https://doi.org/10.31891/csit-2023-3-9
UDC 004.65

Valerii NIKITIN, Evgen KRYLOV
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

PRIMARY-BASED SPECTRAL BLOOM FILTER FOR THE ENSURING CONSISTENCY IN DISTRIBUTED DOCUMENT-BASED NoSQL DATABASES USING ACTIVE ANTI-ENTROPY MECHANISM

The purpose of this work is to compare the existing methods of forming the Spectral Bloom filter using hash functions and the proposed method using prime numbers. The proposed method allows obtaining snapshots from documents that can be used to maintain data consistency in distributed document-oriented NoSQL databases as part of the Active Anti-Entropy mechanism. Data consistency is an important and challenging task due to the need for horizontal scaling of information systems. Neglecting this can lead to material or even human losses, since digitalization covers absolutely all spheres of human activity and there is a need for distributed processing and storage of information.

Consistency can be ensured in various ways, including an architectural approach and Active Anti-Entropy mechanisms. The architectural approach refers to centralized write operations that are distributed to secondary nodes. Accordingly, read operations take place from secondary nodes. This approach is not flexible, as it requires stable and fast communication with the central node, which is not always possible.

The Active Anti-Entropy mechanism is a background process that checks the consistency of data between nodes using special snapshots that can be obtained using hash functions or such a data structure as a Merkle Tree. Using the latter is ideal for checking the consistency of entire data sets, but for mission-critical data, this solution is not suitable. The probability of collisions or the computational cost can lead to inconsistency of the entire data set and this requires a special solution for critical data.

The proposed method makes it possible to obtain the Spectral Bloom filter from the original data set faster. In addition, it has higher collision resistance compared to the use of hash functions, which allows faster identification of inconsistencies in documents stored on different nodes.

Keywords: NoSQL, document-oriented databases, distributed databases, Spectral Bloom filter, consistency, Active Anti-Entropy

Валерій НІКІТІН, Євген КРИЛОВ
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

СПЕКТРАЛЬНИЙ ФІЛЬТР БЛУМА НА ОСНОВІ ПРОСТИХ ЧИСЕЛ ДЛЯ ВИКОРИСТАННЯ В ACTIVE-ANTI ENTROPY МЕХАНІЗМІ УЗГОДЖЕННЯ ДАНИХ У РОЗПОДІЛЕНІЙ ДОКУМЕНТООРІЄНТОВАНІЙ НЕРЕЛЯЦІЙНІЙ БАЗІ ДАНИХ

Метою даної роботи є порівняння існуючих методів формування спектрального фільтра Блума з використанням хеш-функцій та запропонованого методу з використанням простих чисел. Запропонований метод дозволяє отримати знімки з документів, які можна використовувати для підтримки узгодженості даних в розподілених документоорієнтованих NoSQL базах даних як частину механізму Active Anti-Entropy. Узгодженість даних є важливою та складною задачею через необхідність горизонтального масштабування інформаційних систем. Нехтування цим може призводити до матеріальних або навіть людських втрат, оскільки цифровізація охоплює абсолютно всі сфери діяльності людини і є необхідність у розподіленій обробці та зберіганні інформації.

Консистентність може забезпечуватись різними шляхами, серед яких архітектурний підхід та Active Anti-Entropy механізми. Під архітектурним підходом мається на увазі централізовані операції запису, які розповсюджуються до другорядних вузлів. Відповідно операції читання відбуваються з другорядних вузлів. Даний підхід не є гнучким, оскільки вимагає стабільного та швидкого зв'язку з центральним вузлом, що не завжди можливо.

Active Anti-Entropy механізм представляє собою фоновий процес, який перевіряє узгодженість даних між вузлами використовуючи спеціальні знімки, які можуть бути отримані з використанням хеш-функцій або такої структури даних як Merkle Tree. Використання останнього ідеально підходить для перевірки узгодженості цілих наборів даних, але для критично важливих даних це рішення не підходить. Ймовірність колізій або обчислювальні витрати можуть призводити до неузгодженості цілого набору даних і це вимагає спеціального рішення для критично важливих даних.

