Text editing and real time collaboration

Research plan

Ivan Ovcharov, 4090993

Introduction

Text editing is present in many people's day to day activities, whether it is related to work, being a hobbyist writer or even programming for fun. Furthermore, as time has progressed, a lot of businesses and schools have slowly adapted to using real-time collaboration tools, such as Google docs, Word share, Miro boards, Canva and more.
All of these tools have one thing in common - the ability to freely edit text and create new files and invite people to collaborate real time.

Not a lot of people put much thought into how this is actually achieved in the background. Everything related to this - namely preserving text consistency, the deletion of sections that are being edited at the same time by two people or even overriding saves are concerns that are all dealt with in the background by all of the popular tools.

The first part of this research paper covers a technical section of how text editing is achieved in most tools. The second part is targeted towards exploring real-time collaboration and how text is able to be preserved, even while being edited at the same time by multiple people.

This research is based around a real time collaborative text editing application (Copycloud) built for Fontys university of Applied sciences, software specialisation, semester 6.

Literature Review

RAM - RAM or Random access memory is a type of volatile memory that is lost upon powering off the machine/inconsistent memory refresh. RAM is useful for fast reads and writes, quickly feeding data to the CPU and more.
Data buffer - a small area of memory that resides within a computer's RAM. It serves the purpose of holding data before releasing it either within an internal program or outputting it externally.
Data structure - data structures usually refer to storing and ordering data in a specific way in memory. Data structures vary by different use cases and each has its own benefits/pain points.
Websocket - A bi-directional communication protocol between a server and a client. This allows both the client and the server to send each other data in the form of messages.
Gap buffer - A data structure commonly used in text editors for efficient and consistent editing of text and preserving changes. Gap buffers achieve this by keeping a gap before and after characters, allowing new character insertion before and after a character, as well as the deletion of one.
Array - An array is a data structure that stores a fixed-size sequence of elements of the same type. It allows efficient access to individual elements based on their index position within the array.
Dynamic array - A dynamic array is a resizable array that can grow or shrink in size during runtime. It automatically handles the memory allocation and reallocation, allowing for flexible storage of elements and efficient memory usage.

Research questions

The scope of this research involves text editing and the technical background of how that is achieved within a computer. Secondly, the importance of real-time text collaboration tools, as well as the details of how such tools are created and what technology is used to provide the best user experience.

This research paper is conducted on-top of one main question, as well as two provisional sub-questions.

MAIN QUESTION: Text editing and collaboration - how is it achieved?
Summary: The main question of the research aims to provide a deeper look into what happens on a technical level in almost any existing text editing application. Furthermore, a higher overview of how data-consistent collaboration is achieved within tools such as Google Docs, Canva, HackMD and more. This question also keeps CaptionAI and its text editing and collaboration capabilities. This question can be broken down into two sub-questions that will ultimately lead to the conclusion.

SUB QUESTION 1: How is consistent text data achieved within text editing applications?
Summary: This sub-question delves into the technical aspects of ensuring consistency in text data within text editing applications. It investigates the various techniques and data structures employed to maintain data integrity and consistency.

SUB QUESTION 2: How are text collaboration tools created and what is their importance?
Summary: This sub-question focuses on the creation and significance of text collaboration tools. It examines the development process, methodologies, and technologies used in building real-time collaboration features within text editing applications. By studying popular collaboration tools like Google Docs, Canva, HackMD, and others, this research seeks to uncover the underlying architectures, communication protocols, and data delivery methods employed to facilitate collaborative editing. Furthermore, it highlights the importance of such tools in enhancing productivity, enabling remote collaboration, and fostering effective teamwork. Understanding the creation and importance of text collaboration tools is crucial for developing efficient and user-friendly collaborative editing environments.

