НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
март–апрель 2019 Том 19 № 2 ISSN 2226-1494 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
March–April 2019 Vol. 19 No 2 ISSN 2226-1494 http://ntv.ifmo.ru/en

УДК 004.7

МОДЕЛЬ МНОГОУРОВНЕВОЙ СИСТЕМЫ ХРАНЕНИЯ ДАННЫХ

Т.М. Татарникова[a,b], Е.Д. Пойманова[a]

[a] Санкт-Петербургский государственный университет аэрокосмического приборостроения, Санкт-Петербург, 190000, Российская Федерация
[b] Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина), Санкт-Петербург, 197376, Российская Федерация
Адрес для переписки: Tm-tatarn@yandex.ru

Информация о статье
Поступила в редакцию 15.01.19, принята к печати 20.02.19
doi: 10.17586/2226-1494-2019-19-2-271-279
Язык статьи – русский

Ссылка для цитирования: Татарникова Т.М., Пойманова Е.Д. Модель многоуровневой организации системы хранения данных // Научно-технический вестник информационных технологий, механики и оптики. 2019. Т. 19. № 2. С. 271–279. doi: 10.17586/2226-1494-2019-19-2-271-279

Аннотация
Исследована организация системы хранения данных, основанная на последовательном применении алгоритмов вертикального размещения файлов по уровням системы хранения данных, горизонтального – по секциям определенного уровня и динамического – при перемещении данных по уровням. Выбраны и нормализованы метаданные, задающие характеристики сохраняемых файлов. Модель позволяет организовать хранение файлов в соответствии с их характеристиками, в том числе требованиями ко времени гарантированного хранения. Представление системы хранения в виде матрицы позволяет использовать инструмент нейронной сети Кохонена для размещения файлов по уровням и по секциям определенного уровня. Использование нейронной сети Кохонена позволяет перейти от последовательного выполнения алгоритмов к размещению за один шаг. Предложена модель многоуровневого хранения данных. Разработаны алгоритмы размещения файлов в системе хранения данных с многоуровневой структурой. Приведены тестовые примеры, демонстрирующие возможности аппарата нейронной сети Кохонена при размещении файлов в соответствии с требуемыми параметрами. Совместное применение алгоритмов размещения файлов позволяет организовать многоуровневое хранение данных в соответствии с характеристиками файлов и обеспечением требований к времени гарантированного хранения.

Ключевые слова
многоуровневое хранение, система хранения данных, эффективное размещение данных, метаданные файла, миграция данных, кластеризация файлов, нейронная сеть Кохонена

MODEL OF MULTI-LEVEL DATA STORAGE SYSTEM

T.M. Tatarnikova[a,b], E.D. Poymanova[a]

[a]Saint Petersburg State University of Aerospace Instrumentation, Saint Petersburg, 190000, Russian Federation
[b]Saint Petersburg Electrotechnical University "LETI", Saint Petersburg, 197376, Russian Federation
Corresponding author: Tm-tatarn@yandex.ru

Article info
Received 15.01.19, accepted 20.02.19
doi: 10.17586/2226-1494-2019-19-2-271-279
Article in Russian

For citation: Tatarnikova T.M., Poymanova E.D. Model of multi-level data storage system. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2019, vol. 19, no. 2, pp. 271–279 (in Russian). doi: 10.17586/2226-1494-2019-19-2-271-279

Abstract
A model of multi-level organization for data storage system is studied, based on the sequential application of algorithms for vertical file distribution by the levels of data storage system, horizontal placement in sections of a certain level, and dynamic placement as a result of data migration. Selection and normalization of metadata specifying the characteristics of the stored files were performed. The model of multi-level data storage provides the storage of files in accordance with their characteristics and meets the requirements for guaranteed storage time. Representation of the storage system in the form of a matrix enables the usage of Kohonen neural network tool to arrange files by levels and sections of a specific storage system level. The application of Kohonen neural network provides the transfer from sequential execution of algorithms to placement in one step. We have proposed the model of multi-level data storage. Algorithms have been developed for file placement in a multi-level data storage system. Test examples are given which demonstrate the ability of Kohonen neural network apparatus as a tool for solving the file allocation problem in accordance with the required parameters. The combined use of file

allocation algorithms gives the possibility to organize multi-level data storage in accordance with the files characteristics and the assurance of time requirements for guaranteed storage.

Keywords
multi-level storage, data storage system, efficient data placement, file metadata, data migration, file clusterization, Kohonen neural network

Введение

Практически все организации в настоящее время нуждаются в формировании развитой инфраструктуры хранения данных. Существующие технологии дубликации, зеркалирования, виртуализации и т.д. обеспечивают гарантированное хранение данных [1, 2]. Однако в последнее время большое внимание уделяется организации многоуровневого хранения данных, при котором для каждого файла в зависимости от его жизненного цикла выбирается технология хранения. В центрах обработки данных для эффективного размещения информационных ресурсов руководствуются этой практикой, однако современные системы хранения по-прежнему остаются одномерными, поскольку миграция данных между уровнями определяется одной метрикой – временем, прошедшим с момента последнего обращения к информации [3, 4].