Запропонований метод дозволяє швидше отримувати спектральний фільтр Блума з вихідного набору даних. Окрім цього, він має вищу колізійну стійкість в порівнянні з використанням хеш-функцій, що дозволяє швидше ідентифікувати неузгодженість документів, які зберігаються на різних вузлах.

Ключові слова: нереляційні бази даних, документоорієнтовані бази даних, розподілені бази даних, спектральний фільтр Блума, узгодженість даних, Active Anti-Entropy технологія

Introduction

Distributed databases need special mechanisms that could ensure the necessary level of data consistency between replicas. Some modern databases solve this through architectural solutions. For example, a database like MongoDB uses a centralized approach for all write operations, while read operations are performed through replicas.

There is also a decentralized approach, in which there is no master node, and its functions are distributed among all replicas of the system. Cassandra and Riak work according to this principle. Riak, in turn, has several

additional mechanisms: Active Anti Entropy and Read Repair. Riak's Active-Anti Entropy technology is a background process that uses Merkle Tree to identify data inconsistencies. The identifiers received on different nodes in the form of hash values are compared with each other, which allows you to start the reconciliation process if necessary.

There are also some other methods of maintaining consistency that are a kind of centralized approach. For example, a transaction clock receives all transactions and merges them into a resulting transaction that is sent to the replicas. The advantage is that the resulting transaction will have the latest update and the number of writes to the database is optimized [1].

Related Works

Bloom Filter is a probabilistic data structure that is designed to check the presence of an element in a set. A filter consists of an array of values that take the value 1 or 0. To form the filter, hash functions are used that determine the elements of the array that will have the value 1. The size of the filter and the hash functions can be configured depending on the context of the task.

There are different variations of this data structure: Distributed Bloom Filter, Spectral Bloom Filter, Space-Code Bloom Filter. Each of these variants has its own differences and specific applications.

The Distributed Bloom Filter is a probabilistic data structure designed for distributed systems that require fast synchronization both spatially and temporally. This is achieved by sending a filter by each node with its ID to another node in the system, which results in a change in the probability of data replication. It should be noted that the probability of a false positive result decreases with an increase in the total number of nodes [2].

SCBF achieves a good compromise between counting accuracy and the number of bits used for counting. It represents a multiset, extending the capabilities of the traditional Bloom filter to represent a set. Given an element x, it not only allows you to check whether x is in the multiset, but also counts the number of occurrences of x. SCBF has several important usages in network measurement. For example, traffic measurement by flow for traffic design and anomaly detection [3].

Spectral Bloom Filter replaces a vector of bits with a vector of counters. Counters are positive natural numbers that have leading zeros. When calculating the position in the data vector, the corresponding counter is increased by one. Deleting an element proceeds in a similar way, but by subtracting 1 from the related counters [4].

Any hash function can be used for hashing, but since they can be combined and there is a limit of the size of the Bloom filter, it is difficult to calculate the probability of collisions. Even using an algorithm like PH-2, which is not a cryptographic algorithm and clearly defines collision situations, does not guarantee an optimal allocation [5].

The method of forming the Spectral Bloom filter using simple numbers

The Active-Anti Entropy mechanism is a background process of data reconciliation in distributed databases. The use of the Bloom filter to solve this problem is due to a convenient algorithm for forming a snapshot, which can be used to compare critically important documents that are located on different nodes of a distributed database.

To be used in the Active-Anti-Entropy mechanism, the Bloom filter must meet certain requirements:

1) low probability of collisions;
2) high speed of calculation;
3) small size.

Each of the above-mentioned components has an impact on the others, which requires the search for an optimal solution.

In the context of the task to support consistency between nodes of a distributed document-oriented database, the spectral Bloom filter is a set of eight non-negative integers. Each number is a counter that increases by one each time it is accessed. At the very beginning, the values of all counters are 0.

