Оптимизация операций в СУБД на основе прогнозирования и понятия веса

А.В. Миронова

***Аннотация*— В условиях стремительного развития информационных технологий требования к информации и скорости её обработки увеличиваются столь же быстро. В статье предлагается метод повышения эффективности поиска элементов в широких таблицах с большим количеством столбцов, который также применим к задаче соединения таблиц. Метод основан на специфическом индексировании элементов, используя концепцию, близкую к понятию нормы пространства — веса. Это позволяет формировать эквивалентное отношение на множестве элементов таблицы и одновременно устранить некоторые ограничения при использовании индексирования от элементов, такие как соотношение порядка характеристик или ограничения на типы обрабатываемых данных в таблицах. Кроме того, в статье рассматривается прогнозирование объёма результата соединения на основе метаданных без непосредственного выполнения соединения. Такая концепция позволяет выстраивать цепочку соединений оптимальным или близким к оптимальному образом, что значительно повышает эффективность операции соединения за счёт уменьшения размера обрабатываемых таблиц.**

***Ключевые слова*— нормы, вес кортежа, BigData, поиск сложных структур, оптимизация JOIN, базы данных, распределенные вычисления, фактор-множество, вес элемента, индексы, хеши.**

I. Введение

В статье рассматривается возможность применения распределения, а также понятия близкого к понятию нормы элемента для дополнительной оптимизации последовательности поисковых запросов к базам данным. Современные сервисы зачастую сталкиваются с проблемами чрезвычайной нагрузки за счет следующих факторов:

- Тенденция на усложнение структур данных;
- Количество данных неуклонно растет[1];
- Общность данных снижается что влечет за собой снижение эффективности универсальной обработки данных для каждого случая;
- Вместе с усложнением структуры самих данных повышается сложность запроса, при обращении к этим данным.

В настоящее время подобные задачи чаще всего решаются за счёт экстенсивных подходов, главным образом путём увеличения числа процессоров и прямого разделения данных [2, 3]. Однако такие подходы оказываются затратными и имеют строгие ограничения при применении к различным операциям. Например, увеличение числа вычислительных кластеров в конечном итоге может привести к перегрузке шины данных и другим негативным последствиям, которые снижают эффективность выбранных методов оптимизации.

Вместе с этим рассматривается возможность оптимизации цепочки запросов по типу join, путем прогнозирования мощности результатов выполнения при различной последовательности выполнения операций. При возможности хотя бы примерно представить себе объем рассуждая одним и тем же способом, можно сильно сэкономить время выполнения операции основываясь на том, что разные промежуточные результаты дают различную оценку общего времени.

II. Цель исследования

Статья изучает применение распределения и нормирования для улучшения оптимизации последовательностей поисковых и соединительных запросов к базам данных большого объема (Big Data). Многие современные сервисы сталкиваются с проблемами высокой нагрузки за счет роста данных и усложнения их структуры.

Предложенная концепция в перспективе может помочь оптимизировать большую долю запросов, происходящих для получения информации из реляционных СУБД за счет правильной оценки метаданных существующих таблиц.

III. Введение понятия веса

Для оптимизации, разбиения по вычислителям и кластеризации необходимо ввести для каждого элемента признак, который назовем весом элемента w. Предположим, что имеется таблица с подмножеством столбцов, которые нуждаются в оптимизации поиска и соединения.

$$\left(x_1, x_2, x_3, ..., x_n\right).$$

Для простоты изложения будем брать в рассмотрения числовой формат (не теряя общности). Для каждого столбца выберем элемент и назовем его точкой отсчета.

$$\left(A_1', A_2', A_3', ..., A_n'\right).$$

Будем говорить для i-ой координаты элемента t, что ее удельный вес равен единице, если для x_i выполняется условие

Статья получена 7 марта 2024.

