алгоритм организации многоуровневого хранения данных

Исходя из многоуровневой архитектуры СХД будем считать, что каждый уровень структуры предполагает свои технологии хранения: RAID-массивы, автоматизированные библиотеки и носители длительного хранения [10–12].

Предлагается распределять файлы данных по уровням СХД в соответствии с последовательным применением следующих механизмов:

— выбора уровня СХД в зависимости от времени хранения (вертикальное размещение);

— выбора логического тома уровня хранения СХД в зависимости от размера файла и длины логического блока данных (горизонтальное размещение);

— миграции данных по уровням СХД в зависимости от частоты обращения к ним (динамическое размещение).

Таким образом, СХД можно представить в виде матричной структуры, где каждая ее ячейка — это том соответствующего уровня СХД, предназначенного для хранения данных с определенными метрическими характеристиками (рис. 1).

RAID	$ФС_1$	$ФС_2$	...	$ФС_n$
Автоматизированные библиотеки	Носитель 1	Носитель 2	...	Носитель *n*
Носители длительного хранения	Носитель 1	Носитель 2	...	Носитель *n*

■ ***Рис. 1.*** Матричная структура СХД

■ ***Fig. 1.*** The structure of the storage system in the form of a matrix

Идея механизма выбора уровня СХД (вертикальное размещение файлов) основана на анализе организационных метаданных, содержащих сведения о типе данных. Например, можно указывать тип в имени файла через точку перед расширением *F.bck.txt*. Таким образом, при сохранении необходимо указывать атрибут файла *F*:

ind (*initial data*) — исходные данные, которые размещаются на уровень RAID;

bck (*backups*) — резервные копии, архивные данные, которые будут храниться на уровне автоматизированной библиотеки;

ngd (*next generation data*) — данные бессрочного хранения.

Во вторую очередь происходит горизонтальное распределение файлов. Идея горизонтального размещения основана на выборе файловой системы (ФС) для уровня RAID и по типам носителей на нижних уровнях СХД.

На уровне RAID предлагается проводить анализ размера сохраняемых файлов и в зависимости от его значения размещать файлы в разных томах RAID. Каждый том при этом может иметь свою ФС с определенным размером логического блока данных. Основное правило горизонтального размещения:

$$\text{Если } f \in (f_i; f_{i+1}], \text{ то} \rightarrow a_{i+1} \Leftrightarrow F \rightarrow \text{Том}_{i+1},$$

где f — размер сохраняемого файла F; f_i, f_{i+1} — левая и правая границы соответственно размера файла F, с которым может работать ФС; a_{i+1} — размер логического блока данных, которым оперирует ФС; Том_{i+1} — номер тома RAID, управляемый соответствующей ФС.

На нижних уровнях СХД предлагается разделение емкости по типам носителей, например, стример, DVD, BD для уровня автоматизированных библиотек и М-диск, стеклянный диск, ДНК для носителей длительного хранения:

$$\text{Если } f \in (f_i, f_{i+1}], \text{ то } F \rightarrow al_{i+1}\ (lt_{i+1}),$$