Research methodology

This research paper is based on the DOT framework methodology for research and answering questions. This chapter serves as a breakdown of what methodologies are used per the above shown questions. Furthermore,

For the main question, "Text editing and collaboration - how is it achieved?", a multi-method approach is utilised to gain comprehensive insights. Firstly, a systematic literature review is conducted to gather existing knowledge and understand the foundational concepts of text editing and real-time collaboration. This review involves analysing academic papers, research articles, books, and relevant online resources. It helps establish a theoretical foundation and identify key areas of investigation.

To supplement the literature review, a case study methodology is employed to examine popular text editing applications such as Google Docs, Canva, HackMD, and CaptionAI. This involves in-depth exploration and analysis of their technical architecture, underlying data structures, synchronisation mechanisms, and collaborative features. Interviews and surveys with developers and users of these tools are conducted to gain firsthand insights into their design principles, implementation challenges, and user experiences. This empirical data enhances the understanding of how text editing and collaboration are achieved in practical applications.

For sub-question 1 “How is consistent text data achieved within text editing applications?” Product analysis from library methodology is used in order to examine some of the most popular tools, such as Word, Notepad, Visual studio & Visual Studio Code, Vim (includes Neovim and Spacevim) and more. Furthermore, community research is done under the form of a questionnaire that is aimed specifically towards developers to pinpoint what their tool of preference is and what some of the issues stand to be with text editing tools.

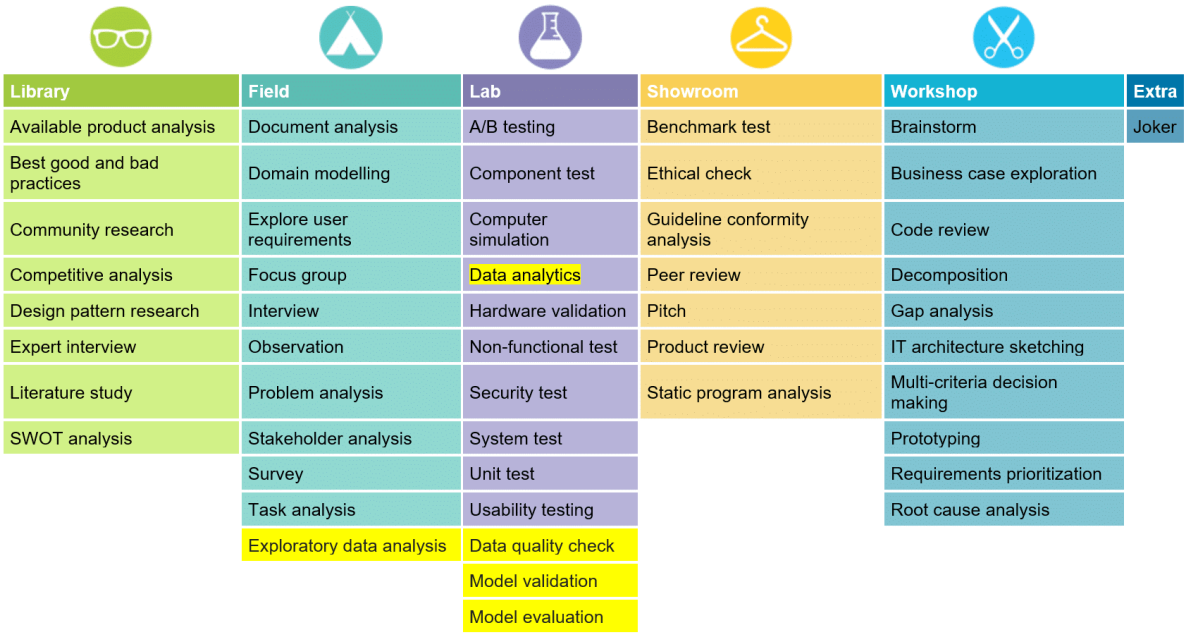
For sub-question 2: “How are text collaboration tools created and what is their importance?” a product review methodology is used to explore some of the most popular collaborative tools and what technologies they use behind the scenes. Additionally, this question explores what some of the user requirements and improvements can be given by people who use collaboration tools on a day-to-day basis. This is achieved by using the user

requirements and focus group methodologies from the field section of the DOT framework.