А.В. Миронова – Смоленский государственный университет (email: anastasiakiricova52@gmail.com)

$$\frac{e}{2^1} \geq \left| X_i - A_i' \right|.$$

Иными словами, если по i-ой координате t находится несколько дальше от точки отсчета по данному столбцу. Весом элемента t назовем следующую сумму:

$$w_i = \sum_{i=1}^{n} X_i,$$

где x_i -координаты (значения t в i-тых столбцах) элемента x [рис1].

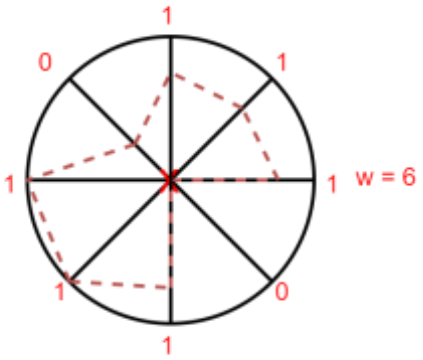

Рис. 1: вес элемента

Эту характеристику нужно определить для каждого элемента, и желательно хранить ее как метаданные для поиска. Очевидно, что для всех элементов таблицы можно вычислить эту характеристику, и также легко доказать, что у одинаковых элементов будут одинаковые веса. Эти два факта позволяют нам рассматривать данную характеристику как естественную перцептивную хеш-функцию, на основе которой можно улучшить эффективность поиска и операции объединения таблиц, исключая элементы с другим весом из соответствующей выборки. Кроме того, если необходим поиск по части столбцов среди элементов таблицы или же по сочетанию условий вида

$$x_1 = a_1 \vee x_2 = a_2 \vee x_3 = a_3 \ldots,$$

где a – поисковый образ, на данном этапе можно использовать правило следующее из неравенства треугольника для веса, поскольку каждый вес можно представить в качестве расстояния от нулевого элемента для таблицы – элемент, лежащий посреди каждого отрезка [4]:

$$\left| W_1 - W_2 \right| \leq \rho(x_1, x_2),$$

где под ρ – понимается расстояние Хемминга по весам координат[рис2]. Иными словами, если модуль разности весов равен определенной величине, расстояние Хемминга между весами координат элементов отличаются не более чем на эту величину.

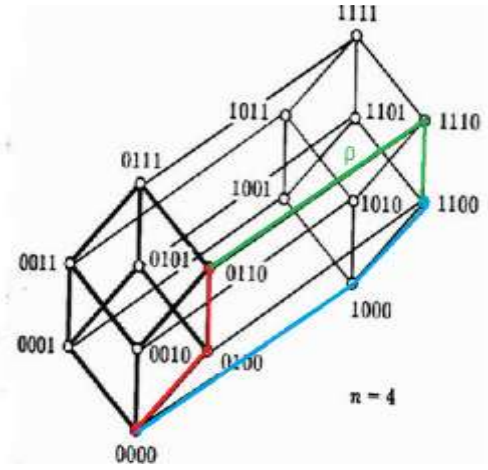

Рис. 2: расстояния хемминга между элементами на двоичном кубе

Для оптимизации поиска представляется возможным определить вес поискового образа и сначала выбирать для сравнения элементы с наиболее близким весом, так как они, вероятнее всего, будут соответствовать поисковым критериям по большинству столбцов, что увеличивает вероятность совпадения с условиями поиска.

Понятие веса также позволяет формировать отношения эквивалентности и делить множество элементов на классы эквивалентности, создавая симметричное распределение по множеству таблицы как начальный этап обработки сложных запросов, таких как цепочки операций Join по столбцам, входящим в определение веса. Кроме того, можно измерять веса выборок каждой группы и оценивать их возможное сходство. Поскольку вычисление веса не требует сложных вычислений и обеспечивает равномерное деление множества на подмножества с похожими элементами, такое разбиение может значительно повысить эффективность обработки трудоёмких запросов в распределённых системах за счёт эффективного распределения данных без возможных конфликтов.

