BULLETIN of Cherkasy State Technological University

Journal homepage: https://bulletin-chstu.com.ua/en

Vol. 28 No. 4. 2023

UDC 004.91
DOI: 10.62660/2306-4412.4.2023.10-18

Review of joint text editing algorithms Conflict-free Replicated Data Types (CRDT)

Yurii Rabeshko*
Postgraduate Student
National University of Water Management and Nature Management
33000, 11 Soborna Str., Rivne, Ukraine
https://orcid.org/0009-0002-0763-7138

Yurii Turbal
Doctor of Technical Sciences, Professor
National University of Water Management and Nature Management
33000, 11 Soborna Str., Rivne, Ukraine
https://orcid.org/0000-0002-5727-5334

Abstract. Analysing and selecting algorithms for collaborative text editing, especially implementing conflict-free replicated data types, is critical to understanding how modern systems can achieve real-time collaboration while ensuring data integrity. The purpose of the study is to review various collaborative editing algorithms and conduct a comparative analysis to understand their advantages, disadvantages, and applications. Statistical methods, methods for analysing algorithms and their use in real-world scenarios are used. The study results showed that the use of collaborative text editing algorithms contributes to solving important tasks and challenges in the modern world of information technology. Collaborative editing algorithms are determined to facilitate real-time communication and information exchange. This is especially important in the context of remote work and communication, which have become the standard for many organisations. Collaborative text editing is widely used in various fields and industries where teamwork, real-time collaboration, and document sharing are essential, such as scientific research, education, software development, book and manuscript editing, legal cooperation, contract draughting, medical reports, etc. In addition, using optimised collaborative editing algorithms helps reduce the time required to process data and create text materials. Collaborative editing algorithms have a wide range of applications in research, business, and education. They allow teams and individual users to solve problems more efficiently and work together on projects. The ability to use the results obtained in practical activities will allow using algorithms for joint text editing for further innovation and development of information technologies, which will allow working together and exchanging information with the whole world in real-time

Keywords: distributed systems; key operations; Logoot; Astrong; Logarithmic Sequence; Replicated Growable Array

Article's History: Received: 04.09.2023; Revised: 10.11.2023; Accepted: 18.12.2023.

INTRODUCTION

The relevance of this study lies in the development of conflict-free replicated data type algorithms (CRDT), which are becoming an essential part of digital life and have a wide range of applications in various industries. The use of CRDT has become an important tool for editing text and large amounts of information, and working together on a document. In addition, the use of collaborative text editing algorithms can help increase efficiency and productivity. Due to CRDT, users can work together on texts in real-time, make joint

Suggested Citation:
Rabeshko, Yu., & Turbal, Yu. (2023). Review of joint text editing algorithms Conflict-free Replicated Data Types (CRDT). *Bulletin of Cherkasy State Technological University,* 28(4), 10-18. doi: 10.62660/2306-4412.4.2023.10-18.

*Corresponding author

edits and changes. This is especially important in teams and groups of people working on joint projects or documents from a distance. Research on collaborative text editing algorithms is important because it helps understand how these algorithms work, their advantages, and limitations. This opens up the possibility of developing new and improved collaborative editing techniques that help improve the efficiency and productivity of working with texts. Such investigations can also have practical applications in various fields, including education, scientific, business, etc.

The main problem of the study is the preservation of consistency in distributed systems, as they are becoming more common in the modern digital world, and the emergence of conflict situations when trying to update data simultaneously. This problem is particularly relevant in modern digital world, where working together on documents and data in real-time is becoming the standard. Investigating collaborative text editing algorithms allows understanding how these algorithms help solve this problem and what aspects of distributed systems and conflicting text editing affect their consistency and efficiency. The study results can have important practical applications for developing systems that provide reliable and conflict-free collaborative text editing, which is becoming an increasingly important aspect in the modern world of information technology.

According to N. Saquib *et al.* (2022), CRDT (Conflict-free Replicated Data Types) are abstract data types that provide a principled approach to asynchronous data mismatch matching. The researchers noted that this is very important for modern distributed systems, where data access must be reliable and fast, even in possible data discrepancies. The paper does not provide detailed instructions on the practical implementation of algorithms. A study on cloud services that support real-time document collaboration was conducted by O. Medvedovska and V. Yatsenko (2021). They examined the functionality of individual applications, such as Google Docs and Google Drive, that use CRDT structures. The results obtained proved that the use of a tool that supports joint work on the document showed labour efficiency. The study does not consider collaborative text editing algorithms as an object of research, but the effectiveness of using these algorithms can be seen in the example of the Google Docs cloud environment since collaboration on a document is conducted without conflicts.