By employing this research methodology, the paper aims to provide a comprehensive and robust analysis of text editing and real-time collaboration. The combination of literature review, case studies, experiments, interviews, and data analysis techniques allows for a well-rounded exploration of the research questions. It ensures that the findings and conclusions are grounded in both theoretical understanding and empirical evidence.

For reference of the DOT framework, the below figure gives an overview of the different methodologies available for research.

Fig 1. ICT research methods in the DOT framework (new ones for machine learning in yellow)



Introduction to text editing

Text editing is the manipulation of characters, usually done in a specialised application that allows additional functionality/options to be used when editing.

Text editing is present in almost any application, whether it is on your phone, desktop or when browsing the web. It is also something that is often overlooked and not many people look further than what sits on the surface level.

Looking at text editing from a computer science perspective, we can break it up into two parts: the way that the text is stored in memory and the way that the text is manipulated. This is done by using specific data structures that serve the purpose of quick and accurate changes. Some of those data structures will be discussed in later sections of this research paper.

This research paper also goes over collaborative text editing and what goes behind some of the most popular tools, such as Google Docs, Miro board, Word online and more.

In order to best understand what people think of text editing and what topics should be described in more detail, it is best to hold a questionnaire over a group of people. For this research paper, a short 9 question survey was conducted over 15 people, all of which are developers and working in different fields. The following section shows the results from the survey and briefly discusses each part.

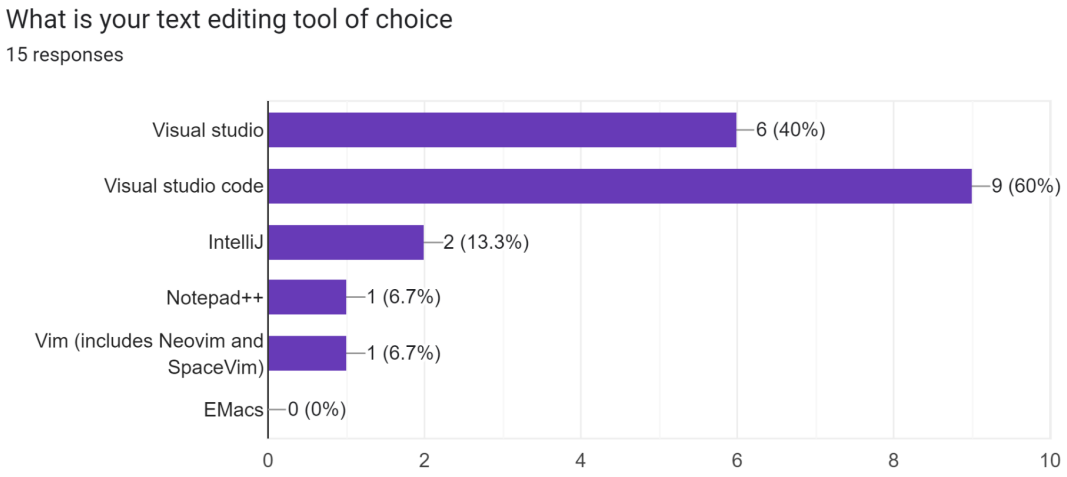
Survey and results

The survey that was conducted contains 9 questions that aim to question whether people know what goes behind text editing and collaboration and where their interests lay. The questions within the form were as follows:

1) What is your text editing tool of choice?
2) What is your background knowledge on how text editing tools work on a technical level?
3) Which aspects of text editing and collaboration are you most interested in learning about?
4) How familiar are you with the concept of operational transformation (OT) in the context of collaborative text editing?
5) In your opinion, what are the key challenges in achieving consistent text data within text editing applications?
6) How important do you think real-time collaboration tools are in enhancing productivity and teamwork?
7) Have you ever experienced any issues related to data consistency or synchronisation while using a text editing tool with real-time collaboration features?
8) Are you familiar with any specific technologies or frameworks commonly used to develop real-time text collaboration tools? If yes, please specify.
9) In your opinion, what makes a text collaboration tool stand out from others in terms of its effectiveness and user-friendliness?

