UDC 004.65
DOI https://doi.org/10.32689/maup.it.2023.3.8

Valerii NIKITIN
Postgraduate Student at the Department of Information Systems and Technologies, Igor Sikorsky Kyiv Polytechnic Institute (19valeranikitin96@gmail.com)
ORCID: 0000-0002-4509-1204

Evgen KRYLOV
Candidate of Technical Sciences, Senior Lecturer at the Department of Information Systems and Technologies, Igor Sikorsky Kyiv Polytechnic Institute (ekrylov1964@ gmail.com)
ORCID: 0000-0003-4313-938X

Валерій НІКІТІН
аспірант кафедри інформаційних систем та технологій, КПІ ім. Ігоря Сікорського (19valeranikitin96@gmail.com)

Євген КРИЛОВ
кандидат технічних наук, доцент кафедри інформаційних систем та технологій, КПІ ім. Ігоря Сікорського (ekrylov1964@gmail.com)

Bibliographic description of the article: Nikitin, V., Krylov, E. (2023). Mekhanizm Active Anti-Entropy na osnovi spektralnoho filtru Bluma ta PH-2 alhorytmu kheshuvannia dlia uzghodzhennia replik u nereliatsiinykh rozpodilenykh dokumentooriientovanykh bazakh danykh [Active Anti-Entropy mechanism based on Spectral Bloom Filter and PH-2 hash algorithm for reconcilation of replicas of NoSQL distributed document oriented databases]. *Informatsiini tekhnolohii ta suspilstvo – Information technology and society,* 3, 63–67. DOI: https://doi.org/10.32689/maup.it.2023.3.8

Бібліографічний опис статті: Нікітін В., Крилов Є. Механізм Active Anti-Entropy на основі спектрального фільтру Блума та PH-2 алгоритму хешування для узгодження реплік у нереляційних розподілених документоорієнтованих базах даних. *Інформаційні технології та суспільство.* 3, 63–67. DOI: https://doi.org/10.32689/maup.it.2023.3.8

ACTIVE ANTI-ENTROPY MECHANISM BASED ON SPECTRAL BLOOM FILTER AND PH-2 HASH ALGORITHM FOR RECONCILATION OF REPLICAS OF NOSQL DISTRIBUTED DOCUMENT ORIENTED DATABASES

Abstract. *Information systems are used in many areas of human activity, which are not limited to one country or continent. This may require horizontal scaling for the system to function properly. Ignoring this can affect performance and availability, which in turn can lead to a loss of reputation and users.*

Horizontal scaling increases the number of database replicas, which creates the need for data reconciliation, since writing operations to different nodes increases entropy. There are various technologies aimed at reducing it, including Active Anti-Entropy. Its essence is to detect inconsistencies and start the reconciliation process between replicas. It is actively used in a database such as Riak and uses the Merkle Tree data structure, which is based on the use of hashing algorithms. The speed of inconsistency identification depends on the chosen hashing algorithms and the number of documents in the collection. An increase in the number of documents or even their size can worsen the even distribution and lead to an increase in the number of collisions. The occurrence of collisions increases the time period of data inconsistency, because the system cannot detect the inconsistency in time.

In addition to the collisions that can occur, you need to consider the delay due to data transfer over the network when nodes interact, and remember that such verification is not a one-time operation, but requires constant computation on replicas and sending for verification. Minimizing the time of these operations will speed up the data reconciliation process.

Critically important data must be reconciled with minimal delay, as an untimely or incorrectly made decision can lead to material or even human losses. To prevent this, there must be a solution that will minimize the delay of matching such data.

Key words: *NoSQL, distributed system, Active Anti-Entropy, Spectral Bloom Filter, consistency, PH2 hash algorithm.*

МЕХАНІЗМ ACTIVE ANTI-ENTROPY НА ОСНОВІ СПЕКТРАЛЬНОГО ФІЛЬТРУ БЛУМА ТА PH-2 АЛГОРИТМУ ХЕШУВАННЯ ДЛЯ УЗГОДЖЕННЯ РЕПЛІК У НЕРЕЛЯЦІЙНИХ РОЗПОДІЛЕНИХ ДОКУМЕНТО-ОРІЄНТОВАНИХ БАЗАХ ДАНИХ

Анотація. *Інформаційні системи використовуються у багатьох сферах діяльності людини, які не обмежуються однією країною або континентом. Це може призводити до необхідності горизонтального масштабування, щоб система могла нормально функціонувати. Ігнорування цього може впливати на швидкодію та доступність, що у свою чергу призведе до втрати репутації та користувачів.*