F. Hu and R.H. Trivedi (2020) used the RGA (Replicated Growable Array) algorithm to position a hotel brand and map the competitive landscape using text-based content analysis. This approach demonstrates the importance of collaborative text editing algorithms in various areas, including analysing large amounts of text information for making business decisions and improving brand strategies. The use of algorithms such as RGA helps in effective data management and analysis and in solving tasks related to text editing and collaboration on documents and information. However, the study does not include a comparative analysis of the RGA algorithm with other text analysis methods.

According to M. Kleppmann *et al.* (2021), replicated tree data structures are the fundamental building block of distributed file systems. Their paper introduced the LSEQ (Logarithmic Sequence) algorithm, which handles arbitrary simultaneous modifications on trees while ensuring that the tree structure remains valid. However, this paper does not provide detailed guidance on the practical implementation of the LSEQ algorithm. O. Shchetynina *et al.* (2022) reviewed the Trello multi-platform project management system. In the study, an experiment was conducted that proved the effectiveness of using Trello. The paper did not consider CRDT as an object of research, but an experiment and the use of the Trello platform demonstrated the effectiveness of using these algorithms. From a study by H.H. Kirychek and A.I. Chubich (2020), this platform uses a project management paradigm known as kanban. This paradigm uses methods that prevent overloading team members.

According to D. Brahneborg *et al.* (2022), most CRDTs are based on some variant of atomic translation, as this allows them to maintain causal relationships between updates of multiple objects. However, the overhead of this atomic translation is unnecessary in systems that process only independent CRDT objects. The researchers identified a set of use cases for tracking resource usage, where there is a need for a replication mechanism with less complexity and network usage compared to using atomic translation. The study does not include a comparative analysis of CRDT with other data replication methods, including traditional approaches.

The purpose of the study is to review and compare CRDTs, such as Logoot, RGA, LSEQ, Astrong, and understand the advantages and disadvantages of each of the algorithms. The following tasks were formed in the examination of joint text editing algorithms to achieve this goal: detailed analysis of joint text editing algorithms, in particular, Logoot, RGA, LSEQ, and Astrong, with the definition of their basic principles and functionality; identifying the advantages and disadvantages of each of these algorithms to evaluate their effectiveness and suitability for specific use cases; considering the problems of conflicts and consistency that may arise when using these algorithms and suggesting possible strategies for solving them. This study can help to understand the algorithms used in text editing and create new approaches to collaborative text editing.

MATERIALS AND METHODS

By analysing the papers of researchers such as: B. Nédelec *et al.* (2013), M. Nicolas *et al.* (2020) and F. Jacob *et al.* (2021), data containing information about the performance and efficiency of various algorithms and their functioning under different usage conditions were selected. This analysis allowed obtaining

objective information on how different text editing algorithms work in real-world scenarios. Using this, the advantages and disadvantages of each of the algorithms under study, their performance, and their interaction with other parts of the system were established. This approach collecting objective data that plays an important role in making informed decisions when choosing and configuring algorithms for specific applications.

In addition, statistical methods were used for a detailed examination of joint text editing algorithms. Using these methods allowed conducting a quantitative analysis of data to determine the statistical importance of various parameters and factors affecting the performance of algorithms. Statistical methods also helped in the comparative analysis of various algorithms, identifying their advantages and disadvantages in terms of statistical indicators. This approach allowed obtaining objective data on the effectiveness and performance of various CRDTs, which is important for making informed decisions when choosing a specific algorithm for a specific application. This analysis also allowed identifying opportunities for improving and optimising existing algorithms to improve their efficiency.

In the course of the study, the method of analysing algorithms was used. This technique involved a detailed review and analysis of the operation of specific algorithms for joint text editing and their theoretical foundations. The analysis of algorithms helped to reveal the specifics and features of each algorithm, understand exactly how they solve the problem of joint text editing and what aspects are important for their effective work. In addition, using the analysis, it was possible to determine exactly how each algorithm solves the problem of joint text editing. The analysis method thoroughly investigated the mechanisms underlying various CRDTs and allowed understanding how they handle and synchronise text editing between users.

Algorithms such as Logoot, RGA, LSEQ, and Astrong were also considered. These algorithms were chosen for examination and comparison because of their relevance and importance in the field of collaborative text editing. The selected algorithms are representative examples of different approaches to solving the problem of joint text editing, and each of them has its own unique features and advantages. A comparative analysis of these algorithms allowed identifying which specific algorithms were more effective under certain conditions. A detailed analysis of the data identified the strengths and weaknesses of each algorithm, their speed and stability under different conditions of use. Implementation methods were considered for some of the above algorithms. This enabled a greater understanding of how these algorithms can be implemented in practise. In addition, the implementation of algorithms allowed clearly understanding the functioning of the algorithm and determining how they interact with real data and users in specific conditions.