For this research paper, questions 5, 8 and 9 will be omitted, as they are open-ended and do not give a clear preference/idea of what was being asked.

What is your text editing tool of choice?



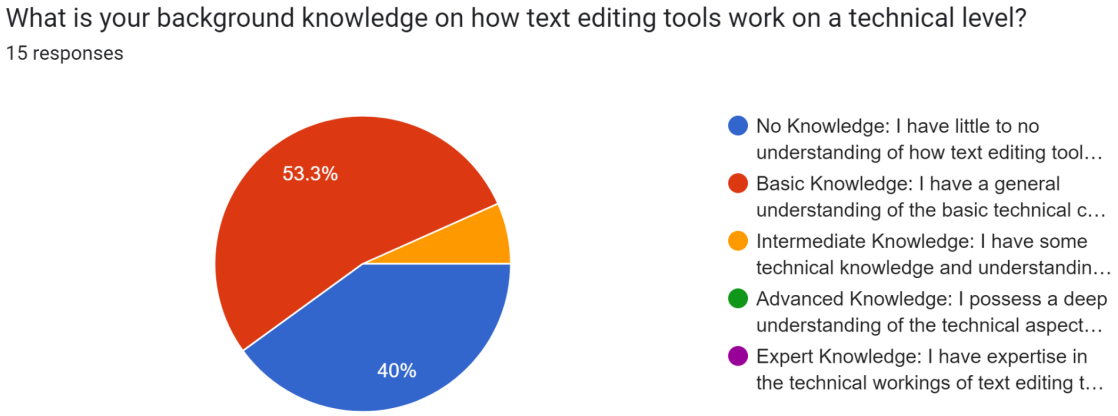
Since this questionnaire was targeted directly at developers, the allowed answers contain only developer tools, as well as the option to specify a different answer. Furthermore, people could specify up to two choices.

As the above graph shows, the most used text editors for programming are visual studio code and visual studio respectively, with others such as notepad++ and vim being less used.

The most popular choice here is clear. The reason behind this is that visual studio code is marketed as a very light weight and easy to use text editor that is full of features and available extensions.

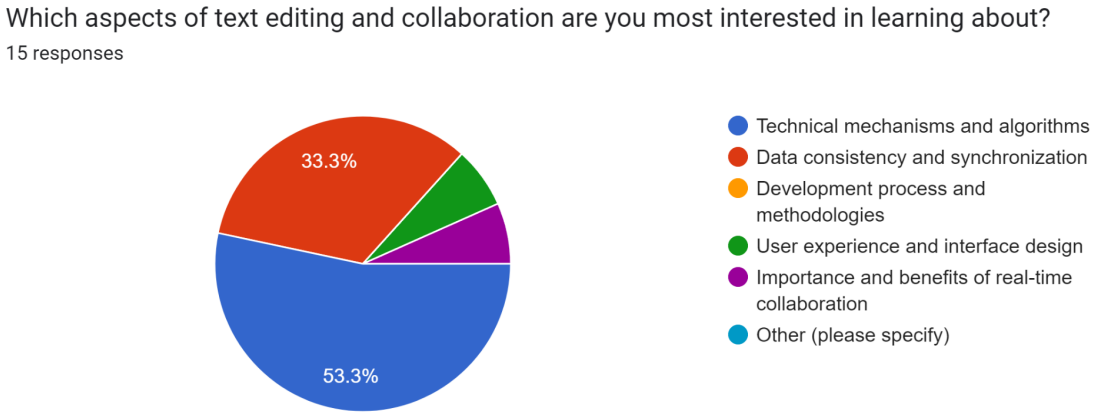
Visual studio code uses a data structure called a **piece table.** What this means is that text is held in sequential blocks, where each one after the one currently being edited is held in an immutable state. Furthermore, this allows text to be cached (temporarily saved), which has a clear boost on the performance when editing.