Под системой хранения данных (СХД) будем понимать архитектурное решение для подключения внешних устройств хранения. СХД являются инженерным решением для реализации инфраструктуры хранения данных как домашнего компьютера, так и центра обработки [5–7].

Реализация многоуровневого хранения данных осложняется следующим:

– разнородность СХД – устройства хранения в СХД могут различаться физической природой и архитектурой доступа: магнитные, оптические, твердотельные; прямого или сетевого доступа [8];

– наличие разных требований к хранению данных – транзакционным системам требуются высоконадежные и производительные СХД, аналитическим – высокая производительность и низкая стоимость в расчете на единицу хранения, для работы с файлами нужна низкая стоимость хранения [9];

– отсутствие алгоритмов эффективной организации хранения данных с разными требованиями [10];

– необходимость параллельного выполнения основной функции СХД и функций, обеспечивающих восстановление данных без остановки выполняемых бизнес-процессов [11];

– проблема модернизации и утилизации СХД, делающая дорогостоящей стоимость хранения единицы данных. Зачастую СХД, приобретенная три-пять лет назад, не справляется с растущими объемами и не отвечает требованиям по скорости доступа к данным. В этом случае приобретается новая СХД, на которую переносятся данные с прежней. Таким образом, владелец СХД, повторно платит за объемы хранения. При этом прежние СХД, как правило еще не настолько устаревшие, по своему прямому назначению не эксплуатируются, т.е. происходит ранняя утилизация емкости СХД.

Цель работы состоит в создании модели многоуровневого хранения данных и алгоритмов управления ресурсами многоуровневой СХД, которые позволят частично преодолеть перечисленные проблемы.

Алгоритмы размещения данных в многоуровневой системе хранения данных

Анализ инженерных решений в области хранения данных позволяет выделить три уровня в архитектуре СХД, каждый из которых предполагает свои технологии хранения: RAID (Redundant Array of Independent Disks – избыточный массив независимых дисков), автоматизированные библиотеки и носители длительного хранения[1] [12].

Предлагается комплексно использовать три алгоритма, реализующие многоуровневое хранение данные:

– алгоритм вертикального размещения для выбора уровня СХД. Основная метрика, на основе которой происходит выбор уровня СХД – требуемое время хранения;
– алгоритм горизонтального размещения для выбора секции (логического тома) уровня хранения СХД, например, для RAID-массивов. Основная метрика – размер файла и длина логического блока данных;
– алгоритм миграции для перемещения файла данных по уровням СХД. Основная метрика – частота обращения к файлу.

Все алгоритмы основаны на анализе метаданных файлов [13, 14].

[1] 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).
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).
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).

С учетом многоуровневой организации хранения данных представим структуру СХД в виде матрицы размером $m \times n$, где m – это число уровней хранения; n – число физических или логических носителей. Элементами матрицы являются множества файлов, *type* – тип файла, f – размер файла (рис. 1). Частота обращения к файлам λ при их первоначальном размещении в СХД не учитывается.

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

Рис. 1. Представление системы хранения данных в виде матрицы

Матрица всегда содержит три строки, а число столбцов выбирается исходя из физической реализации каждого уровня СХД [15].

При поступлении файла на хранение в СХД сначала выбирается уровень хранения. Алгоритм вертикального размещения файла основан на анализе организационных метаданных, содержащих сведения о типе файла. 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 данных, архивные данные, которые будут храниться на уровне автоматизированной библиотеки;

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

Затем происходит горизонтальное распределение файлов. Идея горизонтального размещения основана на выборе файловой системы для уровня RAID и типа носителей на нижних уровнях СХД.

На уровне RAID предлагается анализировать размер сохраняемого файла, и в зависимости от этого размещать файлы в разных томах RAID. Каждый том при этом может иметь свою файловую систему с определенным размером логического блока данных. Горизонтальное размещение происходит в соответствии со следующим правилом:

если $f \in (f_i; f_{i+1}]$, то $\rightarrow a_{i+1} \Leftrightarrow F \rightarrow V_{i+1}$,

где f_i, f_{i+1} – левая и правая границы файла F, с которым может работать файловая система; a_{i+1} – размер логического блока данных, которым оперирует файловая система; V_{i+1} – номер тома RAID, управляемого соответствующей файловой системой.

На нижних уровнях СХД предлагается размещать файлы в соответствии с типом носителя:

– стример, DVD, BD – для уровня автоматизированных библиотек (*al*);
– М-диск, стеклянный диск и ДНК – для длительного хранения (*lt*).

Правило горизонтального размещения:

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

где al_{i+1} или lt_{i+1} – тип носителя на уровне *al* или на уровне *lt*.