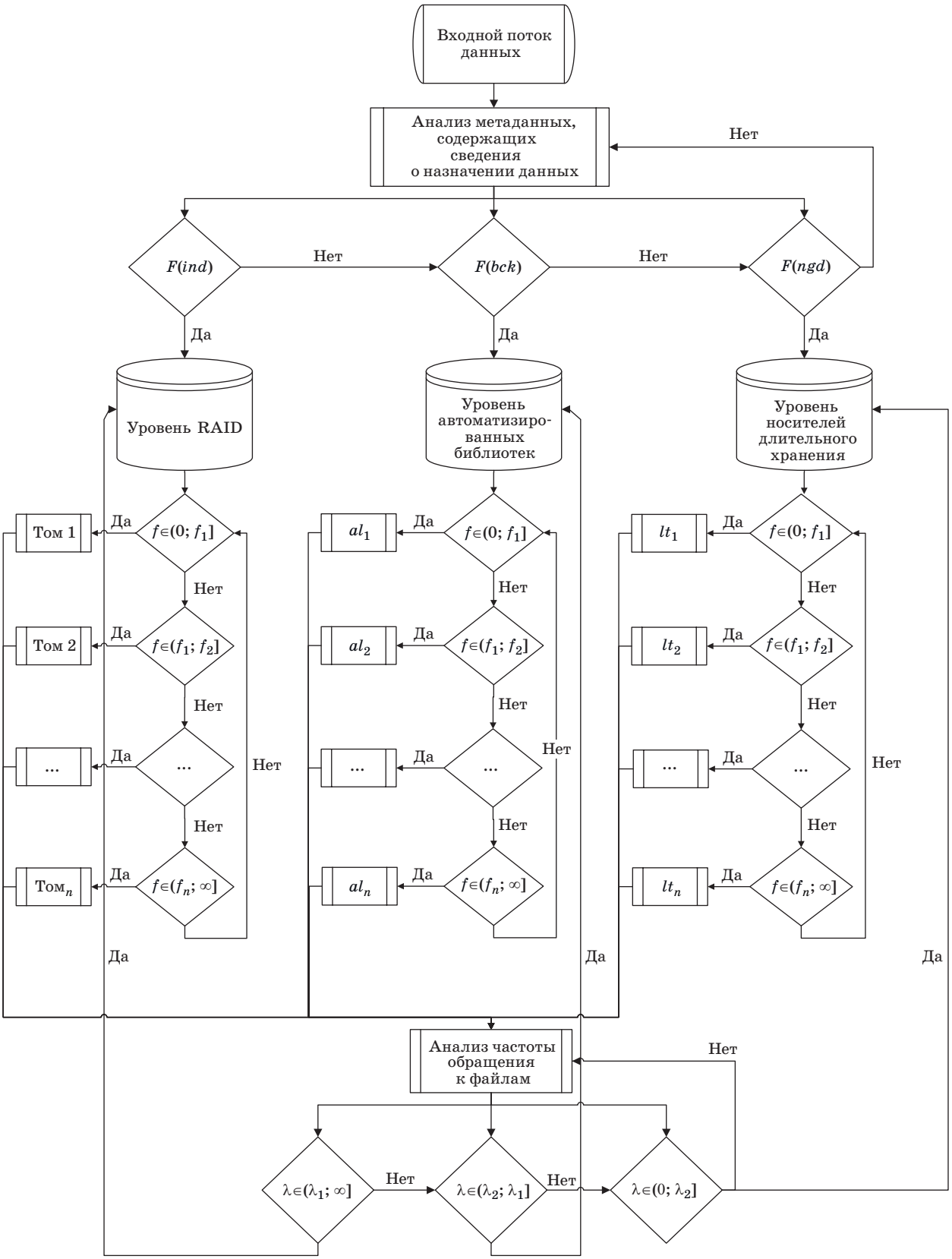
где al_{i+1} или lt_{i+1} — тип носителя на уровне автоматизированной библиотеки (al) или на уровне длительного хранения (lt).

Идея динамического размещения основана на миграции данных по уровням СХД в зависимости от частоты обращения к файлам данных [13]:

$$\text{Если } \lambda_F \in (\lambda_i, \lambda_{i+1}], \text{ то } F \rightarrow l,$$

где λ_F — частота запроса файла F; λ_i, λ_{i+1} — левая и правая границы соответственно частоты запроса файла; l — номер уровня СХД, на который мигрирует файл F.

Механизм миграции начинает функционировать только после первоначального распределе-



■ ***Рис. 2.*** Алгоритм управления емкостью СХД на основе анализа метаданных
■ ***Fig. 2.*** Storage capacity management algorithm based on metadata analysis

ния файлов при их записи в СХД. Введением механизма миграции можно преодолеть недостатки субъективного выбора типа сохраняемых файлов при реализации первого этапа (вертикального распределения).

Совокупность указанных механизмов позволяет управлять емкостью хранилища и осуществлять рациональное использование носителей. Отметим, что все указанные выше механизмы основаны на анализе метаданных сохраняемых файлов.

Блок-схема предлагаемого алгоритма управления емкостью СХД представлена на рис. 2.

Модель управления емкостью СХД

Работу механизмов вертикального и горизонтального размещения файлов в СХД в общем виде можно представить как матрицу размера $m \times n$, где m — это количество уровней хранения; n — количество физических или логических носителей. Элементами матрицы являются множества файлов, имеющие определенные значения характеристик: тип файла — *type*, размер файла — f (рис. 3). Частота обращения к файлам λ при их первоначальном размещении в СХД не учитывается, поскольку еще отсутствует статистика обращения к данным [14, 15].