Access to a certain element is as follows:

1) represent the document as bytes;
2) each byte is converted to an unsigned integer;
3) division by modulo of each number by a set of prime numbers from 2 to 17 inclusive. If the remainder during division is zero, then the corresponding element in the Bloom filter is increased by one. If the number was not divided by any prime number from the set, then the last element of the filter is incremented by one;
4) if the number is not divisible by any prime number, then the last counter is increased;
5) if an integer overflow of the counter occurs, it is reset to zero.

Figure 1 shows the filter formation algorithm for a document consisting of 3x bytes. This document size was chosen for ease of visualization.

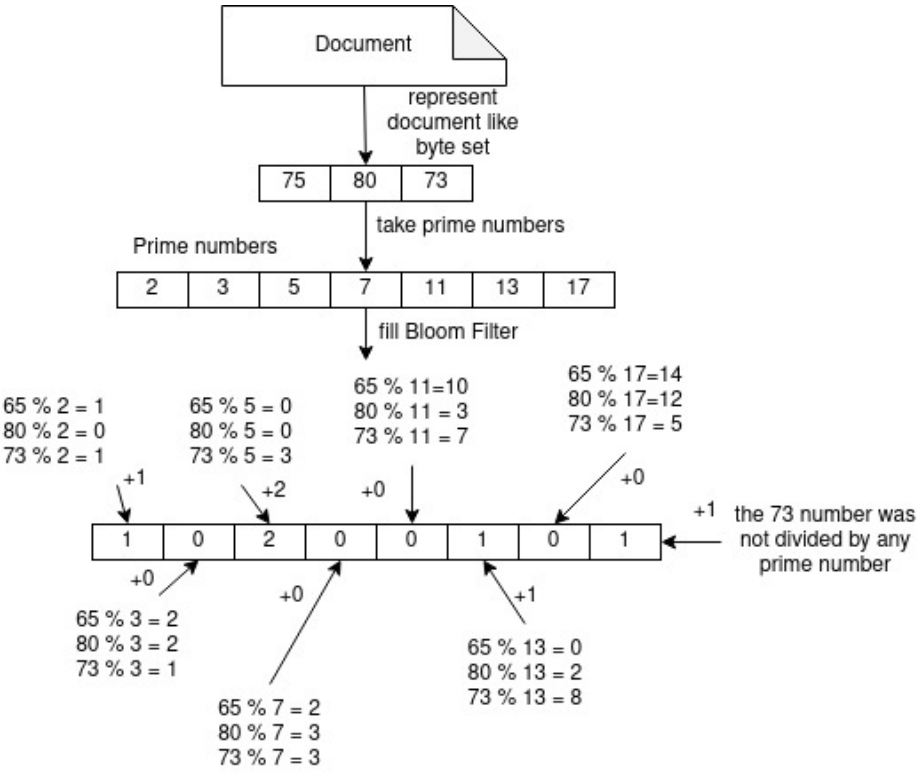

Fig. 1. Prime-based algorithm to produce Spectral Bloom filter

Experiments

The experimental procedure for comparing collision resistance consists of randomly generating input data of a fixed length and a certain step. The amount of data is 100,000. Test data are ASCII characters used to represent an information. The dependent variable in this case is the number of unique Spectral Bloom filters, and the independent variable is the amount of data. MD5, SHA1 and SHA256 were selected as hash functions, which are cryptographic functions and should ensure an even distribution of hash values.

Algorithm of forming a filter using a hash function:

1) present the document in the form of bytes;
2) hash each byte;
3) represent each hash as an unsigned integer;
4) fill the filter using the modulo division operation.

Figure 2 shows a visualization of the results obtained when generating 100,000 random data with a size of 100 to 450 bytes and a step of 50 bytes.