Правило динамического размещения основано на миграции данных по уровням СХД в зависимости от частоты обращения к файлам [16]:

если $\lambda_F \in (\lambda_i, \lambda_{i+1}]$, 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 перемещается файл F.

Механизм миграции начинает работать после записи файлов в СХД. Введение механизма миграции позволяет преодолеть недостатки субъективного выбора типа сохраняемых файлов при реализации вертикального распределения.

Алгоритм размещения файлов в многоуровневой СХД приведен на рис. 2.

Последовательное выполнение алгоритмов размещения файлов требует больших энергетических затрат [14]. В связи с этим предлагается проводить распределение файлов по ячейкам матрицы (см. рис. 1) с использованием аппарата нейронных сетей Кохонена.

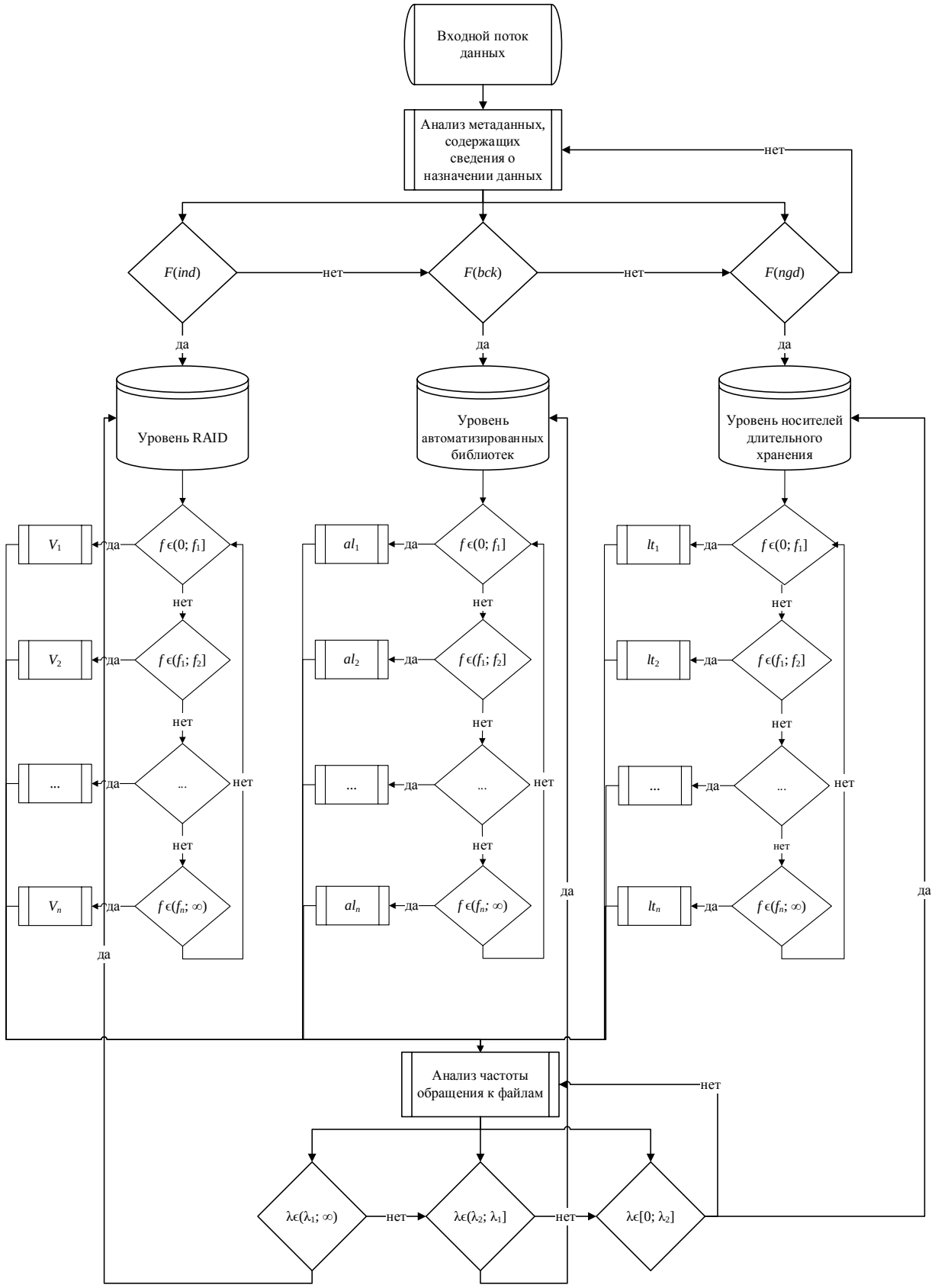

Рис. 2. Алгоритм размещения файлов в многоуровневой системе хранения данных

Сеть Кохонена как инструмент кластеризации файлов данных

Самоорганизующаяся нейронная сеть (карта) Кохонена, в отличие от многих других видов нейронных сетей, обучается без учителя [17].