What is your background knowledge on how text editing tools work on a technical level?



The next question aims to see how many people are aware of the technical workings of a text editor and to what degree their knowledge spans. As the above diagram shows, 93.3% of the 15 people who have answered have basic to no knowledge about text editing on a technical level.

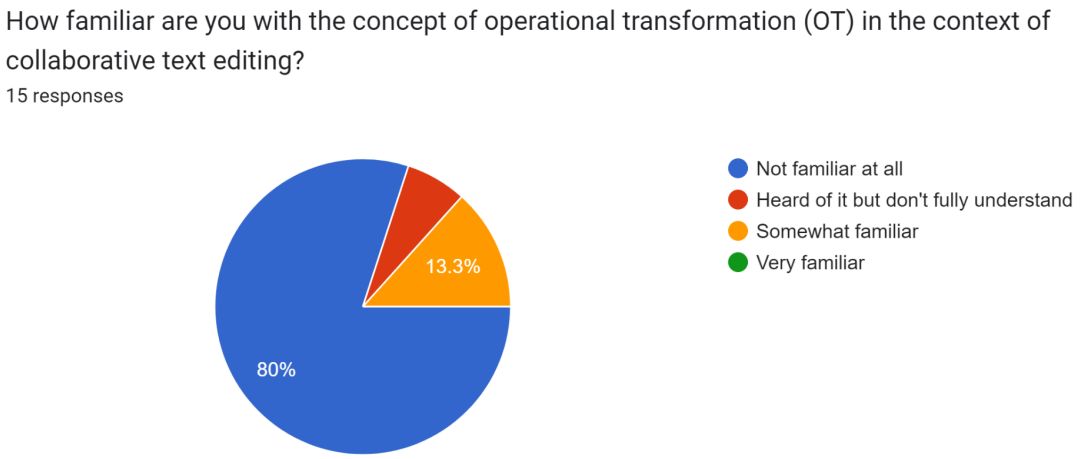
Which aspects of text editing and collaboration are you most interested in learning about?



In the above question, people are asked to specify what realm of text editing interests them the most. This is useful for this research paper, as it allows us to rule out and focus on the most important concepts while keeping people's interests in mind.

For this particular question, 53.3% of people have shown interest in technical details of text editing and 33% for data consistency and synchronisation.

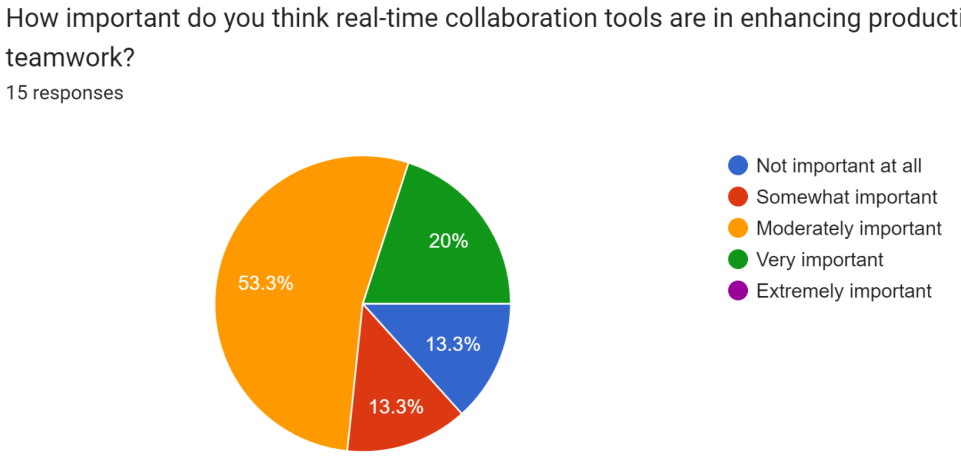
How familiar are you with the concept of operational transformation (OT) in the context of collaborative text editing?



The above question questions people's knowledge of one of the most important concepts for collaborative text editing and data synchronisation. OT or operational transformation is a collaboration of tools that allows both collaborative text editing and other types of collaboration (such as a real-time board like Miro.com).