В матрице всегда будет содержаться три строки, количество столбцов выбирается исходя из физической реализации каждого уровня СХД.

Последовательное выполнение алгоритмов размещения файлов требует больших энергетических затрат [16]. В связи с этим предлагается проводить распределение файлов по ячейкам матрицы рис. 3 на основе анализа метаданных файлов с использованием аппарата нейронных сетей Кохонена.

Выбор данного метода анализа обусловлен особенностями алгоритма, который реализует сеть Кохонена [17]:

	1	...	n
1	$type_1, f_1$	...	$type_n, f_n$
...	...	...	...
m	...	$type_m, f_m$	$type_{mxn}, f_{mxn}$

■ ***Рис. 3.*** Матрица управления емкостью СХД

■ ***Fig. 3.*** Storage capacity matrix

1) используется неконтролируемое обучение, при котором правило обучения нейрона основано на информации о его расположении;

2) отсутствуют эталонные значения обучающего множества: каждому объекту из исходного множества соответствует строка таблицы, и исходное множество разбивается на классы в зависимости от векторов значений признаков его объектов;

3) результатом алгоритма является топологическая карта, в которой входные данные классифицируются на группы (кластеры).

В случае предлагаемого распределения файлов по ячейкам хранилища данных очевидно, что выбранные характеристики файлов задают векторы значений признаков объектов [18]. Таким образом, каждая ячейка получающейся карты должна соответствовать элементу матрицы управления емкостью СХД.

Описание и результаты эксперимента по распределению входящего потока файлов по ячейкам матрицы хранения

Задачей эксперимента являлась демонстрация метода нейронной сети Кохонена для распределения множества файлов по ячейкам матрицы хранения после проведенного обучения сети и нормализации исходных метаданных.

Работу предложенного метода проследим последовательно в соответствии с применяемыми механизмами:

1) вертикальное распределение файлов по уровням матрицы хранения в зависимости от предполагаемого времени хранения файлов;

2) горизонтальное распределение файлов по ячейкам уровня матрицы хранения в зависимости от типа и размера файлов;

3) динамическое распределение по уровням матрицы хранения в зависимости от частоты обращения к файлам.

В эксперименте участвовало 5000 файлов с различными характеристиками: тип файла, предполагаемое время его хранения, размер файла, частота обращения к данным. Поток сгенерирован на основе анализа 43 000 файлов, взятых с экспериментального файл-сервера [19]. Размер файлов не превышал 1 ГБ, частота обращения к данным является случайной величиной в интервале [0÷300]. Множество экспериментальных файлов имело следующий состав:

файлов типа *ind* — 4150 (83 % от общего количества файлов эксперимента);

файлов типа *bck* — 750 (15 % от общего количества файлов эксперимента);

файлов типа *ngd* — 100 (2 % от общего количества файлов эксперимента).

Кроме того, экспериментальное множество файлов было разбито на три части: 70 % всех файлов множества использовалось для обучения сети; 15 % — тестовое множество; 15 % — множество валидации.

Размещение файлов данных выполнялось в соответствии со структурой матрицы хранения размера 3 × 3.

Нормализация типов файлов принята исходя из эксперимента по подбору значений, которые давали бы непересекающиеся классы. В результате получено следующее соответствие: тип файлов: *ind* соответствует значению 1000, *bck* — 2000, *ngd* — 3000.

Нормализация размера файлов выполнена в десятичных порядках и также принята исходя из эксперимента: при размере файла от 0 до 999 Б присваиваем значение атрибута 1000; от 1000 до 999 999 Б — 5000; от 1 000 000 до 999 999 999 Б — 10 000.

Частота обращения рассматривалась в абсолютных единицах.