Основное назначение сети Кохонена – кластерный анализ (кластеризация). Кластеризация – разбиение заданной выборки объектов (ситуаций) на непересекающиеся подмножества так, чтобы каждый кластер состоял из схожих объектов, а объекты разных кластеров существенно различались [14].

Сеть Кохонена включает два слоя: входящий и исходящий. Каждый нейрон x_i, $i=\overline{1,n}$ входящего слоя соединяется с каждым нейроном y_j, $j=\overline{1,s}$, исходящего слоя и все связи, имеют определенный вес, который корректируется в процессе обучения. Исходящий слой также называется слоем топологической карты. Нейроны топологической карты рассредоточиваются в некотором поле, как правило, двумерном. Поэтому сеть Кохонена чаще изображают в виде топологической карты, т.е. только нейроны выходного слоя (рис. 3) [18].

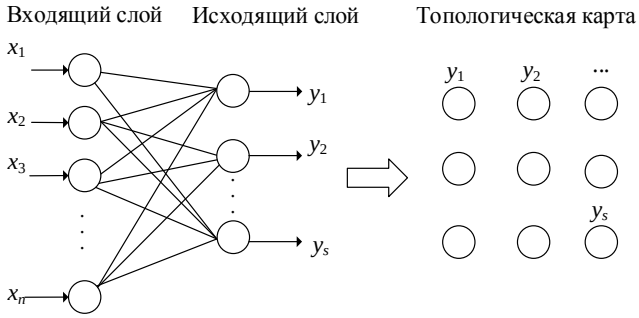

Рис. 3. Архитектура сети Кохонена

Карты Кохонена имеют набор входных элементов, число которых совпадает с размерностью подаваемых на вход векторов, и набор выходных элементов, каждый из которых соответствует одному кластеру (группе).

Работа сети Кохонена заключается в следующем. При подаче некоторого вектора **X** на вход сеть должна определить, к какому из кластеров этот вектор ближе всего. В роли кластера выступает выходной нейрон, а в качестве критерия близости может быть выбрана метрика квадрата евклидова расстояния:

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

где d_{ij} – квадрат расстояния между точкой **X** и кластером **Y**; координаты точки **X** – x_{i1}, x_{i2}, …, x_{is}, центра кластера **Y** – y_{i1}, y_{i2}, …, y_{is}.

Вектор **X** представляет собой точку в *s*-мерном пространстве, где *s* – число координат вектора, которые подаются на вход нейронов сети. Вычисление расстояния между этой точкой и центрами разных кластеров позволяет определить кластер, расстояние до которого минимально.

Центр кластера – точка, координатами которой в *s*-мерном пространстве являются веса всех связей, которые приходят к данному выходному нейрону от входных.

Сеть Кохонена решает задачу кластеризации следующим образом: после подачи вектора на входящий слой будет получен один кластер-победитель, который определяет принадлежность входного вектора к этому кластеру.

Обучение сети Кохонена происходит методом последовательных уступок. Основной рекуррентный алгоритм обучения сети Кохонена перебирает одну за другой множество эпох, при этом на каждом этапе обрабатывается любой обучающий пример в такой последовательности [17]:

- взять нейрон-победитель, т.е. тот, который ближе расположен к введенному вектору (примеру);
- выполнить коррекцию нейрона-победителя так, чтобы он стал максимально похожим на введенный пример, найдя квадрат евклидова расстояния от центра нейрона-победителя до этого примера.

При вычислении центра нейрона-победителя используется коэффициент скорости обучения, который постепенно убывает так, что на каждом новом этапе скорректированный результат приближается к заданному. В итоге расположение центра наилучшим образом отражает примеры, для которых этот нейрон является победителем.

После такой рекуррентной процедуры обучения нейронная сеть будет организована так, что нейроны, расположенные близко друг к другу в пространстве входящего слоя, находятся близко друг к другу и на топологической карте.

Построение карты Кохонена основано на понятии окрестности кластера.

Окрестность (группа нейронов, которые окружают нейрон-победитель) характеризуется радиусом R (рис. 4). Радиус окрестности уменьшается со временем обучения таким образом, что сначала в окрестности находится значительное число нейронов (возможно, почти все, что расположены на топологической карте); на конечных этапах окрестность стремится к нулевому значению, т.е. включает только сам нейрон-победитель.

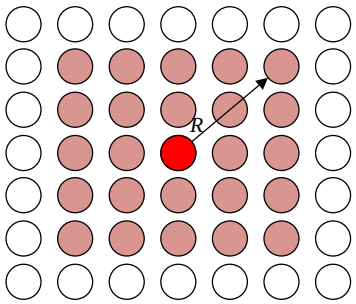

Рис. 4. Окрестность кластера

В результате такого обучения первоначальные окрестности «перетягиваются» в направлении примеров обучения. Формируется грубая структура топологической карты, в которой похожие данные образуют наборы близко расположенных нейронов.