IV. Разбиение по вычислителям на основании весов элементов

Понятие веса позволяет установить отношение эквивалентности на множестве элементов через сравнение их весов, что даёт возможность разделить множество на классы эквивалентности. Это, в свою очередь, порождает фактор-множество, которое можно использовать для создания горизонтально симметричного распределения элементов таблицы как первого шага при обработке сложных запросов, таких как цепочки операций Join по столбцам, связанным с понятием веса.

Для обеспечения более равномерного распределения по мощности множеств можно применить простой подход: в отсортированном по мощности списке классов эквивалентности для одного вычислителя выбирать первые и последние kkk классов, для второго — следующие kkk классов с обоих концов, и так далее. Это обеспечивает непересекающиеся множества элементов

на каждом вычислителе. Поскольку вычисление веса обладает низкой алгоритмической сложностью и позволяет достаточно равномерно разделить множество на подмножества, это разбиение может значительно повысить эффективность распределённой обработки сложных запросов, обеспечивая эффективное распределение данных без пересечений.

V. Кластеризация на основе понятия веса

При помощи весов можно произвести кластеризацию и более того индексирование таблицы для поиска.

Будем разбивать множество элементов таблицы по их весу на подмножества, далее получившиеся подмножества будут разбиты снова, когда в результирующих множествах не останется малого числа элементов или по достижении большого числа глубины процесс возможно остановить.

Каждый новый шаг можно назвать уровнем разбиения. На каждом уровне функция веса будет видоизменена с учетом глубины кластеризации. А именно положим, что при кластеризации на каждом j-том уровне будем говорить, что координата имеет вес 1 тогда, когда

$$\frac{e}{2^j} \geq \left| X_i - A_i' \right|$$

То есть для первого уровня, как и было сказано формула будет иметь вид

$$\frac{e}{2^1} \geq \left| X_i - A_i' \right|$$

Далее на втором уровне, когда изначальное множество будет разбито на подмножества, для каждого из этих подмножеств вес одной координаты будет равна 1 если

$$\frac{e}{2^2} \geq \left| X_i - A_i' \right|$$

И так далее – в общем виде можно сказать, что условие получения координатой веса на каждом уровне упрощается относительно общего отрезка возможных значений. Таким образом общую формулу веса элемента x на j-ом уровне для таблицы из n столбцов можно записать в следующем виде[рис 3]:

$$W(x) = \sum_{i=1}^{n} \frac{e}{2^j} \geq \left| X_i - A_i' \right|$$

Можно заметить, что с таким разделением и данной функцией веса на каждом шаге, количество элементов в множестве с увеличением числа вершин будет гарантировано уменьшаться [рис3]. Также из равенства функции веса для одного уровня и вышеизложенного следует, что одинаковые элементы на каждом шаге будут присутствовать в соответствующих результирующих множествах. Как для индекса, можно рассчитать, что в среднем количество элементов на каждом шаге по всем листьям будет крайне невелико.

Необходимо заметить, что все вышеизложенное распространяется и на один конкретный столбец, который в свою очередь можно представить в качестве двоичного вектора.

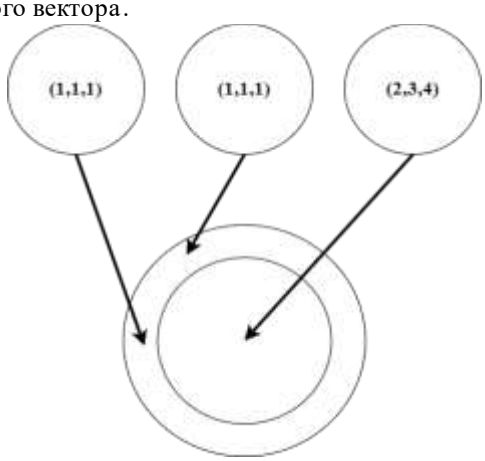

Рис. 3: определения элементов по весу

VI. Возможности оптимизация запросов типа Join

Вернувшись к озвученному выше

$$\left| W_1 - W_2 \right| \leq \rho(x_1, O_2),$$

где O – поисковый образ

Можно отметить, что при поиске нескольких объектов, которые потенциально схожи с другой группой нескольких объектов возможно использование прямого следствия

$$\left| W_1 - W_2 \right| + \left| W_4 - W_3 \right| \leq \rho(x_1, O_2) + \rho(x_4, O_3)$$