RESULTS

Detailed overview of CRDT collaborative text editing algorithms and their functions. For a better understanding of collaborative text editing algorithms, it is necessary to get familiarised with the theoretical aspects and usage examples. Notably, an important aspect in examining of CRDT is the understanding of conflict-proof replicated data structures that underlie many collaborative text editing algorithms. In this section, the algorithms that are widely used in practise are discussed. Logoot is a CRDT algorithm for supporting collaborative editing on distributed systems. It was presented by S. Weiss *et al.* (2009). Logoot was designed to provide better performance than previous CRDT algorithms such as Treedoc and WOOT (WithOut Operational Transforms). Logoot is based on the idea of encoding each element in a data structure using a sequence of integers, enabling the efficient insertion and deletion operations.

The Logoot algorithm presented an editable document as a sequence of lines, where each line is a sequence of characters. Each character was represented as a tuple (i, c), where i – the unique identifier for that character and c – its value. Logoot gave each character a unique identifier, encoding its position in the document using a sequence of integers. Logoot used a tree-based data structure, which allowed efficient character insertion and deletion operations and fast search and retrieval. Figure 1 shows part of the Logoot algorithm implementation code.

```
import Logoot exposing (empty, insert, toList)
pid = ([(2, 3)], 0)
logoot1 = empty
    |> insert pid "hey!"
    |> remove pid "hey!"
    |> toList
logoot2 = empty
    |> remove pid "hey!"
    |> insert pid "hey!"
    |> toList
result = logoot1 == logoot2 -- True
```

Figure 1. Implementation of part of the Logoot algorithm in the Elm programming language
Source: GitHub (2016)

The presented code demonstrated the "commutativity" property of the Logoot algorithm when inserting and deleting characters. Notably, Logoot ensures that character insertion and deletion operations are commuted, meaning that the order in which they are performed does not affect the final result. Logoot supports three key operations: insert, delete, and merge. Logoot first generates a unique identifier for that character based on its position in the document to insert a new character into a document. The ID consisted of a sequence of integers that indicated a place in the document

structure. The algorithm then found a suitable place to insert a new character in the document, considering the order of identifiers. This allowed inserting characters in the correct order and ensuring that their IDs are unique. After that, the document structure was updated accordingly. Logoot simply marks the character as deleted, setting its value to null to remove a character from a document. The deletion operation does not remove the character from the document but marks it as deleted, which allows saving the document's editing history and resolving contradictions between different versions. Merging was used to resolve contradictions between different versions of a document that were edited simultaneously by different users. Logoot has integrated two versions of the document, applying changes from each version in the order they are executed. This maintains the consistency of the document and avoids conflicts.

The algorithm showed good performance in terms of scalability and memory usage. It can process documents of any size, and its memory usage increases logarithmically with the document size. Logoot also proved to be faster than previous CRDT algorithms, such as Treedoc and WOOT, especially for large documents. In one study, Logoot was compared with other CRDT algorithms for collective real-time editing. The results showed that Logoot has the best performance in terms of speed with low latency and high throughput while maintaining consistency and correctness (Karayel & Gonzàlez, 2022). RGA is a CRDT algorithm for supporting list-like data structures in distributed systems. It was introduced by Hyun-Gul Roh, Myeongjae Jeon, Jin-Soo Kim, and Joonwon Lee in 2011. RGA was designed to address the limitations of previous CRDT algorithms for list-like data structures such as WOOT and Treedoc. These algorithms were inefficient for large lists or for many simultaneous edits. RGA is based on a sequence of positions and characters, similar to Logoot, but uses the latest method for processing simultaneous edits (Roh *et al.,* 2011).

The algorithm represented a list as a sequence of positions and symbols. Each position represented the location of a character in the list, and each character had a unique identifier and value. Unlike Logoot and LSEQ, RGA did not use a hierarchical data structure but presented the list as a sequence of nodes connected by pointers. Figure 2 shows part of the implementation code for this algorithm. The above code is a representation of the sequence of operations for editing a document in joint text editing. RGA supports two main operations: insert and delete. Each character in the document has its own unique identifier, which is generated based on its value and position in the document. This allowed accurately defining each character and avoiding conflicts when inserting at the same time. RGA generates a unique identifier based on the previous and next characters in the sequence and assigned the character a position between them to insert a new character. This allowed inserting characters at any point in the sequence. RGA marked it as deleted, setting its value to null to delete the character. Deleted characters are not removed from the data structure, but they are marked as deleted and excluded from subsequent operations. If there are conflicts where two or more users delete or insert characters in the same place, the RGA uses the "bookmarks" technique (special markers that indicate that the character has been deleted). They are distributed to other nodes in the system to ensure consistency.

```
{
  agent: 'abc',
    seq: 10,
    ops: [{
      type: 'insert', // <-- This insert is ('abc', 10)
      content: 'x',
    }, {
      type: 'delete', // <-- deletes don't bump the version
    }, {
      type: 'insert', // <-- This insert is ('abc', 11)
      content: 'y',
  }]
}
```

Figure 2. Implementation of part of the RGA algorithm in the Rust programming language
Source: GitHub (2021)