Таким образом, суть обучения сети Кохонена состоит в настройке весов всех связей для максимального совпадения с входными примерами.

Можно представить алгоритм процесса обучения сети Кохонена в виде последовательности шагов.

Шаг 1. Для входного вектора **X** определить координаты нейронов выходного слоя, вычислить d_{ij} от **X** до каждого нейрона сети.

Шаг 2. Найти минимальное d_{ij} из полученных значений, определить нейрон-победитель.

Шаг 3. Для нейрона-победителя, а также для тех нейронов, которые попали в заданный R, выполнить корректировку весов связей:

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

где $w_{ij}(t+1)$ – вес на шаге t+1; $w_{ij}(t)$ – вес на шаге t; Δ – скорость обучения; x_i – координата входного вектора.

Шаг 4. Обновить значения нормы обучения и R.

Шаг 5. Продолжить обучение до выполнения условия остановки обучения. Остановка обучения происходит, если величина изменения весов становится незначительной.

После того как сеть обучена решать задачу кластеризации, используем ее как средство визуализации в виде карты Кохонена [19].

Выбор данного метода для размещения файлов обусловлен особенностями алгоритма, который реализует сеть Кохонена [17]:

1) используется неконтролируемое обучение, при котором правило обучения нейрона основано на информации об 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 выбранные характеристики файлов задают векторы значений признаков объектов [20]. Таким образом, каждый кластер карты должна соответствовать элементу матрицы, представляющей СХД.

Описание эксперимента

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

Работу метода проследим на последовательном применении алгоритмов.

В эксперименте использовано 5000 файлов с различными характеристиками: тип файла, его предполагаемое время хранения, размер файла, частота запроса файла. Поток сгенерирован на основе анализа 43 000 файлов, взятых с экспериментального файл-сервера [21]. Размер файлов не превышает 1 ГБ, частота обращения к данным в интервале [0–300 1/ч] является случайной величиной. Множество экспериментальных файлов имеет следующий состав:

– файлов типа *ind* – 4150, что составляет 83 % общего числа файлов эксперимента;
– файлов типа *bck* – 750, 15 %;
– файлов типа *ngd* – 100, 2 %.

Экспериментальное множество файлов было разбито на три части:

1) 70 % всех файлов множества используется для обучения сети;
2) 15 % – тестовое множество;
3) 15 % – множество валидации.

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

Нормализация типов файлов была принята для получения непересекающихся классов: типу файлов *ind* соответствует значение 1, *bck* – 2, *ngd* – 3.

Нормализация размера файлов выполнена в десятичных порядках и также принята исходя из эксперимента: при размере файла от 0 до 999 Б присваиваем значение атрибута 1; от 1000 до 999 999 – 2; от 1 000 000 до 999 999 999 – 3.

Частота обращения к данным рассматривалась в абсолютных единицах – число запросов в час.

Число эпох обучения – 1000.

Евклидово расстояние – 10^{-2}.

На рис. 5 приведены две карты Кохонена при разных выборках экспериментальных файлов, демонстрирующие распределение файлов по уровням хранилища данных в зависимости от предполагаемого времени хранения.

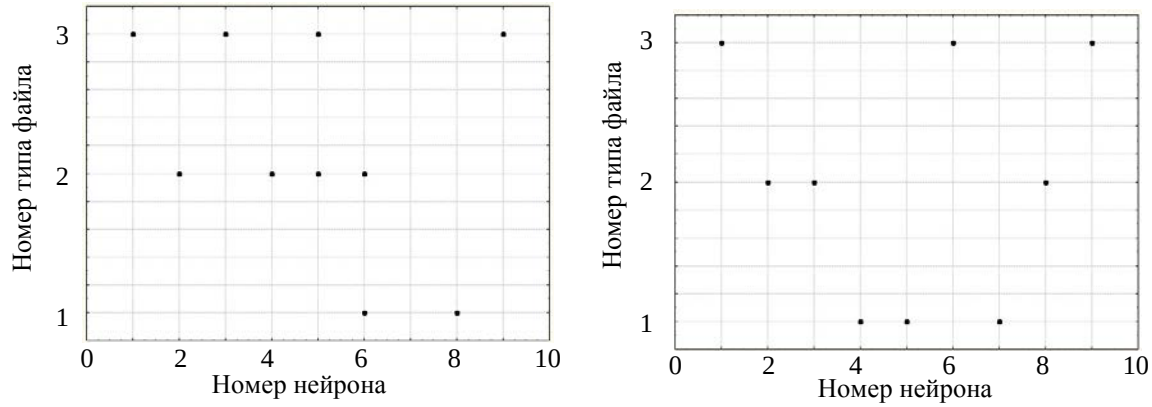

Рис. 5. Распределение файлов по уровням хранилища данных

На рис. 6 приведены две карты Кохонена при разных выборках экспериментальных файлов, демонстрирующие распределение файлов по ячейкам матрицы хранения в зависимости от типа и размера файлов.