А значит можно утверждать

$$\left| W_1 - W_2 + W_4 - W_3 \right| \leq \rho(x_1, O_2) + \rho(x_4, O_3)$$

И также можно сказать:

$$\left| W_1 + W_4 - 2W_2 \right| \leq \rho(x_1, O_2) + \rho(x_4, O_2)$$

Между двумя группами элементов O1 и O2 расстояние можно посчитать как

$$\left| L_2 W_1 - L_1 W_2 \right| \leq \rho(O_1, O_2)$$

Легко проверить что суммарное расстояние можно точно рассчитать по формуле

$$\rho(O_1, O_2) = L_2 \cdot \sum_{i=1}^{L_1} w_{1,i} + L_1 \cdot \sum_{i=1}^{L_2} w_{2,i} - 2 \cdot \sum_{j=1}^{L_2} \sum_{i=1}^{L_1} w(x_i \wedge x_j)$$

Отсюда

$$2 \cdot \sum_{j=1}^{L_2} \sum_{i=1}^{L_1} w(x_i \wedge x_j) = L_2 \cdot \sum_{i=1}^{L_1} w_{1,i} + L_1 \cdot \sum_{i=1}^{L_2} w_{2,i} - \rho(O_1, O_2)$$

Если взглянуть на двоичные вектора суммой единиц которого и является вес, можно заключить что:

$$\sum_{j=1}^{L_2}\sum_{i=1}^{L_1} w(x_i \wedge x_j) \geq W\left(O_1 \; join O_2\right)$$

Поскольку для веса join необходимо строгое равенство. И тогда:

$$2 \cdot W(O_1 \; join \; O_2) \leq L_2 \cdot \sum_{i=1}^{L_1} w_{1,i} + L_1 \cdot \sum_{i=1}^{L_2} w_{2,i} - \rho\left(O_1, O_2\right),$$

$$2 \cdot W(O_1 \; join \; O_2) \leq L_2 \cdot \sum_{i=1}^{L_1} w_{1,i} + L_1 \cdot \sum_{i=1}^{L_2} w_{2,i} - \left|L_2 W_1 - L_1 W_2\right|,$$

$$W(O_1 \; join \; O_2) \leq \frac{\left(L_2 \cdot \sum_{i=1}^{L_1} w_{1,i} + L_1 \cdot \sum_{i=1}^{L_2} w_{2,i} - \left|L_2 W_1 - L_1 W_2\right|\right)}{2}.$$

Таким образом можно предполагать будущий вес основываясь только лишь на данных о множестве. Также понятно, что суммарный w вес коррелирует с длиной множества.

$$W_1 \square L_1$$

И все элементы множества результата join происходят из одного из двух множеств родительских, а значит хотя бы на уровне корреляции можно предположить, что:

$$L(join) \square \frac{W(join)}{\frac{W_1}{L_1} + \frac{W_2}{L_2}}$$

А значит имея последние три формулы можно от изначальных множеств, не производя никаких реальных соединений рассуждать об мощности результата соединения. После чего выстраивать join в нужной последовательности.

Таким образом, при выполнении операции JOIN появляется возможность заранее, без прямого сравнения, разбить группы элементов из двух множеств на основе их весов, что увеличивает скорость выполнения самой операции и сравнения множеств. Поскольку вычисление веса имеет низкую алгоритмическую сложность и позволяет равномерно разделить множество на подмножества с похожими элементами, такое разбиение может значительно повысить эффективность распределённой обработки сложных запросов за счёт оптимального распределения данных без пересечений. Кроме того, с помощью данного подхода можно распределить элементы между физическими вычислителями, что уменьшит затраты на перемещение данных между процессорами и снизит нагрузку на этап постобработки [9, 10].