The RGA algorithm showed good results in terms of scalability and performance. It can handle large lists, and a large number of simultaneous edits, and memory usage increases linearly with the size of the list. RGA also performed well in real-world applications, such as collective text editing and chat applications. V.B. Gomes *et al.* (2017) compared RGA with other CRDT algorithms for list-like data structures in a collective text editor. The results showed that RGA had better performance than other algorithms, with lower latency and higher throughput. RGA also had better convergence properties, which means it handled conflicts better and provided consistency when editing simultaneously. LSEQ is a CRDT algorithm for maintaining list data structures in distributed systems. It was developed by B. Nédelec *et al.* (2013). LSEQ was designed to improve the scalability and performance of previous CRDT algorithms for list data structures such as Treedoc and RGA. These algorithms had limitations in terms of scalability and efficiency, especially when working with large lists or high levels of simultaneous editing. LSEQ is based on a sequence of integers, similar to Logoot, but uses a more efficient encoding scheme.

The algorithm represented a list as a sequence of integers, where each number represents the position of a character in the list. The encoding scheme used by LSEQ enabled the efficient insertion and deletion of characters and quick searching for and bypassing them. LSEQ uses the binary search tree data structure to store integers, which allows efficiently combining different versions of the list. Each character in the document is represented by an integer corresponding to its position in the document. These numbers are encoded to

be unique and allow efficient insertion and deletion of characters. One of the critical features of LSEQ is the ability to effectively combine different versions of the list that occur when users edit simultaneously. This is done using a binary search tree and a special join algorithm. The main goal of LSEQ is to ensure efficient insertion and removal of characters in a text document and avoid conflicts between users who are editing the document at the same time.

LSEQ supports two key operations: insert and delete. A unique integer is generated for to insert a new character into the LSEQ list. This integer determines the position of the character in the document. Unique numbers are generated in such a way that they are unique and ensure that the characters are sorted. A character is removed from the LSEQ list by marking the character as deleted and setting its value to null. Deleted characters remain in the tree, but they are not counted in further editing operations. When conflicts occur during simultaneous edits, LSEQ uses a technique known as "path copying" to resolve these conflicts. This means creating a new subtree that represents the version of the list with the changes made. The path is copied from the root of the binary search tree to the position of the edited character.

The LSEQ algorithm showed good results in terms of scalability, performance, and memory usage. It is capable of handling very large lists and high levels of simultaneous edits, with memory usage increasing logarithmically as the list size increases. LSEQ also has good convergence properties, maintaining consistency and correctness even under simultaneous editing conditions. In the study of by M. Kleppmann *et al.* (2021), LSEQ was compared with other CRDT algorithms for list-like data structures in distributed text editors. The results showed that LSEQ has the best performance in terms of low-latency and high-throughput speeds, while maintaining consistency and correctness. LSEQ was identified to be more efficient than previous CRDT algorithms such as Treedoc and RGA, especially for large lists.

Astrong is a CRDT algorithm for supporting set-like data structures in distributed systems. It was presented by H. Attiya *et al.* (2016). Astrong is a CRDT algorithm designed to perform efficient and scalable operations on large sequences of ordered elements. It was proposed as a variant of the LSEQ algorithm to improve its scalability and performance characteristics. Astrong was specifically designed for use cases where the sequence size can be very large, such as in text editors or document management systems. The algorithm uses a multi-level tree structure to represent an ordered sequence of elements. At each level of the tree, the sequence is divided into several segments, each of which is represented by a unique identifier. These segment IDs are used to efficiently localise and manipulate individual elements in a sequence. Astrong uses a combination of linear and logarithmic data structures to represent segments and corresponding identifiers, allowing efficient indexing and manipulation of large sequences.

It provides efficient and scalable operations for adding and removing elements from a sequence. When an element is added to a sequence, it is assigned a unique identifier based on its position in the sequence. This ID specifies exactly where to insert an element into the data structure. Because the main data structure in Astrong is a multi-level tree, and due to the unique identifier, insertion was performed efficiently and quickly. This ID is then used to find the corresponding segment in a multi-level tree, where the element can be inserted without the need for any additional restructuring or restructuring of the data structure. Similarly, when an element is removed from a sequence, it is simply marked as deleted, without the need to actually delete data from the main data structure. This allows performing deletion operations without substantially affecting performance and rebuilding the structure. The Astrong algorithm was evaluated in various contexts and use cases, shown to provide efficient and scalable operations on large sequences of ordered elements. Compared to other CRDT algorithms, Astrong showed improved performance characteristics, especially when working with very large sequences or a large number of concurrent users. Astrong's data structure and implementation were carefully designed to balance the trade-off between space complexity, time complexity, and scalability, allowing efficient handling of a wide range of use cases.