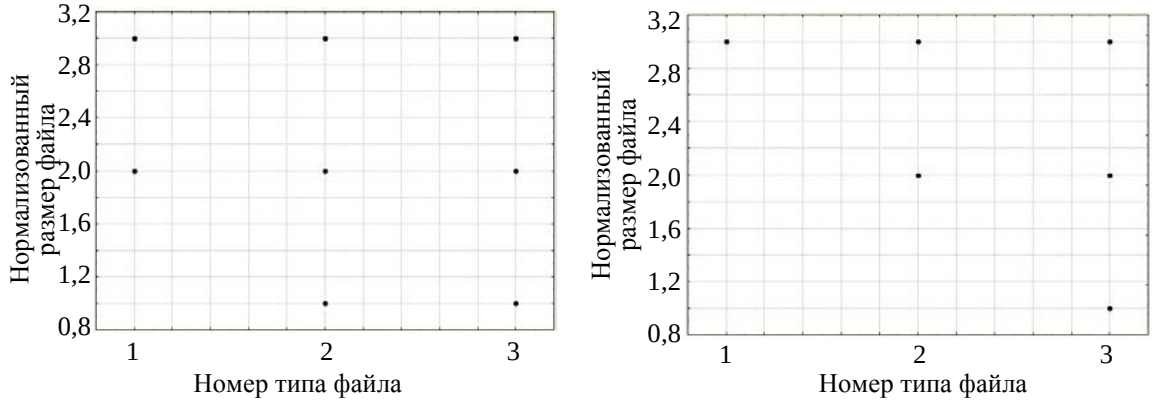

Рис. 6. Распределение файлов в зависимости от типа и размера

На рис. 7 приведены две карты Кохонена при разных выборках экспериментальных файлов, демонстрирующие распределение файлов по ячейкам матрицы хранения в зависимости от типа, размера файлов и частоты обращения к ним.

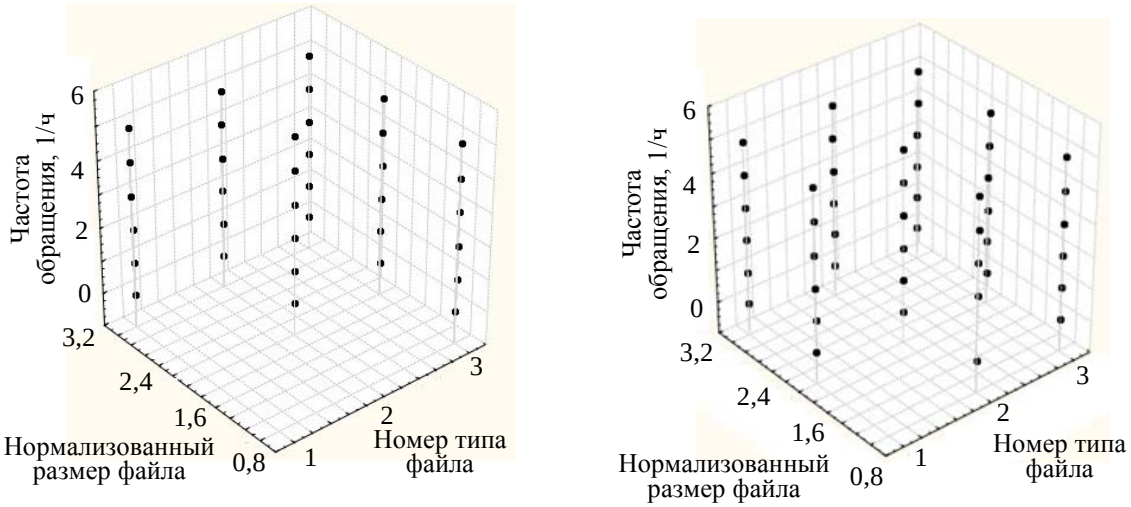

Рис. 7. Распределение файлов в зависимости от типа, размера и частоты обращения

Как показывают результаты эксперимента, аппарат нейронной сети Кохонена может быть инструментом решения задачи размещения файлов с разными характеристиками и требованиями к времени хранения. Основную сложность представляют выбор параметров классификации и нормализация параметров.

Заключение

Предложена модель многоуровневого хранения данных. Файлы в системе хранения данных распределяются в соответствии с последовательным применением алгоритмов вертикального, горизонтального размещения и миграции данных. Вертикальное и горизонтальное размещение основано на анализе организационных метаданных, задающих тип и размер файла. Алгоритм миграции данных основан на анализе частоты обращения к файлу.