VII. Заключение

Описанный выше метод не только эффективно позволяет искать равные или схожие последовательности ключей для операций поиска и сопоставления, но и формирует основу для распределённой системы, оптимизирующей более сложные задачи, такие как объединение (JOIN). Предложенный подход, использующий метаданные, в частности, концепцию веса, основанную на "заполняемости" столбцов для каждого элемента, обладает следующими преимуществами:

- Возможность эффективного разделения данных перед выполнением сложных запросов для их распределения по процессорам без пересечений данных;
- Возможность создания индексов структуры для оптимизации поиска данных без ограничений на элементы [11, 12];
- Возможность предварительной оптимизации и улучшения процесса соединения таблиц на основе веса элементов;
- Способ разбиения данных на основе отношения эквивалентности, которое сохраняется как метаданные.

Использование метаданных, основанных на наборе характеристик, наиболее эффективно при усложнении структуры обрабатываемых данных и увеличении сложности запросов к этим данным. Предлагаемое решение сохраняет ключевые черты индексных методов, но во многом решает проблему увеличения объёма памяти, необходимого для работы с большим количеством столбцов и их более сложными типами. Это существенно расширяет возможности применения метода и позволяет оптимизировать цепочки операций JOIN за счёт анализа схожести элементов множества.

Библиография

[1] Abdel-Basset M. et al. An improved nature inspired meta-heuristic algorithm for 1-D bin packing problems //Personal and Ubiquitous Computing. – 2018. – Т. 22. – №. 5-6. – С. 1117-1132.

[2] Chamoso, Pablo, et al. "Social computing for image matching." PloS one 13.5 (2018): e0197576.

[3] Das S. et al. Automatically indexing millions of databases in microsoft azure sql database //Proceedings of the 2019 International Conference on Management of Data. – 2019. – С. 666-679.

[4] Kirikova A., Mironov A., Munerman V. The Method of Composition Hash-functions for Optimize a Task of Searching Images in Dataset //2020 IEEE Conference of Russian Young Researchers in Electrical