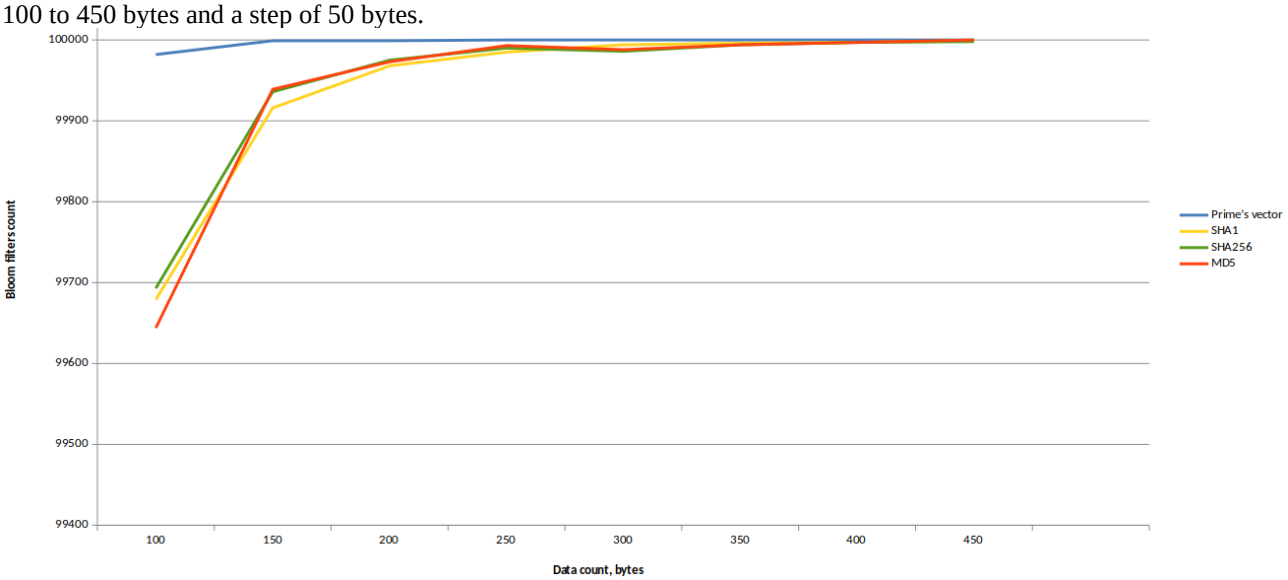

Fig. 2. Unique Spectral Bloom filters from 100,000 random data from 100 to 450 bytes and step 50

Figure 3 shows a visualization of the results obtained when generating 100,000 random data with a size from 5 to 174 bytes and a step of 1 byte.