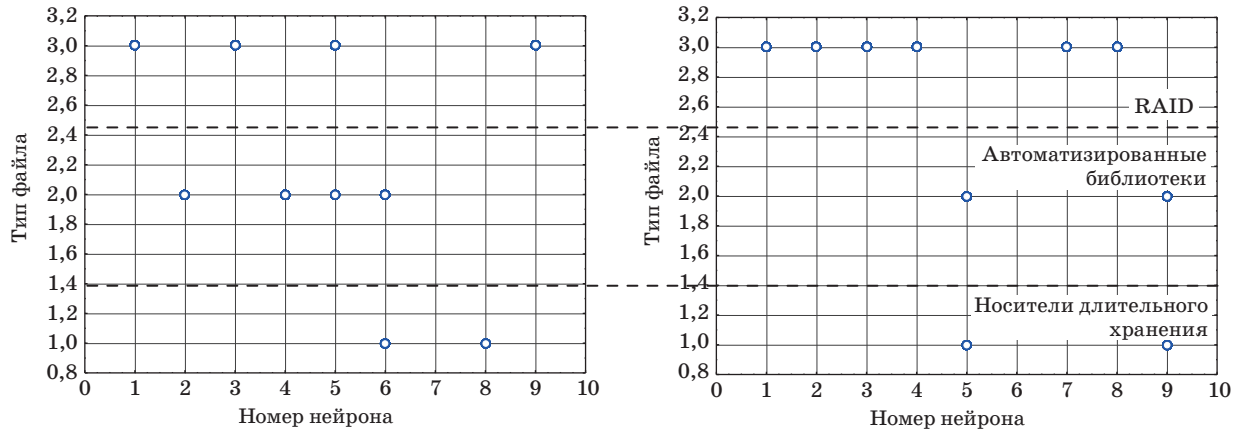
Количество тренировочных циклов — 1000.

Евклидово расстояние — 3.

Две карты Кохонена при разных выборках экспериментальных файлов демонстрируют распределение файлов по уровням хранилища данных в зависимости от предполагаемого времени хранения файлов (рис. 4).

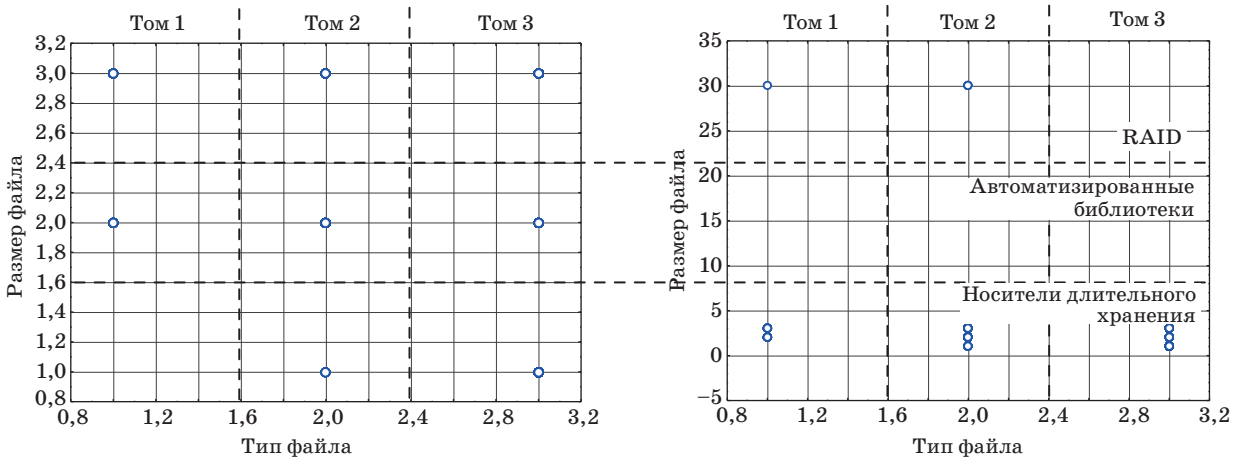
Две карты Кохонена при разных выборках экспериментальных файлов демонстрируют распределение файлов по ячейкам матрицы хранения в зависимости от типа и размера файлов (рис. 5).

Две карты Кохонена при разных выборках экспериментальных файлов демонстрируют распределение файлов по ячейкам матрицы хранения в зависимости от типа, размера файлов и частоты обращения к файлам (рис. 6).