При горизонтальному масштабуванні збільшується кількість реплік бази даних, що створює необхідність в узгодженні даних, оскільки операції запису до різних вузлів збільшує ентропію. Є різні технології, які направлені на її зменшення, серед яких Active Anti-Entropy. Суть її полягає у тому, щоб виявити неконсистентність та розпочати процес узгодження між репліками. Вона активно використовується у такій базі даних, як Riak та використовує структуру даних Merkle Tree, яка базується на використанні алгоритмів хешування. Швидкість ідентифікування неузгодженості залежить від обраних алгоритмів хешування та кількості документів в колекції. Збільшення кількості документів або навіть їх розмір може погіршувати рівномірний розподіл та призводити до збільшення кількості колізій. Виникнення колізій збільшує проміжок часу неузгодженості даних, оскільки система не може вчасно виявити неконсистентність.

Окрім колізій, які можуть виникати, потрібно враховувати затримку через передачу даних мережею при взаємодії вузлів та пам'ятати, що така перевірка не є поодинокою операцією, а вимагає постійного обчислення на репліках та відправки для перевірки. Мінімізація часу виконання цих операцій дозволить пришвидшити процес узгодження даних.

Критично важливі дані повинні бути узгоджені з мінімальною затримкою, оскільки невчасно або неправильно прийняте рішення може призвести до матеріальних, або навіть людських втрат. Для запобігання цьому, повинно існувати рішення, яке дозволить мінімізувати затримку узгодження таких даних.

Ключові слова: *нереляційна база даних, розподілена система, активна антіентропія, спектральний фільтр Блума, консистентність, алгоритм хешування PH2.*

Problem statement. Horizontal scaling of information systems increases speed and availability due to increased computing power, but in turn, creates additional tasks. One of these tasks is the reconciliation of data on different replicas of a distributed database [1].

Since digitalization permeates almost all spheres of human activity, there may be completely different cases that emphasize the need to pay attention to it. It can be the coordinates of objects in space, and various financial transactions that can be carried out in different point of the Earth. Given the development of mobile technologies, which cause an increase in bandwidth and radius of coverage, the opportunities for automating processes that could not be automated before are increasing. In addition, such a direction as the Internet of Things is actively developing, which requires a large number of sensors, which in turn increase the amount of information in cyber-physical systems. It should be noted that technologies are actively being implemented in the goverment sector of countries and information about citizens should be consistent for all public services from different sources. In addition, the consistency of certain data on citizens can be useful for partner countries to simplify the work of border services, perhaps even to find the necessary qualified engineers or scientists [2].

Latest research and publications analysis. Distributed databases can use different techniques to maintain consistency. They can represent not only additional mechanisms, but be provided at the level of the architecture of the database itself and its CRUD operations [3].

One method may be to centralize write operations, which guarantees synchronous writing to replicas. Data is read from replicas, which significantly reduces the load from the central node. In addition, the presence of replicas improves the availability of the database, because if one node goes down, it is possible to continue work using copies.

Ведення версій записів також є методом вирішення конфліктів, яке використовує вектор часу. Вектор часу – це послідовність пар, яка описує порядок поновлення цього запису. Перевагами вектору часу є відсутність єдиної точки відмови системи, тому що при використанні тимчасових міток в записах необхідно виконувати точну синхронізацію часу з одним еталоном. Недоліками вектору часу є відсутність можливості автоматично вирішувати конфлікти, а також збільшення довжини вектору часу при багаторазовому оновленні запису. Однак в NoSQL існують механізми урізання вектору часу. Наприклад, в системі Riak можна задавати частоту обрізання вектору на рівні сегмента, а також максимальний розмір (довжину) вектора часу [4].

Active Anti-Entropy mechanism consists in detecting and correcting inconsistency. The active antientropy process involves periodically comparing and synchronizing data between nodes to detect any differences. This ensures that copies of data on different nodes of a distributed system always remain consistent with each other [5].

A Merkle tree is a data structure used in distributed systems to effectively check the integrity of data in the Active Anti-Entropy mechanism. It is created by recursively hashing pairs of data, creating a tree-like structure where each leaf represents a particular piece of data and each non-leaf represents the hash of its children. The highest level of the tree, known as the root node, contains a single hash value, often called the root Merkle hash [6].