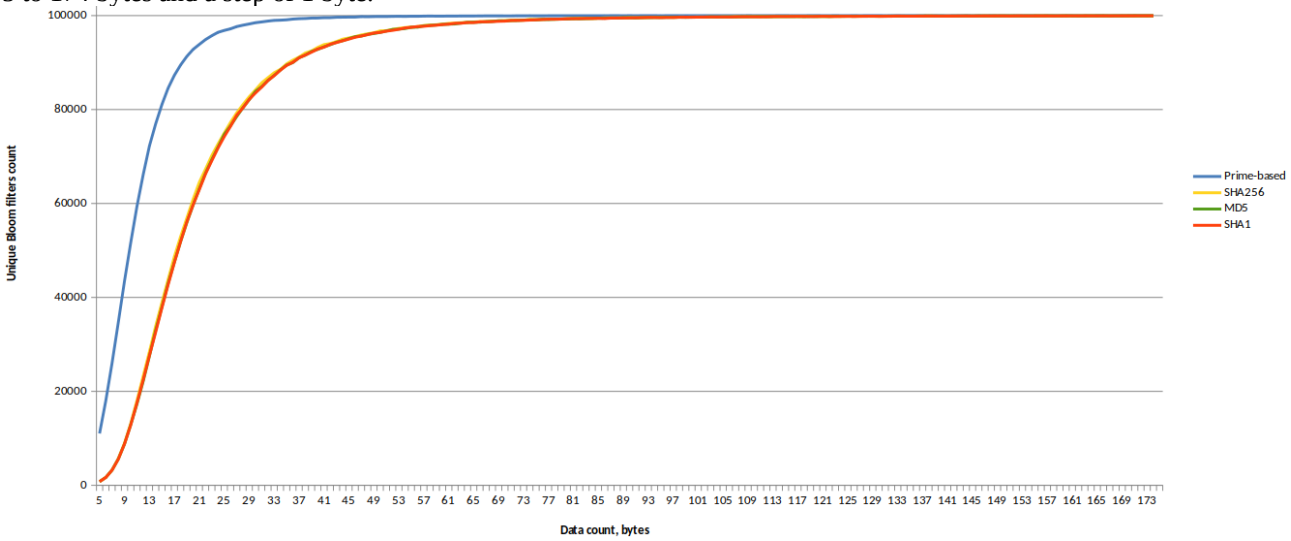

Fig. 3 - Unique Spectral Bloom filters from 100,000 random data from 5 to 174 bytes and step 1

Figures 2 and 3 show that the Prime-based Spectral Bloom filter has higher collision resistance and more quickly approaches the expected value under the given experimental conditions.

To compare the filter calculation speed, the test procedure will be the same as the previous one, but the number of filter calculation iterations for the generated test set is 100. Statistical tools such as arithmetic mean, median, minimum and maximum value will be used to evaluate the methods. The dependent variable in this case is the time required to form the filters.

Figure 4 shows a chart of the arithmetic mean value for data of different lengths.

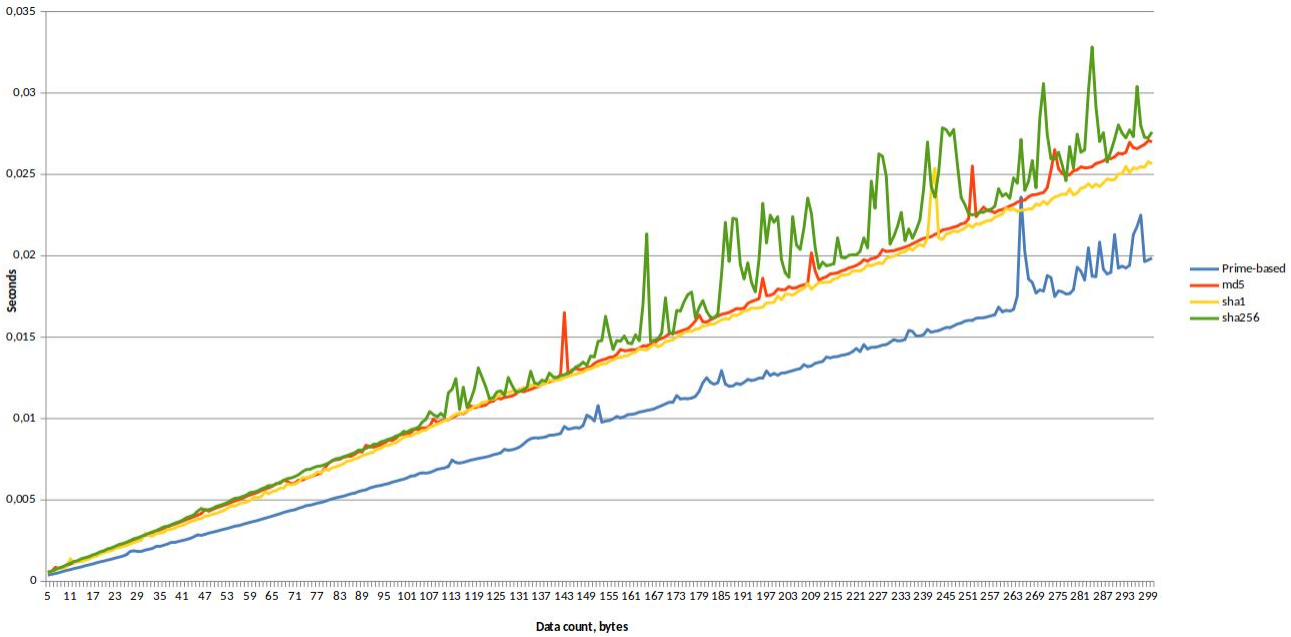

Fig. 4. Mean of elapsed time to get 100 prime-based spectral Bloom filters

Figure 5 shows a chart of the median for data of different lengths.