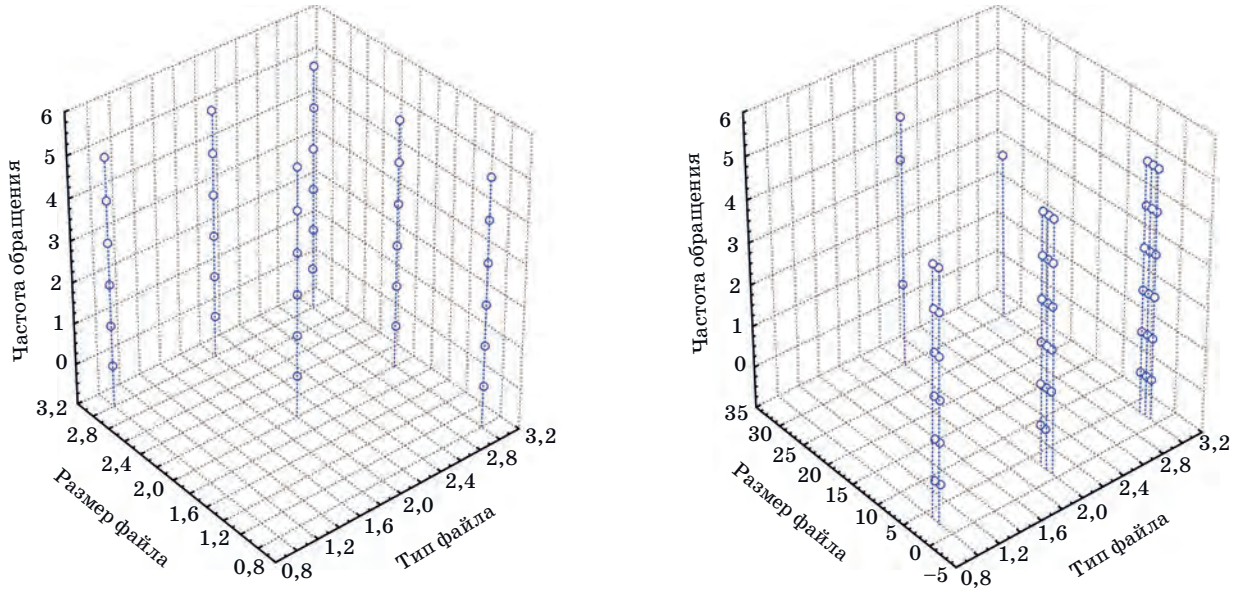
■ ***Рис. 4.*** Распределение файлов по уровням хранилища данных

■ ***Fig. 4.*** Distributing files by data storage level



■ ***Рис. 5.*** Распределение файлов в зависимости от типа и размера

■ ***Fig. 5.*** File distribution according to type and size



■ ***Рис. 6***. Распределение файлов в зависимости от типа, размера и частоты обращения
■ ***Fig. 6***. File distribution, depending on the type, size and frequency of treatment

Как показывают результаты эксперимента, аппарат нейронной сети Кохонена может быть инструментом решения задачи размещения объектов в соответствии с требуемыми параметрами. Основной сложностью при этом является выбор параметров классификации и нормализация параметров.

Заключение

В статье предложен алгоритм многоуровневого хранения данных, позволяющий распределять файлы в СХД в соответствии с последовательным применением механизмов:

— вертикального размещения по уровням системы хранения в зависимости от времени хранения файла;

— горизонтального размещения по логическим томам уровня хранения в зависимости от размера файла и длины логического блока данных;

— динамического размещения — миграции файлов по уровням СХД в зависимости от частоты обращения к ним.

Вертикальное и горизонтальное размещение основано на анализе организационных метаданных, приписанных файлам.

С учетом механизмов вертикального и горизонтального размещения файлов в системе хранения предложена модель структуры хранения в виде матрицы, размер которой соответствует количеству вертикальных и горизонтальных уровней СХД.

Предложено проводить распределение файлов по ячейкам матрицы с использованием аппарата нейронных сетей Кохонена, что позволяет отказаться от последовательного выполнения механизмов размещения.

Результаты эксперимента показали способность аппарата нейронной сети Кохонена как инструмента решения задачи размещения файлов в соответствии с требуемыми параметрами.

Литература

1. **Проскуряков Н. Е., Ануфриева А. Ю.** Анализ и перспективы современных систем хранения цифровых данных. *Изв. Тульского государственного университета. Технические науки*, 2013, вып. 3, с. 368–377.
2. **Бурмистров В. Д., Заковряшин Е. М.** Создание хранилища данных для распределенной системы. *Молодой ученый*, 2016, № 12, с. 143–147.
3. **Farley M.** *Building Storage Networks*. Osborne, McGraw-Hall, 2001. 576 p.
4. *Information Storage and Management*. 2nd Edition. New Jersey, John Wiley & Sons, 2016. 544 p.
5. **Татарникова Т. М., Пойманова Е. Д.** Технологии долговременного хранения данных. *Материалы Междунар. науч.-практ. конф. «Наука и образование в XXI веке»*, Тамбов, Бизнес-Наука-Общество, 2013, ч. 31, с. 136–137.
6. **Landauer R.** Information is physical. *Physics Today*, 1991, vol. 44, no. 5, pp. 23–29.
7. **Kish L. B., Granqvist C. G.** Does information have mass? *Proc. of the IEEE*, 2013, vol. 101, no. 9, pp. 1895–1899.