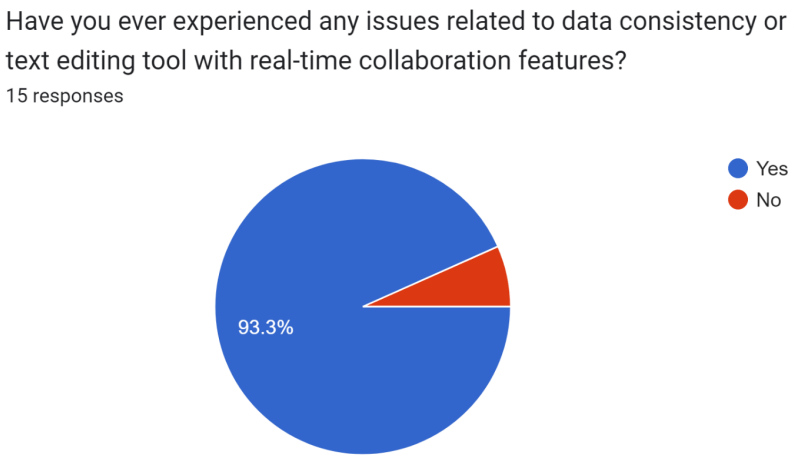
80% of all people have specified no prior knowledge to what operational transformation is in text collaboration. This research paper will later dive deeper into what OT is in essence and where it is used in today's software.

How important do you think real-time collaboration tools are in enhancing productivity and teamwork?



The above question, as well as the one after, are aimed to gather people's opinions on collaborative text editing and whether it is an essential part of team work. Additionally, it further questions whether the text collaboration technology is there yet in terms of reliability and issues.

Have you ever experienced any issues related to data consistency or synchronisation while using a text editing tool with real-time collaboration features?



How is consistent text data achieved within text editing applications?

On a technical level, text editing is all about storing the text in such a manner that it is readily accessible and easy to undergo manipulations. There are many available data structures that allow text to be edited, each having its own positives and negatives. In this research paper, 3 of the most popular and widely used data structures are described on a technical level. Those are namely:

1. *Gap buffer*
2. *Ropes data structure*
3. *Piece table*

The following sections are dedicated to a breakdown of each data structure and its use cases .

Gap buffer

Gap buffer is the most popular and widely used data structure for small to midsize text editors. It has gained its popularity due to being very easy to implement, while delivering very good performance and a time complexity that is rounded up to a constant **O(1)**.