Fig. 5. Median of elapsed time to get 100 prime-based spectral Bloom filters

Figure 6 shows a chart of the minimum value for data of different lengths.

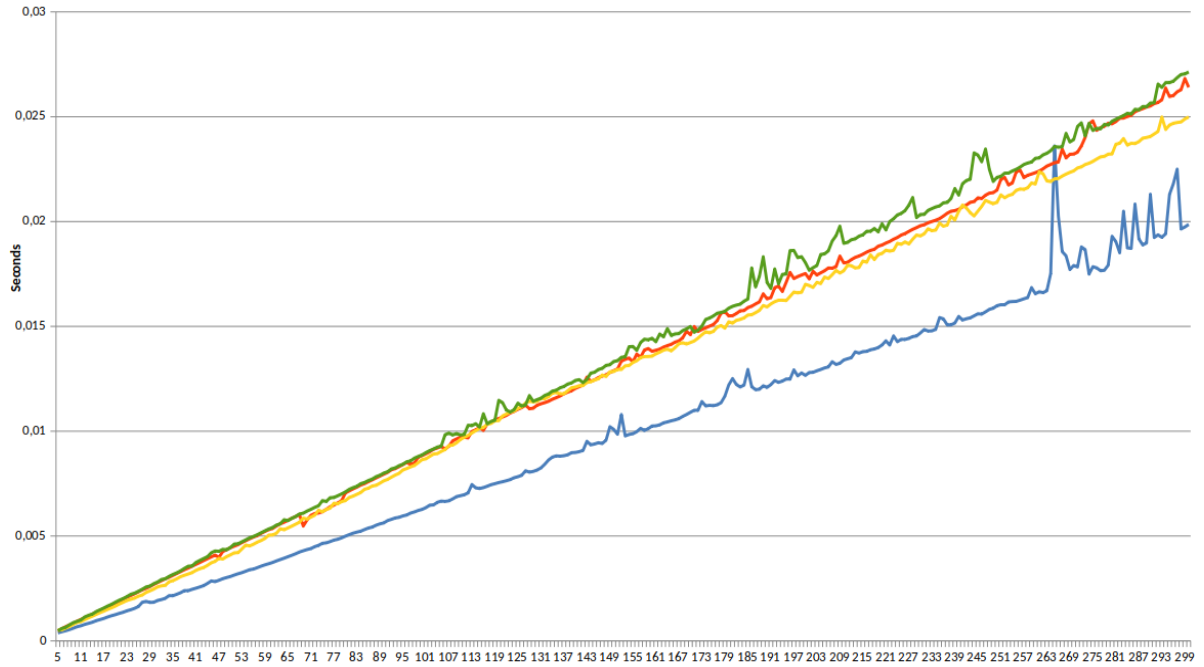

Fig. 6. Min of elapsed time to get 100 prime-based spectral Bloom filters

Figure 7 shows a chart of the maximum value for data of different lengths.