and Electronic Engineering (EIConRus). – IEEE, 2020. – С. 1983-1986.
[5] Dodonov A. et al. Method of Parallel Information Object Search in Unified Information Spaces //International Journal of Computer Network and Information Security (IJCNIS). – 2021. – Т. 13. – №. 4. – С. 1-13.
[6] Gorokhovatskyi V. A., Gorokhovatskiy A. V., Peredrii Y. O. Hashing of structural descriptions at building of the class image descriptor, computing of relevance and classification of the visual objects //Telecommunications and Radio Engineering. – 2018. – Т. 77. – №. 13.
[7] Graefe G. et al. Modern B-tree techniques //Foundations and Trends® in Databases. – 2011. – Т. 3. – №. 4. – С. 203-402.
[8] Haynes, David, et al. "High performance analysis of big spatial data." 2015 IEEE International Conference on Big Data (Big Data). IEEE, 2015.
[9] Munerman V.I. The experience of massive data processing in the cloud using windows azure (as an example) High availability systems. - 2014. - V. 10. - №. 2. - p. 8-13.
[10] Iljin P. L., Munerman V. J. Recursive computation of the multidimensional matrix determinant. Systems of computerized mathematics and their appendices: XX International Scientific Conference. Smolensk: SmolSU publishing. 2019. Vol. 1, Issue 20. pp. 162-166. (In Russ).
[11] Kirikova A., Mironov A. Using Metadata-indexing to Improve the Efficiency of Complex Operations //2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). – IEEE, 2021. – С. 2124-2127.
[12] Levin N. A., Munerman V. I. Models of big data processing in massively parallel systems //Системы высокой доступности. – 2013. – Т. 9. – №. 1. – С. 035-043.
[13] Lomet D. The evolution of effective b-tree: Page organization and techniques: A personal account //ACM SIGMOD Record. – 2001. – Т. 30. – №. 3. – С. 64-69.
[14] Lvovich I. et al. Modeling and optimization of processing large data arrays in information systems //2021 International Conference on Information Technology and Nanotechnology (ITNT). – IEEE, 2021. – С. 1-5.
[15] Monga, Vishal, and Brian L. Evans. "Perceptual image hashing via feature points: performance evaluation and tradeoffs." IEEE Transactions on Image Processing 15.11 (2006): 3452-3465.
[16] Munerman V., Munerman D. Realization of Distributed Data Processing on the Basis of Container Technology //2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). – IEEE, 2019. – С. 1740-1744.
[17] Munerman V., Munerman D., Samoilova T. The Heuristic Algorithm For Symmetric Horizontal Data Distribution //2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). – IEEE, 2021. – С. 2161-2165.
[18] Pushpa Rani Suri, Sudesh Rani, "A New Classification for Architecture of Parallel Databases", Information Technology Journal, vol. 7, pp. 983. (2008).
[19] Pyurova T. A., Skvortsov S. V. CUDA technology and parallel computing On GPU //Informatics and applied mathimatic: interuniversity compendium of treatises, no. 21, pp. 163-166. 2015. (In Russ).
[20] Sridhar R. et al. Optimization of heterogeneous Bin packing using adaptive genetic algorithm //IOP Conference Series: Materials Science and Engineering. – IOP Publishing, 2017. – Т. 183. – №. 1. – С. 012026. 16.
[21] Syrotkina O. et al. Mathematical Methods for optimizing Big Data Processing //2020 10th International Conference on Advanced Computer Information Technologies (ACIT). – IEEE, 2020. – С. 170-176.
[22] Wajszczyk B., Gruszka I. M. Analysis of possibilities to increase the efficiency of the relative database management system using the methods of parallel processing //Radioelectronic Systems Conference 2019. – SPIE, 2020. – Т. 11442. – С. 385-398.
[23] Zakharov V. et al. Architecture of Software-Hardware Complex for Searching Images in Database //2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). – IEEE, 2019. – С. 1735-1739.
[24] Zobel J., Moffat A., Sacks-Davis R. An efficient indexing technique for full-text database systems //PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES. – INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS (IEEE), 1992. – С. 352-352.

Optimization of search queries by introducing the concept of weight

Anastasia Mironova

***Abstract* - With the rapid development of information technology, the requirements for information and the speed of its processing are increasing just as quickly. This article proposes a method for improving the efficiency of element searches in wide tables with a large number of columns, which is also applicable to the task of table joins. The method is based on specific indexing of elements, using a concept similar to the notion of space norm — weight. This allows the formation of an equivalence relation on the set of table elements, while simultaneously eliminating certain limitations associated with indexing by elements, such as the relationship between the order of characteristics or restrictions on the types of data being processed in the tables. Additionally, the article discusses the prediction of the join result size based on metadata without actually performing the join. This concept enables the construction of join sequences in an optimal or near-optimal manner, significantly improving the efficiency of the join operation by reducing the size of the tables being processed.**

***Keywords* - norms, tuple weight, Big Data, complex structure search, JOIN optimization, databases, distributed computing, factor set, element weight, indexes, hashes.**

REFERENCES

[1] Abdel-Basset M. et al. An improved nature inspired meta-heuristic algorithm for 1-D bin packing problems //Personal and Ubiquitous Computing. – 2018. – Т. 22. – №. 5-6. – С. 1117-1132.

[2] Chamoso, Pablo, et al. "Social computing for image matching." PloS one 13.5 (2018): e0197576.

[3] Das S. et al. Automatically indexing millions of databases in microsoft azure sql database //Proceedings of the 2019 International Conference on Management of Data. – 2019. – С. 666-679.

[4] Kirikova A., Mironov A., Munerman V. The Method of Composition Hash-functions for Optimize a Task of Searching Images in Dataset //2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). – IEEE, 2020. – P. 1983-1986.