Merkle trees are often used to ensure data integrity in blockchain technologies. By comparing root Merkle hashes between different nodes, it is possible to effectively determine whether the data between these nodes is consistent or whether there are differences.

Bloom filter is a probabilistic data structure that is used to efficiently determine whether a certain element belongs to a set of data. Use this filter to quickly respond to queries about the presence or absence of certain data in a set without having to store all of that data separately [7].

One of the varieties of this data structure is the Spectral Bloom filter, which is a vector of counters. The value of the counter increases when it is accessed accordingly. The peculiarity is that it becomes possible to add and remove elements that have been added to the filter, unlike the classic Bloom filter [8].

Aim of the research. The main goal of the research is to present the proposed Active Anti-Entropy mechanism for matching critical data in distributed NoSQL document-oriented databases, as existing methods are vulnerable due to insufficient collision resistance in the context of critical data.

Active Anti-Entropy mechanism based on Spectral Bloom Filter and PH-2 algorithm. The Active Anti-Entropy mechanism is a process that occurs in the background of a running distributed database. The essence of the mechanism is to search for entropy and perform the process of matching data between different replicas.

The diagram of the Active Anti-Entropy mechanism using the Spectral Bloom filter and the PH-2 algorithm is shown in Fig. 1.

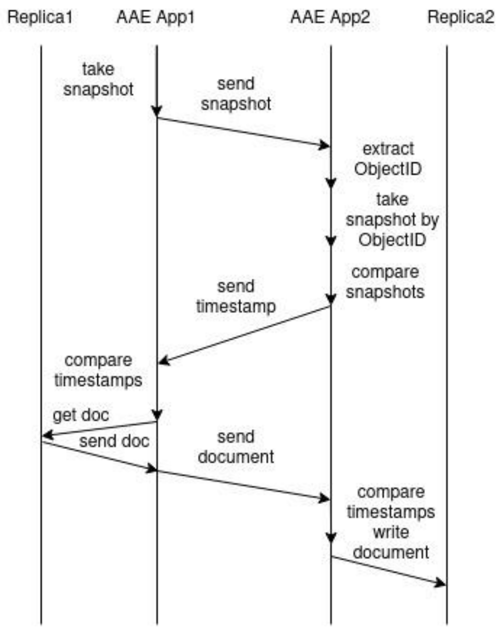


Fig. 1. Scheme of operation of the proposed Active Anti-Entropy mechanism

The essence of the mechanism is that there is a constant exchange of snapshots between the nodes, which are compared with each other. Consider the mechanism on the example of a distributed NoSQL document-oriented database, which consists two replicas.

The first replica takes a snapshot for a particular document and sends it to another replica. When the second replica received this snapshot, it calculates the snapshot of the same document by the document ID and compares it with what it received. If the snapshots are the same, the second replica simply ignores the received snapshot. In this case, this node can send a certain message to the 1st node that the data is agreed and no additional actions are required. If the snapshots are different, the second node sends the document identifier and the timestamp of its last modification. This is necessary so that the first node can determine which replica stores the most recent data. If, after comparing the timestamps, it turns out that the first replica contains a newer document, then the document is taken from it and sent to the second node. The second node receives the document, compares the timestamps, and writes it to the database if the time of the last modification of the local document is older than the time of the received document.

If the first replica contains an outdated document, it simply ignores that the timestamps are different because the second node performs the same actions in parallel, providing simultaneous monitoring.

Since the exchange of messages occurs continuously, the following requirements for messages arise:

– the snapshot must be collision-resistant;

– the snapshot calculation speed should be maximum;

– the size of the snapshot should be minimal to reduce the time required for transmission between nodes over the computer network;

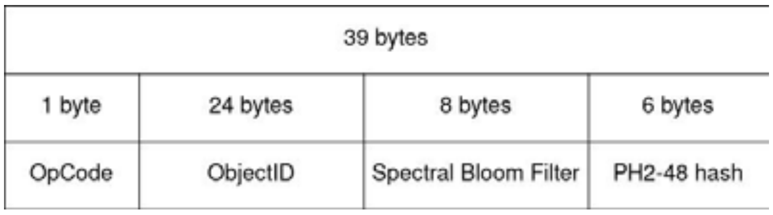
Collision resistance is a critically important indicator, as it depends on how quickly the mechanism can detect inconsistencies and start the data reconciliation process.