8. **Buyya R., Broberg J., Goscinski A.** *Cloud Computing. Principles and Paradigms*. New Jersey, John Wiley & Sons, 2011. 637 p.
9. **Mesnier M., Ganger G., Riedel E.** Object-based storage. *IEEE Communications Magazine*, 2003, vol. 41, no. 8, pp. 84–90.
10. *Recommendation Y.3501: Cloud computing framework and high-level requirements*. Geneva, ITU-T, 2013. https://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11917 (дата обращения: 29.01.2019).
11. *Recommendation Y.3510: Cloud computing infrastructure requirements*. Geneva, ITU-T, 2013. https://www.itu.int/rec/T-REC-Y.3510-201305-S (дата обращения: 29.01.2019).
12. *Recommendation Y.3520: Cloud computing framework for end-to-end resource management*. Geneva, ITU-T, 2015. https://www.itu.int/rec/T-REC-Y.3520-201509-I (дата обращения: 29.01.2019).
13. **Hastic T., Tibshirani R., Friedman J.** *The Elements of Statistical Learning*. Springer, 2003. 552 p.
14. **Todman C.** *Designing a Data Warehouse: Supporting Customer Relationship Management*. Prentice Hall, 2001. 414 p.
15. **Stacey M., Salvatore J., Jorgensen A.** *Visual Intelligence: Microsoft Tools and Techniques for Visualizing Data*. New Jersey, John Wiley & Sons, 2013. 432 p.
16. **Kish L. B.** Moore's law and the energy requirement of computing versus performance. *IEEE Proceedings: Circuits, Devices and Systems*, 2004, vol. 151, no. 2, pp. 190–194.
17. **Kohonen T.** *Self-Organizing Maps*. New York, 2001. 501 p.
18. **Kutuzov O. I., Tatarnikova T. M.** Model of a self-similar traffic generator and evaluation of buffer storage for classical and fractal queuing system. *Moscow Workshop on Electronic and Networking Technologies, MWENT*, 2018, pp. 1–3.
19. **Morville P., Callender J.** *Search Patterns: Design for Discovery*. O'Reilly, 2010. 192 p.

UDC 004.07
doi:10.31799/1684-8853-2019-2-68-75

Organization of multi-level data storage

Sovetov B. Ya.[a], Dr. Sc., Tech., Professor, orcid.org/0000-0003-3116-8810
Tatarnikova T. M.[b], Dr. Sc., Tech., Associate Professor, orcid.org/0000-0002-6419-0072, tm-tatarn@yandex.ru
Poymanova E. D.[b], Senior Lecturer, orcid.org/0000-0002-7903-2480
[a]Saint-Petersburg Electrotechnical University «LETI», 5, Prof. Popov St., 197376, Saint-Petersburg, Russian Federation
[b]Saint-Petersburg State University of Aerospace Instrumentation, 67, B. Morskaia St., 190000, Saint-Petersburg, Russian Federation

Introduction: In the face of a constantly increasing data volume, almost all organizations need to form a well-developed storage infrastructure. The infrastructure is implemented using multi-level storage technologies: designing systems of storage devices heterogeneous in physical nature and access architecture. On the other hand, the files are also subject to the requirements about the guaranteed storage time, even at the legislative level. The lack of efficient storage organization algorithms with different storage requirements actualizes new storage resource management models. **Purpose:** Developing new models and algorithms for managing data storage resources, taking into account the layered principle of data storage. **Results:** A multi-level data storage algorithm is proposed which allows you to distribute files in a data storage system in accordance with the sequential use of vertical, horizontal and dynamic file allocation. The algorithms for data placement and migration across the storage hierarchies differ in the use of metadata analysis, providing a way for proactive management of the storage resources based on guaranteed retention periods. A data storage model is proposed in the form of a matrix whose size corresponds to the number of vertical and horizontal levels in the system. It is suggested to distribute files to the cells of the matrix using the Kohonen neural network tools. The effect of using the Kohonen network is the ability to analyze metadata and the frequency of accessing files in one step, avoiding sequential implementation of the allocation procedures. The results of the experiment have demonstrated the effectiveness of the Kohonen network tools in solving the problem of placing files in a multi-level data storage system. **Practical relevance:** The proposed model and algorithms can be used in the design of data storage systems, taking into account the multi-level principle of data storage.