[5] Dodonov A. et al. Method of Parallel Information Object Search in Unified Information Spaces //International Journal of Computer Network and Information Security (IJCNIS). – 2021. – Т. 13. – №. 4. – С. 1-13.

[6] Gorokhovatskyi V. A., Gorokhovatskiy A. V., Peredrii Y. O. Hashing of structural descriptions at building of the class image descriptor, computing of relevance and classification of the visual objects //Telecommunications and Radio Engineering. – 2018. – Т. 77. – №. 13.

[7] Graefe G. et al. Modern B-tree techniques //Foundations and Trends® in Databases. – 2011. – Т. 3. – №. 4. – С. 203-402.

[8] Haynes, David, et al. "High performance analysis of big spatial data." 2015 IEEE International Conference on Big Data (Big Data). IEEE, 2015.

[9] Munerman V.I. The experience of massive data processing in the cloud using windows azure (as an example) High availability systems. - 2014. - V. 10. - №. 2. - p. 8-13.

[10] Iljin P. L., Munerman V. J. Recursive computation of the multidimensional matrix determinant. Systems of computerized mathematics and their appendices: XX International Scientific Conference. Smolensk: SmolSU publishing. 2019. Vol. 1, Issue 20. pp. 162-166. (In Russ).

[11] Kirikova A., Mironov A. Using Metadata-indexing to Improve the Efficiency of Complex Operations //2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). – IEEE, 2021. – С. 2124-2127.

[12] Levin N. A., Munerman V. I. Models of big data processing in massively parallel systems //Системы высокой доступности. – 2013. – Т. 9. – №. 1. – С. 035-043.

[13] Lomet D. The evolution of effective b-tree: Page organization and techniques: A personal account //ACM SIGMOD Record. – 2001. – Т. 30. – №. 3. – С. 64-69.

[14] Lvovich I. et al. Modeling and optimization of processing large data arrays in information systems //2021 International Conference on Information Technology and Nanotechnology (ITNT). – IEEE, 2021. – С. 1-5.

[15] Monga, Vishal, and Brian L. Evans. "Perceptual image hashing via feature points: performance evaluation and tradeoffs." IEEE Transactions on Image Processing 15.11 (2006): 3452-3465.

[16] Munerman V., Munerman D. Realization of Distributed Data Processing on the Basis of Container Technology //2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). – IEEE, 2019. – С. 1740-1744.

[17] Munerman V., Munerman D., Samoilova T. The Heuristic Algorithm For Symmetric Horizontal Data Distribution //2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). – IEEE, 2021. – С. 2161-2165.

[18] Pushpa Rani Suri, Sudesh Rani, "A New Classification for Architecture of Parallel Databases", Information Technology Journal, vol. 7, pp. 983. (2008).

[19] Pyurova T. A., Skvortsov S. V. CUDA technology and parallel computing On GPU //Informatics and applied mathimatic: interuniversity compendium of treatises, no. 21, pp. 163-166. 2015. (In Russ).

[20] Sridhar R. et al. Optimization of heterogeneous Bin packing using adaptive genetic algorithm //IOP Conference Series: Materials Science and Engineering. – IOP Publishing, 2017. – Т. 183. – №. 1. – С. 012026. 16.

[21] Syrotkina O. et al. Mathematical Methods for optimizing Big Data Processing //2020 10th International Conference on Advanced Computer Information Technologies (ACIT). – IEEE, 2020. – С. 170-176.

[22] Wajszczyk B., Gruszka I. M. Analysis of possibilities to increase the efficiency of the relative database management system using the methods of parallel processing //Radioelectronic Systems Conference 2019. – SPIE, 2020. – Т. 11442. – С. 385-398.

[23] Zakharov V. et al. Architecture of Software-Hardware Complex for Searching Images in Database //2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). – IEEE, 2019. – С. 1735-1739.

[24] Zobel J., Moffat A., Sacks-Davis R. An efficient indexing technique for full-text database systems //PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES. – INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS (IEEE), 1992. – С. 352-352.