Comparison of CRDT algorithms for collaborative text editing in distributed systems. Logoot and RGA use different data structures to represent an ordered sequence. Logoot uses a tree data structure, while RGA uses a sequence of nodes connected by pointers. LSEQ and Astrong use hierarchical trees to represent the sequence, but they differ in implementation details. The different data structures used by these algorithms have different trade-offs between space-time complexity and scalability (Karayel & Gonzàlez, 2022). All CRDT algorithms provide some level of support for concurrent editing of ordered sequences but differ in resolving conflicts that occur when multiple users edit the same part of the sequence simultaneously. Logoot and RGA use the "last editor wins" strategy to resolve conflicts, while LSEQ and Astrong use a more sophisticated conflict resolution engine that considers the unique identifiers assigned to each element in the sequence. This mechanism allows LSEQ and Astrong to resolve conflicts more efficiently than Logoot and RGA.

The scalability and performance of these CRDT algorithms depend on a variety of factors, including sequence size, number of users, and frequency of simultaneous edits. Logoot and RGA are known to have scalability and performance limitations, especially when working with large sequences or a large number of users. LSEQ and Astrong were designed to address some of these limitations and demonstrated good performance in various scenarios (Kleppmann *et al.*, 2019). Each of these CRDT algorithms has its own unique

advantages and disadvantages, which makes them more suitable for certain use cases. Logoot and RGA are suitable for applications that require low space complexity and simple conflict resolution mechanisms but may have scalability and performance issues in more complex scenarios. LSEQ is suitable for applications that require efficient indexing and more advanced conflict resolution mechanisms but may have a greater time complexity. Astrong is designed for use cases that require efficient processing of very large sequences but can have greater space complexity.

The main results showed the advantages and disadvantages of each of the algorithms and demonstrated the importance of collaborative text editing algorithms. For example, Logoot is characterised by good performance of add and remove operations but it can cause more conflicts when making simultaneous changes. On the other hand, LSEQ provides a low probability of conflicts but may be less efficient when inserting new characters. The study showed that collaborative text editing algorithms are of great importance for distributed systems. They allow users to work together on documents regardless of their physical location and network availability. This is especially important for teamwork and collaborative editing of real-time text documents. The study also focused on evaluating the performance and performance of each algorithm under different usage conditions. This helps to determine which algorithm is best suited for specific tasks and use cases. It is important to understand that different algorithms may be optimal for different application cases. During the study, it was determined that some algorithms may be more difficult to implement than others. This may affect the speed of development and support of the system. In addition, when multiple users work on the same document simultaneously, conflicts may occur. Research on collaborative editing algorithms helps determine which algorithms better resolve these conflicts and ensure that changes are properly integrated.

Thus, collaborative text editing algorithms are important components for modern distributed systems and collaborative text document editing. They help solve problems related to teamwork on real-time documents and ensure simultaneous availability and consistency of data in the face of network separation and other failures. The main task of the study was also to identify how each algorithm affects the performance and efficiency of working with distributed text documents. The results demonstrate how each algorithm affects the performance and efficiency of working with distributed text documents. This information helps developers choose the best algorithm for specific tasks and use cases. In general, collaborative text editing algorithms are important components for building applications and systems that allow users to effectively collaborate on real-time text documents. Their research and improvement are of great importance for the further development of distributed systems and tools for collaborative text editing.

DISCUSSION

In their paper on the examination of various routing protocols, N. Alsulami *et al.* (2022) used the Logoot algorithm to develop algorithms for data replication and document synchronisation in distributed systems for collective editing. The analysis of the study revealed new opportunities and advantages of using the Logoot algorithm outside of its general application in collective editing. This may encourage other researchers to develop and expand this algorithm in other areas of information technology. The use of data replication algorithms in network protocols can help ensure data consistency and synchronisation between nodes, which is an important aspect in distributed systems. The similarity of the paper lies in the analysis of distributed data replication methods in the context of distributed systems. In addition, both studies address the problem of data synchronisation between replicas or nodes in a distributed system. However, the paper examines new features and benefits of using the Logoot algorithm beyond collective editing.

W. Cai *et al.* (2022) investigated accelerating consistency support algorithms based on a commutative replicated data type using multi-core processors, examined the problem using the example of the RGA algorithm and parallel to RGA. Experimental results showed that when processing millions of simultaneous updates, PRGA achieves a performance increase of about four times compared to RGA on a conventional quad-core processor. Paper analysis can make important contributions to the scientific and practical fields of distributed systems and processing simultaneous data updates. Determining the increase in performance when using the PRGA algorithm compared to RGA on a multi-core processor is an important indicator. This can help developers of distributed systems improve the performance and speed of processing simultaneous data updates in such systems. This study can serve as an example of the efficient use of multi-core processors to speed up data processing in distributed systems. It can be useful for other researchers and developers working on similar tasks. The results obtained can be important for understanding and developing the theoretical foundations of CRDT and computational theory. They can help extend and improve existing approaches to handling concurrent updates. Although both papers focus on similar aspects, they have different research objects: the first paper focuses more on CRDT algorithms for collaborative text editing, while the second paper focuses on speeding up algorithms on multi-core processors using commutative replicated data.