Alternatively, cryptographic hashing algorithms could be applied, but these have more security-related properties. For consistency purposes, this is irrelevant, but can negatively affect the needs of fast computation and size. In other words, they do not meet the second and third requirements. In this case, non-cryptographic hashing algorithms can be used, but the issue of collision resistance remains open for them.

Another possible option is to use a spectral Bloom filter. In its classic form, it is a vector of counters that is formed from input data. The counter is an unsigned integer. Each input block of data is hashed, a digest is obtained and the position in the vector is calculated using it. When addressing a certain element of this vector, the value of the counter is increased by one.

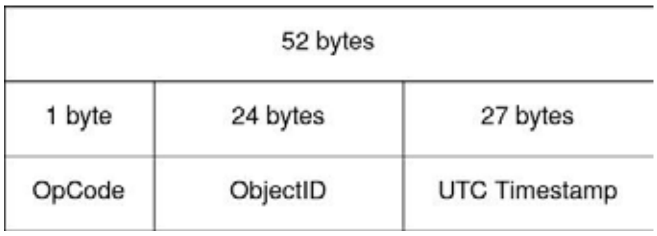
It would be advisable to use it if you change the algorithm of forming this filter and avoid usage of hash functions. Also, it would be advisable to use such an algorithm, which would target data that in most cases is stored in databases to increase collision resistance.

In addition to the snapshot, the message may include the operation ID, document ID, PH2 hash, and timestamp. The structures of possible messages are shown in figures 2 and 3.



39 bytes			
1 byte	24 bytes	8 bytes	6 bytes
OpCode	ObjectID	Spectral Bloom Filter	PH2-48 hash

Fig. 2. Structure of message to check documents



52 bytes		
1 byte	24 bytes	27 bytes
OpCode	ObjectID	UTC Timestamp

Fig. 3. Structure of message to compare timestamps

The operation identifier is 1 byte in size and can take three values:

– if a message is sent to terminate the process, the field is set to 0;

– if a message is sent with a snapshot, the field takes the value 1;

– if a message is sent with a timestamp, the field takes the value 2.

These identifiers allow you to identify the type of message and process it in the appropriate way.

In the figures, there is an ObjectID that identifies a particular document in the database. Its size and format depends on the database for which mechanism is applied.

Figure 3 has a UTC Timestamp, which represents the time stamp when the last changes were made to the document. It is needed to determine the document that is newer on replicas. Its size and format can also be represented differently and depends on the precision with which time is described. For example, you can use a timestamp with seconds only, or you can also include microseconds or nanoseconds. Since not all databases have a dedicated API to retrieve this value, the mechanism must store it separately, or developers must add it to those documents that comply.

There is also a field for PH2-48 hash value, which is required to avoid collision situations. This algorithm is sensitive to changes in the size of the input data, which makes it useful in cases where the filters are the same when formed from different data arrays, but they are differ in size [9].

It should be noted that there is also a process termination operation when there is no need for negotiation, but the message in this case will consist only of the operation code, which is equivalent in size to 1 byte.

The mechanism uses UDP and TCP transport layer protocols for communication, which reduces the processing time of messages on the sender and receiver side. The use of such protocols as HTTP, HTTPs is impractical, as it creates an unnecessary load when encapsulating and decapsulating packets [10].

The UDP protocol is used for all operations. The TCP protocol ensures reliable transmission of documents when inconsistencies are detected. For the mechanism to work, you need to listen two ports at the same time to ensure operation state.

Discussion of the results and conclusions. Thus, the proposed method of the Active Anti-Entropy mechanism can solve the problem of consistency of critical data in distributed NoSQL document-oriented databases. The next step is the implementation of this mechanism for the existing NoSQL document-oriented database with the search for optimal means that will allow to achieve maximum collision resistance, message calculation speed and minimize message size for less delay due to transmission by computer networks. The implementation of such a subsystem will allow conducting experimental studies and identifying the strengths and weaknesses of the method itself.