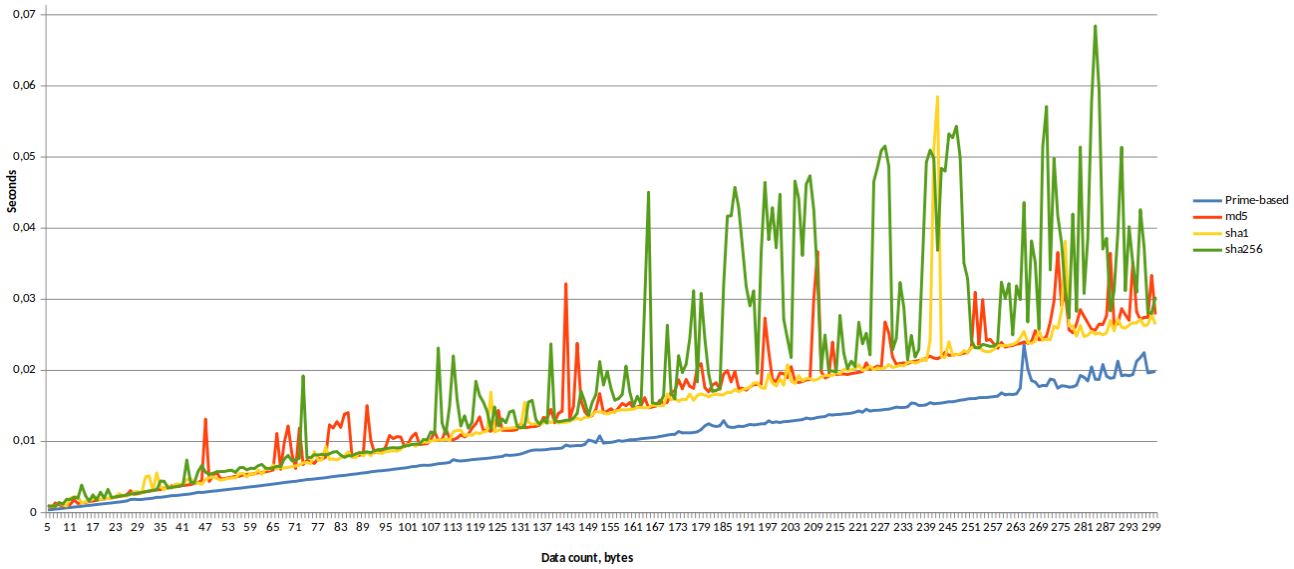

Fig. 7. Max of elapsed time to get 100 prime-based spectral Bloom filters

Figures 4, 5, 6 and 7 show that the time required for the formation of filters is shorter and varies in proportion to the values of the independent variable.

Conclusions

Thus, the proposed method of forming the Spectral Bloom filter has a number of advantages for its use in the data reconciliation process. The small size of the filter allows it to be easily transported between nodes of a distributed database to check the consistency of critical documents, and the time to create it allows for quick creation of the filter from source documents. Also, it is more resistant to collisions compared to forming a filter using hash functions.

The further direction of the work is related to the design of the Active Anti-Entropy mechanism using the proposed method for maintaining the consistency of critical data in distributed document-oriented NoSQL databases and its effectiveness evaluation.

References

1. Nikitin, V. A., & Krylov, E. V. (2023). Using a transactional clock to speed up the process of data reconciliation in distributed systems. Scientific Bulletin of Uzhhorod University. Series "Mathematics and Informatics", 42(1), 188-192. URL: https://doi.org/10.24144/2616-7700.2023.42(1).188-192

2. Ramabaja L. The Distributed Bloom Filter // arXiv. 2019. С.1-17 URL: https://doi.org/10.48550/arXiv.1910.07782

3. Kumar A., Xu J., Wang J. Space-Code Bloom Filter for Efficient Traffic Flow Measurement // IEEE Journal on Selected Areas in Communications. 2006. № 24 (12). С.1-12 URL: https://doi.org/10.1109/JSAC.2006.884032

4. Cohen S., Matias Y. Spectral Bloom Filters // Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data. 2003. С.1-12 URL: http://dx.doi.org/10.1145/872757.872787

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(41), 45–57. URL: https://doi.org/10.20535/1560-8956.41.2022.271338

Valerii Nikitin **Валерій Нікітін**	PhD student, Department of Information Systems and Technologies, Igor Sikorsky Kyiv Polytechnic Institute. e-mail: 19valeranikitin96@gmail.com https://orcid.org/0000-0002-4509-1204	аспірант кафедри інформаційних систем та технологій КПІ ім. Ігоря Сікорського.
Evgen Krylov **Євген Крилов**	PhD in Technical Sciences, Associate Professor, Department of Information Systems and Technologies, Igor Sikorsky Kyiv Polytechnic Institute. e-mail: ekrylov1964@gmail.com https://orcid.org/0000-0003-4313-938X	кандидат технічних наук, доцент кафедри інформаційних систем та технологій КПІ ім. Ігоря Сікорського.