M. Nicolas *et al.* (2020) devoted their study to the issue of effective renaming in the context of CRDTs for sequences and considered the issue of effective

renaming of objects or text parts in CRDTs sequences based on the LSEQ algorithm. Renaming is an important operation that allows correctly handling insertion, deletion, and editing operations in such sequences. Researchers offer effective renaming algorithms to ensure the correctness and speed of performing operations. The analysis of the study will improve CRDTs algorithms for sequences. Effective renaming is an important operation that affects the performance and correctness of data insertion, deletion, and editing operations in replicated sequences. The results of the study can help developers of CRDT structures expand their capabilities and applications. Efficient renaming algorithms make CRDTs more suitable for a variety of use cases, including collective editing of text data. Both studies aim to improve CRDTs to ensure correct and efficient operation in distributed systems, and their results are essential for developers and researchers working on distributed systems and collective data editing.

I. David and E. Syriani (2022) dedicated their paper to the implementation of distributed, collaborative modelling at different levels (multi-level modelling) using CRDT, which means data types that provide conflict-safe operations for distributed systems, discussed how using CRDT helps resolve conflicts that arise when editing a document in real-time, ensuring consistency and integration of changes made by users at different levels of modelling. CRDTs performing operations on distributed data without the need for manual conflict resolution. Applying CRDTs to implement distributed collaborative modelling can be useful in many areas, including software development, systems analysis, and modelling, and in any field where it is necessary to work together on complex models and data in a distributed environment. The analysis of the study will help to improve methods and approaches for implementing distributed collaborative modelling at different levels. Using CRDT to work with data allows resolving conflicts and ensuring real-time data consistency. The results of the study can be useful in software development, systems analysis, and modelling, in any field where collaborative modelling and collaboration on data in a distributed environment is important. CRDTs can be used in collective editing, work on models, document management, and other areas. The examination of CRDTs and their application in the implementation of distributed collaborative modelling can make important contributions to replicated data theory and the field of distributed systems. Both papers address the importance of using CRDTs in distributed systems to resolve conflicts and ensure data consistency. However, the paper highlights the use of CRDTs to implement collaborative modelling at different levels.

F. Jacob *et al.* (2021) discussed the use of CRDTs in Byzantine systems that may face attacks and abuse. The Byzantine model allows considering cases when some nodes of the system may act maliciously or incorrectly (equivocation). Their examination aims to explore how CRDTs can detect and solve problems of equivocation occurrence in the Byzantine environment, ensuring correct and reliable data synchronisation in the face of attacks and abuses. The use of CRDTs in Byzantine systems can provide increased resistance to abuse and attacks. Performance analysis can reveal how CRDTs can detect and solve equalisation problems in such systems, ensuring correct data synchronisation even when attackers are active. The work can contribute to the development of methods and approaches for data protection and ensuring their conflict-proof replication in Byzantine systems. This can be useful for system developers operating in vulnerable or critical environments. The study by F. Jacob *et al.* (2021) may contribute to increased awareness of security in distributed systems and increased attention to security issues in the Byzantine environment. The similarity of the papers lies in the study of CRDT algorithms, and the fact that they indicate the practical importance of using CRDTs in real-world conditions. However, the study focuses on using CRDTs to improve the resilience and security of data in Byzantine systems that may face attacks and abuse.

Thus, examining CRDT algorithms requires a deep understanding of their theory and applications in various distributed system scenarios. Each of the CRDT algorithms presented in this paper has its own unique advantages and disadvantages, and the correct choice will depend on the specific requirements of the application. By carefully evaluating the trade-offs of various algorithms, developers can choose the right CRDT algorithm to maintain consistency and availability in their distributed systems.

CONCLUSIONS

Research in the field of collaborative text editing algorithms shows that CRDT structures are becoming increasingly popular in the context of building distributed systems. This approach allows creating systems that ensure simultaneous consistency and availability of data, even in the event of network splits or other failures. Collaborative text editing algorithms such as Logoot, RGA, LSEQ, and Astrong use CRDT structures to achieve this goal. The main task of these algorithms is to create a mechanism that allows users to work together on documents and edit them, regardless of their physical location and availability on the network. CRDT structures allow avoiding conflicts and automatically combining changes made by users in different parts of the system. This greatly simplifies working with collaborative text editing, making it more efficient and complete. Each of these algorithms has its advantages and disadvantages, and the correct choice will depend on the specific requirements of the application.