Bibliography:

1. Changlin H. Survey on NoSQL Database Technology. *Journal of Applied Science and Engineering Innovation.* 2015. 2, 50-54. URL: http://www.jasei.pub/PDF/2-2/2-50-54.pdf
2. Muniswamaiah M., Agerwala T., C. Tappert C. Performance of databases in IoT applications. *2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom).* 2020. (190-192). New York, NY, USA : IEEE. URL: https://doi.org/10.1109/CSCloud-EdgeCom49738.2020.00041
3. K. Aguilera M., B. Terry D. The Many Faces of Consistency. *IEEE Database Engineering Bulletin.* 2016. 3-13. URL: http://sites.computer. org/debull/A16mar/p3.pdf
4. Belous R., Krylov E. TIME OPTIMIZATION OF PROCESS OF DATA CONSISTENCY IN NOSQL. *Herald of the Khmelnytskyi National University. Series: "Technical Sciences".* 2023. 3, 37-42. URL: http://journals. khnu.km.ua/vestnik/wp-content/uploads/2023/07/vknu-ts-2023-n3321-37-42.pdf
5. Nikitin V., Krylov E. A collision-resistant hashing algorithm for maintaining consistency in distributed NoSQL databases. *Adaptive Systems of Automatic Control Interdepartamental scientific and technical collection.* 2022. 2, 45-57. URL: https://doi.org/10.20535/1560-8956.41.2022.271338
6. Tarkoma S., Rothenberg C., Lagerspetz E. Theory and Practice of Bloom Filters for Distributed Systems. *IEEE Communications Surveys & Tutorials.* 2011. 14, 131-155. URL: https://doi.org/10.1109/SURV.2011.031611.00024
7. Cohen S., Matias Y. Spectral Bloom Filters. *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data.* 2003. 1-12. URL: http://dx.doi.org/10.1145/872757.872787
8. Nikitin V., Krylov E. Comparison of hashing methods for supporting of consistency in distributed databases. *Adaptive Systems of Automatic Control Interdepartmental scientific and technical collection.* 2022. 1, 48-53. URL: http://asac.kpi.ua/article/view/261646/258069
9. Al-Dhief F., Sabri N., Latiff N., Obaid O. Performance comparison between TCP and udp protocols in different simulation scenarios. *International Journal of Engineering & Technology.* 2018. 7, 172-176. URL: https://doi.org/10.14419/ijet.v7i4.36.23739

References:

1. Changlin, H. (2015). Survey on NoSQL Database Technology. *Journal of Applied Science and Engineering Innovation,* 2, 50-54. Retrieved from http://www.jasei.pub/PDF/2-2/2-50-54.pdf
2. Muniswamaiah, M., Agerwala, T., & C. Tappert, C. (2020). Performance of databases in IoT applications. *2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom),* (190-192). New York, NY, USA : IEEE. Retrieved from https://doi.org/10.1109/CSCloud-EdgeCom49738.2020.00041
3. K. Aguilera, M., & B. Terry, D. (2016). The Many Faces of Consistency. *IEEE Database Engineering Bulletin,* 3-13. Retrieved from http://sites.computer.org/debull/A16mar/p3.pdf
4. Belous, R., & Krylov, E. (2023). TIME OPTIMIZATION OF PROCESS OF DATA CONSISTENCY IN NOSQL. *Herald of the Khmelnytskyi National University. Series: "Technical Sciences",* 3, 37-42. Retrieved from http://journals.khnu.km.ua/vestnik/wp-content/uploads/2023/07/vknu-ts-2023-n3321-37-42.pdf
5. Nikitin, V., & Krylov, E. (2022). A collision-resistant hashing algorithm for maintaining consistency in distributed NoSQL databases. *Adaptive Systems of Automatic Control Interdepartamental scientific and technical collection,* 2, 45-57. Retrieved from https://doi.org/10.20535/1560-8956.41.2022.271338
6. Tarkoma, S., Rothenberg, C., & Lagerspetz, E. (2011). Theory and Practice of Bloom Filters for Distributed Systems. *IEEE Communications Surveys & Tutorials,* 14, 131-155. Retrieved from https://doi.org/10.1109/SURV.2011.031611.00024
7. Cohen, S., & Matias, Y. (2003). Spectral Bloom Filters. *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data,* 1-12. Retrieved from http://dx.doi.org/10.1145/872757.872787
8. Nikitin, V., & Krylov, E. (2022). Comparison of hashing methods for supporting of consistency in distributed databases. *Adaptive Systems of Automatic Control Interdepartmental scientific and technical collection,* 1, 48-53. Retrieved from http://asac.kpi.ua/article/view/261646/258069
9. Al-Dhief, F., Sabri, N., Latiff, N., & Obaid, O. (2018). Performance comparison between TCP and udp protocols in different simulation scenarios. *International Journal of Engineering & Technology,* 7, 172-176. Retrieved from https://doi.org/10.14419/ijet.v7i4.36.23739