Формализация вертикального и горизонтального размещения в виде матрицы позволила использовать аппарат нейронной сети Кохонена, основное предназначение которой – кластеризация. В задаче размещения файлов кластерами являются ячейки матрицы, которые ассоциированы с секциями определенного уровня системы хранения данных. Перед кластеризацией выполнена нормализация метаданных, задающих характеристики сохраняемых файлов.

Использование нейронной сети Кохонена позволило отказаться от последовательного выполнения алгоритмов размещения, т.е. перейти к размещению за один шаг.

Результаты эксперимента показали способность аппарата нейронной сети Кохонена к размещению файлов в соответствии с требуемыми параметрами.

Литература

1. Богатырев В.А., Богатырев С.В. Объединение резервированных серверов в кластеры высоконадежной компьютерной системы // Информационные технологии. 2009. № 6. С. 41–47.
2. Проскуряков Н.Е., Ануфриева А.Ю. Анализ и перспективы современных систем хранения цифровых данных // Известия ТулГУ: Технические науки. 2013. № 3. С. 368–377.
3. Бурмистров В.Д., Заковряшин Е.М. Создание хранилища данных для распределенной системы // Молодой ученый. 2016. № 12. С. 143–147.
4. Bogatyrev V.A., Parshutina S.A., Poptcova N.A., Bogatyrev A.V. Efficiency of redundant service with destruction of expired and irrelevant request copies in real-time clusters // Communications in Computer and Information Science. 2016. V. 678. P. 337–348. doi: 10.1007/978-3-319-51917-3_30
5. Farley M. Building Storage Networks. 2nd ed. Osborne: McGraw-Hall, 2001. 576 p.
6. Богатырев В.А., Богатырев С.В., Богатырев А.В. Надежность кластерных вычислительных систем с дублированными связями серверов и устройств хранения // Информационные технологии. 2013. № 2. С. 27–32.
7. Information Storage and Management. 2nd ed. New Jersey:

References

1. Bogatyrev V.A., Bogatyrev S.V. Association reservation servers in clasters highly reliable computer system. *Informatsionnye Tekhnologii*, 2009, no. 6, pp. 41–47. (in Russian)
2. Proskuryakov N.E., Anufrieva A.Y. Analysis and prospects of modern systems of storage of figures. *News of the Tula State University. Technical Sciences*, 2013, no. 3, pp. 368–377. (in Russian)
3. Burmistrov V.D., Zakovryashin E.M. Creating a data warehouse for a distributed system. *Molodoi Uchenyi*, 2016, no. 12, pp. 143–147. (in Russian)
4. Bogatyrev V.A., Parshutina S.A., Poptcova N.A., Bogatyrev A.V. Efficiency of redundant service with destruction of expired and irrelevant request copies in real-time clusters. *Communications in Computer and Information Science*, 2016, vol. 678, pp. 337–348. doi: 10.1007/978-3-319-51917-3_30
5. Farley M. *Building Storage Networks*. 2nd ed. Osborne, McGraw-Hall, 2001, 576 p.
6. Bogatyrev V.A., Bogatyrev S.V., Bogatyrev A.V. Reliability clusters computing systems with the duplicated communications of servers and storage devices. *Information Technologies*, 2013, no. 2, pp. 27–32. (in Russian)
7. *Information Storage and Management*. 2nd ed. New Jersey, John Wiley & Sons Inc., 2016, 544 p.

John Wiley & Sons Inc., 2016. 544 p.
8. Landauer R. Information is physical // Physics Today. 1991. V. 44. N 5. P. 23–29.
9. Todman C. Designing a Data Warehouse: Supporting Customer Relationship Management. Prentice Hall Publ., 2001. 414 p.
10. Kish L.B., Granqvist C.G. Does information have mass? // Proceedings of the IEEE. 2013. V. 101. N 9. P. 1895–1899. doi: 10.1109/JPROC.2013.2273720
11. Mesnier M., Ganger G., Riedel E. Object-based storage // IEEE Communications Magazine. 2003. V. 41. N 8. P. 84–90. doi: 10.1109/MCOM.2003.1222722
12. Buyya R., Broberg J., Goscinski A. Cloud Computing. Principles and Paradigms. New Jersey: John Wiley & Sons Inc., 2011. 637 p.
13. Татарникова Т.М., Пойманова Е.Д. Технологии долговременного хранения данных // Материалы Международной научно-практической конференции «Наука и образование в XXI веке». Тамбов, 2013. Ч. 31. С. 136–138.
14. Татарникова Т.М. Анализ данных. СПб: СПбГЭУ, 2018. 85 с.
15. Kish L.B. Moore's law and the energy requirement of computing versus performance // IEEE Proceedings: Circuits, Devices and Systems. 2004. V. 151. N 2. P. 190–194. doi: 10.1049/ip-cds:20040434
16. Hastic T., Tibshirani R., Friedman J. The Elements of Statistical Learning. Springer, 2003. 552 p. doi: 10.1007/978-0-387-21606-5
17. Kohonen T. Self-Organizing Maps. NY, Springer, 1995. 362 p.
18. Stacey M., Salvatore J., Jorgensen A. Visual Intelligence: Microsoft Tools and Techniques for Visualizing Data. New Jersey: Wiley, 2013. 400 p.
19. Morville P., Callender J. Search Patterns: Design for Discovery. O'Reilly Media, 2010. 192 p.
20. Kutuzov O.I., Tatarnikova T.M. Model of a self-similar traffic generator and evaluation of buffer storage for classical and fractal queuing system // Proc. Moscow Workshop on Electronic and Networking Technologies, MWENT. Moscow, 2018. doi: 10.1109/mwent.2018.8337306
21. Кутузов О.И., Татарникова Т.М. Из практики применения метода Монте-Карло // Заводская лаборатория. Диагностика материалов. 2017. Т. 83. № 3. С. 65–70.