The main problems with using the above algorithms are the complexity of implementation, which may require a deep understanding of the algorithm itself and the specifics of working with distributed systems and networks. The implementation of the algorithm must be efficient and reliable to ensure the quality of service for users. In addition, the problem may be

managing conflicts that may occur when multiple users change simultaneously. This is important to ensure that the data is correct and consistent. Different algorithms have different conflict resolution strategies, and it is vital to determine which approach is best suited for a particular use. Notably, working with CRDT structures may require additional resources, especially in conditions of a large number of users or a large amount of data. This can affect system performance and scalability. The results showed that choosing Logoot and RGA is a good option for supporting collaborative text editing, and RGA has the advantage of better solving concurrent operations. LSEQ is a good choice for supporting scalable and efficient indexing of large data sets. Astrong is a good choice for efficient and scalable processing of large sequences. One of the main advantages of CRDT algorithms is that they can provide high availability and low latency without complex coordination or centralised management mechanisms. However, they have some limitations, especially regarding the types of operations they support and the effectiveness of those operations. Therefore, it is essential to carefully evaluate the trade-offs of different CRDT algorithms when choosing the right one for a particular use case. The improvement and development of new algorithms and methods for joint text editing based on CRDT structures remains a relevant area. Research also can be focused on solving conflict problems in CRDT structures and developing algorithms that can resolve these conflicts more efficiently and automatically synchronise data between different users. Another important area is the development of interfaces and tools that make it easier for users to edit texts together based on CRDT structures. This includes creating user-friendly editors that are intuitive and easy to use and developing mechanisms for syncing data between devices and platforms. In addition, analysing use cases and identifying optimal approaches to implementing CRDT-based collaborative text editing systems in various fields, such as education, business, research, and many others, is relevant.

ACKNOWLEDGEMENTS

None.

CONFLICT OF INTEREST

None.

REFERENCES

[1] Alsulami, N., Cherif, A., & Imine, A. (2022). Collaborative editing over opportunistic networks. *International Journal of Ad Hoc and Ubiquitous Computing*, 39(3), 141-156. doi: 10.1504/IJAHUC.2022.121121.

[2] Attiya, H., Burckhardt, S., Gotsman, A., Morrison, A., Yang, H., & Zawirski, M. (2016). Specification and complexity of collaborative text editing. In *Proceedings of the 2016 ACM symposium on principles of distributed computing* (pp. 259-268). New York: Association for Computing Machinery. doi: 10.1145/2933057.2933090.

[3] Brahneborg, D., Afzal, W., & Mubeen, S. (2022). Resilient conflict-free replicated data types without atomic broadcast. In *Proceedings of the 17th international conference on software technologies* (pp. 516-523). Lisbon: SciTePress. doi: 10.5220/0011314500003266.

[4] Cai, W., He, F., & Lv, X. (2022). Multi-core accelerated CRDT for large-scale and dynamic collaboration. *The Journal of Supercomputing*, 78, 10799-10828. doi: 10.1007/s11227-022-04308-7.

[5] David, I., & Syriani, E. (2022). Real-time collaborative multi-level modeling by conflict-free replicated data types. *Software and Systems Modeling*, 22, 1131-1150. doi: 10.1007/s10270-022-01054-5.

[6] GitHub. (2016). Retrieved from https://github.com/hugooliveirad/elm-logoot.

[7] GitHub. (2021). Retrieved from https://github.com/josephg/simple-crdt-text.

[8] Gomes, V.B., Kleppmann, M., Mulligan, D.P., & Beresford, A.R. (2017). Verifying strong eventual consistency in distributed systems. *Proceedings of the ACM on Programming Languages*, 1, article number 109. doi: 10.1145/3133933.

[9] Hu, F., & Trivedi, R.H. (2020). Mapping hotel brand positioning and competitive landscapes by text-mining user-generated content. *International Journal of Hospitality Management*, 84, article number 102317. doi: 10.1016/j.ijhm.2019.102317.

[10] Jacob, F., Bayreuther, S., & Hartenstein, H. (2021). On conflict-free replicated data types and equivocation in byzantine setups. *arXiv - CS - Distributed, Parallel, and Cluster Computing*, 1. doi: 10.48550/arXiv.2109.10554.

[11] Kyrychek, H.H., & Chubich, A.I. (2020). Kanban-method use for software development organization. *Scientific notes of Taurida National V.I. Vernadsky University. Series: Technical Sciences*, 31(6), 78-82. doi: 10.32838/TNU-2663-5941/2020.6-1/13.

[12] Kleppmann, M., Mulligan, D.P., Gomes, V.B.F., & Beresford, A.R. (2021). A highly-available move operation for replicated trees. *IEEE Transactions on Parallel and Distributed Systems*, 33(7), 1711-1724. doi: 10.1109/TPDS.2021.3118603.

[13] Kleppmann, M., Wiggins, A., van Hardenberg, P., & McGranaghan, M. (2019). Local-first software: You own your data, in spite of the cloud. In *Proceedings of the 2019 ACM sigplan international symposium on new ideas, new paradigms, and reflections on programming and software* (pp. 154-178). New York: Association for Computing Machinery. doi: 10.1145/3359591.3359737.