Keywords — data storage system, multi-level data storage, file storage time, metadata, data migration, Kohonen neural network.

For citation: Sovetov B. Ya., Tatarnikova T. M., Poymanova E. D. Organization of multi-level data storage. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2019, no. 2, pp. 68–75 (In Russian). doi:10.31799/1684-8853-2019-2-68-75

References

1. Proskuryakov N. E., Anufrieva A. Yu. Analysis and prospects of modern digital data storage systems. *Izvestiya Tul'skogo gosudarstvennogo universiteta. Tekhnicheskie nauki* [Proc. of the Tula State University. Technical Sciences], 2013, no. 3, pp. 368–377 (In Russian).
2. Burmistrov V. D., Zakovryashin E. M. Creating a data warehouse for a distributed system. *Molodoj uchenyj* [Young Scientist], 2016, no. 12, pp. 143–147 (In Russian).
3. Farley M. *Building Storage Networks*. Osborne, McGraw-Hall, 2001. 576 p.
4. *Information Storage and Management*. 2nd Edition. New Jersey, John Wiley & Sons, 2016. 544 p.
5. Tatarnikova T. M., Poymanova E. D. Long term storage technologies. *Materialy Mezhdunarodnoi nauchno-prakticheskoi konferentsii "Nauka i obrazovanie v XXI veke"* [Proc. of the Intern. Scientific and Practical Conf. "Science

and Education in XXI Century"], Tambov, Biznes-Nauka-Obshchestvo Publ., 2013, vol. 31, pp. 136–137 (In Russian).
6. Landauer R. Information is physical. *Physics Today*, 1991, vol. 44, no. 5, pp. 23–29.
7. Kish L. B., Granqvist C. G. Does information have mass? *Proc. of the IEEE*, 2013, vol. 101, no. 9, pp. 1895–1899.
8. Buyya R., Broberg J., Goscinski A. *Cloud Computing. Principles and Paradigms*. New Jersey, John Wiley & Sons, 2011. 637 p.
9. Mesnier M., Ganger G., Riedel E. Object-based storage. *IEEE Communications Magazine*, 2003, vol. 41, no. 8, pp. 84–90.
10. *Recommendation Y.3501: Cloud computing framework and high-level requirements*. Geneva, ITU-T, 2013. Available at: https://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11917 (accessed 29 January 2019).
11. *Recommendation Y.3510: Cloud computing infrastructure requirements*. Geneva, ITU-T, 2013. Available at: https://www.itu.int/rec/T-REC-Y.3510-201305-S (accessed 29 January 2019).
12. *Recommendation Y.3520: Cloud computing framework for end-to-end resource management*. Geneva, ITU-T, 2015. Available at: https://www.itu.int/rec/T-REC-Y.3520-201509-I (accessed 29 January 2019).
13. Hastic T., Tibshirani R., Friedman J. *The Elements of Statistical Learning*. Springer, 2003. 552 p.
14. Todman C. *Designing a Data Warehouse: Supporting Customer Relationship Management*. Prentice Hall, 2001. 414 p.
15. Stacey M., Salvatore J., Jorgensen A. *Visual Intelligence: Microsoft Tools and Techniques for Visualizing Data*. New Jersey, John Wiley & Sons, 2013. 432 p.
16. Kish L. B. Moore's law and the energy requirement of computing versus performance. *IEEE Proceedings: Circuits, Devices and Systems*, 2004, vol. 151, no. 2, pp. 190–194.
17. Kohonen T. *Self-Organizing Maps*. New York, 2001. 501 p.
18. Kutuzov O. I., Tatarnikova T. M. Model of a self-similar traffic generator and evaluation of buffer storage for classical and fractal queuing system. *Moscow Workshop on Electronic and Networking Technologies, MWENT*, 2018, pp. 1–3.
19. Morville P., Callender J. *Search Patterns: Design for Discovery*. O'Reilly, 2010. 192 p.