8. Landauer R. Information is physical. *Physics Today*, 1991, vol. 44, no. 5, pp. 23–29.
9. Todman C. *Designing a Data Warehouse: Supporting Customer Relationship Management*. Prentice Hall Publ., 2001, 414 p.
10. Kish L.B., Granqvist C.G. Does information have mass? *Proceedings of the IEEE*, 2013, vol. 101, no. 9, pp. 1895–1899. doi: 10.1109/JPROC.2013.2273720
11. Mesnier M., Ganger G., Riedel E. Object-based storage. *IEEE Communications Magazine*, 2003, vol. 41, no. 8, pp. 84–90. doi: 10.1109/MCOM.2003.1222722
12. Buyya R., Broberg J., Goscinski A. *Cloud Computing. Principles and Paradigms*. New Jersey, John Wiley & Sons Inc., 2011, 637 p.
13. Tatarnikova T.M., Poimanova E.D. Long term storage technologies. *Proc. Int. Conf. on Science and Education in the 21st Century*. Tambov, Russia, 2013, part 31, pp. 136–138. (in Russian)
14. Tatarnikova T.M. *Data Analysis*. St. Petersburg, SPbSEU Publ., 2018, 85 p. (in Russian)
15. Kish L.B. Moore's law and the energy requirement of computing versus performance. *IEE Proceedings: Circuits, Devices and Systems*, 2004, vol. 151, no. 2, pp. 190–194. doi: 10.1049/ip-cds:20040434
16. Hastic T., Tibshirani R., Friedman J. *The Elements of Statistical Learning*. Springer, 2003, 552 p. doi: 10.1007/978-0-387-21606-5
17. Kohonen T. *Self-Organizing Maps*. NY, Springer, 1995, 362 p.
18. Stacey M., Salvatore J., Jorgensen A. *Visual Intelligence: Microsoft Tools and Techniques for Visualizing Data*. New Jersey: Wiley, 2013, 400 p.
19. Morville P., Callender J. *Search Patterns: Design for Discovery*. O'Reilly Media, 2010, 192 p.
20. Kutuzov O.I., Tatarnikova T.M. Model of a self-similar traffic generator and evaluation of buffer storage for classical and fractal queuing system. *Proc. Moscow Workshop on Electronic and Networking Technologies, MWENT*. Moscow, 2018. doi: 10.1109/mwent.2018.8337306
21. Kutuzov O.I., Tatarnikova T.M. Practical experience of using Monte Carlo method. *Industrial Laboratory*, 2017, vol. 83, no. 3, pp. 65–70. (in Russian)

Авторы

Татарникова Татьяна Михайловна – доктор технических наук, доцент, профессор, Санкт-Петербургский государственный университет аэрокосмического приборостроения, Санкт-Петербург, 190000, Российская Федерация; профессор, Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина), Санкт-Петербург, 197376, Российская Федерация, Scopus ID: 36715607400, ORCID ID: 0000-0002-6419-0072, Tm-tatarn@yandex.ru

Пойманова Екатерина Дмитриевна – старший преподаватель, Санкт-Петербургский государственный университет аэрокосмического приборостроения, Санкт-Петербург, 190000, Российская Федерация, ORCID ID: 0000-0002-7903-2480, e.d.poymanova@gmail.com

Authors

Tatiana M. Tatarnikova – D.Sc., Associate Professor, Professor, Saint Petersburg State University of Aerospace Instrumentation, Saint Petersburg, 190000, Russian Federation; Professor, Saint Petersburg Electrotechnical University "LETI", Saint Petersburg, 197376, Russian Federation, Scopus ID: 36715607400, ORCID ID: 0000-0002-6419-0072, Tm-tatarn@yandex.ru

Ekaterina D. Poymanova – Senior lecturer, Saint Petersburg State University of Aerospace Instrumentation, Saint Petersburg, 190000, Russian Federation ORCID ID: 0000-0002-7903-2480, e.d.poymanova@gmail.com