[14] Medvedovska, O., & Yatsenko, V. (2021). Cloud services for organization of joint work on documents in real time mode. *Bulletin of the Cherkasy Bohdan Khmelnytsky National University. Series "Pedagogical Sciences"*, 1, 112-121. doi: 10.31651/2524-2660-2021-1-112-121.

[15] Nédelec, B., Molli, P., Mostefaoui, A., & Desmontils, E. (2013). LSEQ: An adaptive structure for sequences in distributed collaborative editing. In *Proceedings of the 2013 ACM symposium on document engineering* (pp. 37-46). New York: Association for Computing Machinery. doi: 10.1145/2494266.2494278.
[16] Nicolas, M., Oster, G., & Perrin, O. (2020). Efficient renaming in sequence CRDTs. In *proceedings of the 7th workshop on principles and practice of consistency for distributed data* (pp. 1-8). New York: Association for Computing Machinery. doi: 10.1145/3380787.3393682.
[17] Karayel, E., & Gonzàlez, E. (2022). Strong eventual consistency of the collaborative editing framework WOOT. *Distributed Computing*, 35, 145-164. doi: 10.1007/s00446-021-00414-6.
[18] Roh, H.G., Jeon, M., Kim, J.S., & Lee, J. (2011). Replicated abstract data types: Building blocks for collaborative applications. *Journal of Parallel and Distributed Computing*, 71(3), 354-368. doi: 10.1016/j.jpdc.2010.12.006.
[19] Saquib, N., Krintz, C., & Wolski, R. (2022). Ordering operations for generic replicated data types using version trees. In *Proceedings of the 9th workshop on principles and practice of consistency for distributed data* (pp. 39-46). New York: Association for Computing Machinery. doi: 10.1145/3517209.3524038.
[20] Shchetynina, O., Kravchenko, N., Horbatiuk, L., Alieksieieva, H., & Mezhuyev, V. (2022). Trello as a tool for the development of lifelong learning skills of senior students. *Postmodern Openings*, 13(2), 143-167. doi: 10.18662/po/13.2/447.
[21] Weiss, S., Urso, P., & Molli, P. (2009). Logoot: A scalable optimistic replication algorithm for collaborative editing on P2P networks. In *29th IEEE international conference on distributed computing systems* (pp. 404-412). Montreal: IEEE. doi: 10.1109/ICDCS.2009.75.

Огляд алгоритмів спільного редагування тексту Conflict-free Replicated Data Types (CRDT)

Юрій Олександрович Рабешко
Аспірант
Національний університет водного господарства та природокористування
33000, вул. Соборна, 11, м. Рівне, Україна
https://orcid.org/0009-0002-0763-7138

Юрій Васильович Турбал
Доктор технічних наук, професор
Національний університет водного господарства та природокористування
33000, вул. Соборна, 11, м. Рівне, Україна
https://orcid.org/0000-0002-5727-5334

Анотація. Аналіз та вибір алгоритмів для спільного редагування тексту, особливо впровадження безконфліктних реплікованих типів даних, має вирішальне значення для розуміння того, як сучасні системи можуть досягти співпраці в режимі реального часу, забезпечуючи при цьому цілісність даних. Метою дослідження був огляд різних алгоритмів спільного редагування, а також проведення порівняльного аналізу для розуміння їхніх переваг, недоліків та сфер застосування. Серед використаних методів можна зазначити статистичні методи, методи аналізу алгоритмів та їх використання у реальних сценаріях. Результати дослідження показали, що використання алгоритмів спільного редагування тексту сприяє вирішенню важливих завдань і викликів у сучасному світі інформаційних технологій. Виявлено, що алгоритми спільного редагування сприяють полегшенню комунікації та обміну інформацією в реальному часі. Це особливо важливо в умовах віддаленої роботи та спілкування, які стали стандартом для багатьох організацій. Спільне редагування текстів широко використовується в різних сферах і галузях, де командна робота, співпраця в режимі реального часу та обмін документами є важливими, наприклад: дослідницька діяльність, наукові роботи, освіта, розробка програмного забезпечення, редагування книг та рукописів, юридична співпраця, складання контрактів, медичні звіти, тощо. Крім того, використання оптимізованих алгоритмів спільного редагування допомагає зменшити час, необхідний для обробки даних та створення текстових матеріалів. Алгоритми спільного редагування мають широкий спектр застосувань у наукових дослідженнях, бізнесі та освіті. Вони дозволяють командам та індивідуальним користувачам більше ефективно вирішувати завдання, та спільно працювати над проектами. Можливість використання отриманих результатів в практичній діяльності дозволить використовувати алгоритми спільного редагування тексту для подальших інновацій та розвитку інформаційних технологій, що дасть можливість спільно працювати та обмінюватися інформацією з усім світом в реальному часі

Ключові слова: розподілені системи; ключові операції; Logoot, Astrong; Logarithmic Sequence; Replicated Growable Array