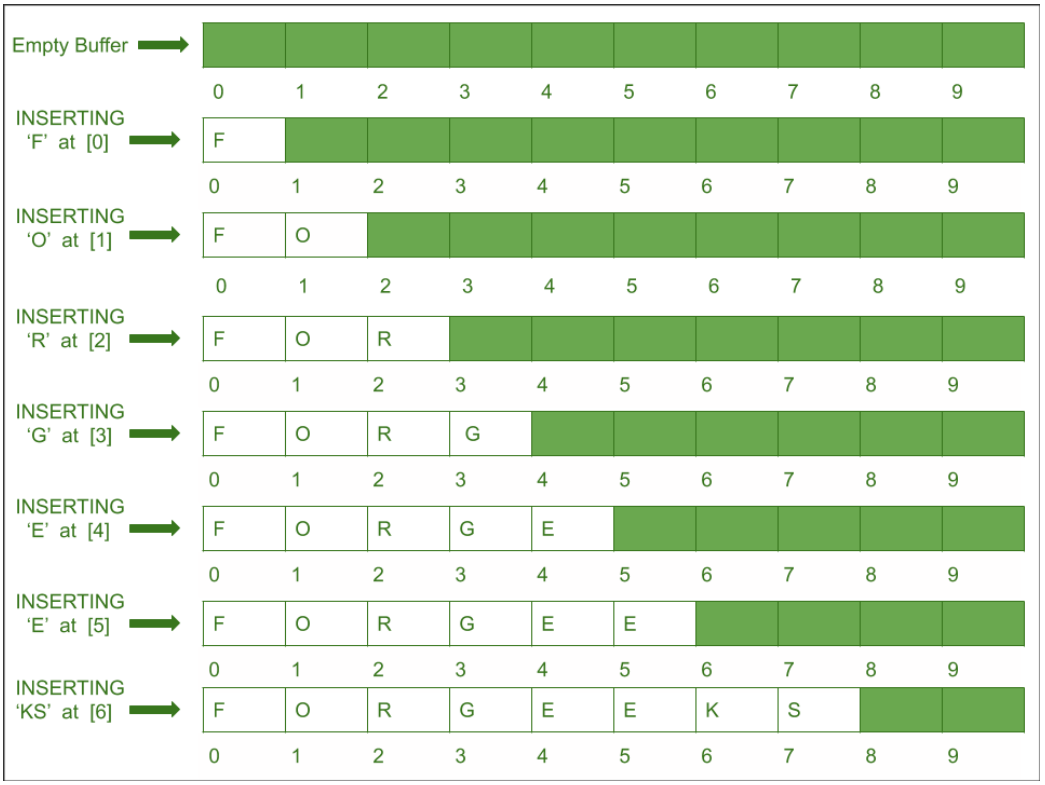
A gap buffer is a dynamic array that allows efficient insertion and deletion of elements that are clustered in the same location. A gap buffer works by creating a gap, which is usually calculated dynamically based on the amount of text in a given file. This gap serves as a means to insert or delete text efficiently at a given location without having to copy the entire array to a new memory location.
The gap buffer works by keeping track of 3 key points:

1. The start of the gap
2. The size of the gap
3. The end of the gap

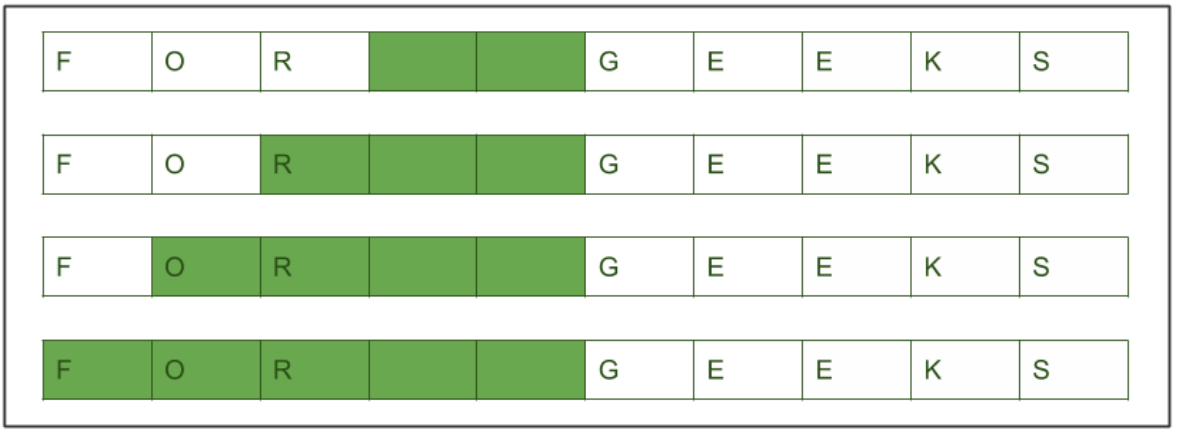
Using this information, we are able to move the gap around the array (usually where the cursor is) and delete and insert characters in an almost constant time complexity. The reason why it is not true constant complexity is that when the gap is filled in with characters, it requires to be

extended. This means that the array then needs to be copied in a new location with the additional size of the gap being added accordingly.

Gap buffer insertion, Image by GeeksForGeeks



Gap buffer insertion, Image by GeeksForGeeks



The gap moves to where the selected index in the array is and all characters are moved either to the left or the right of it, depending on the

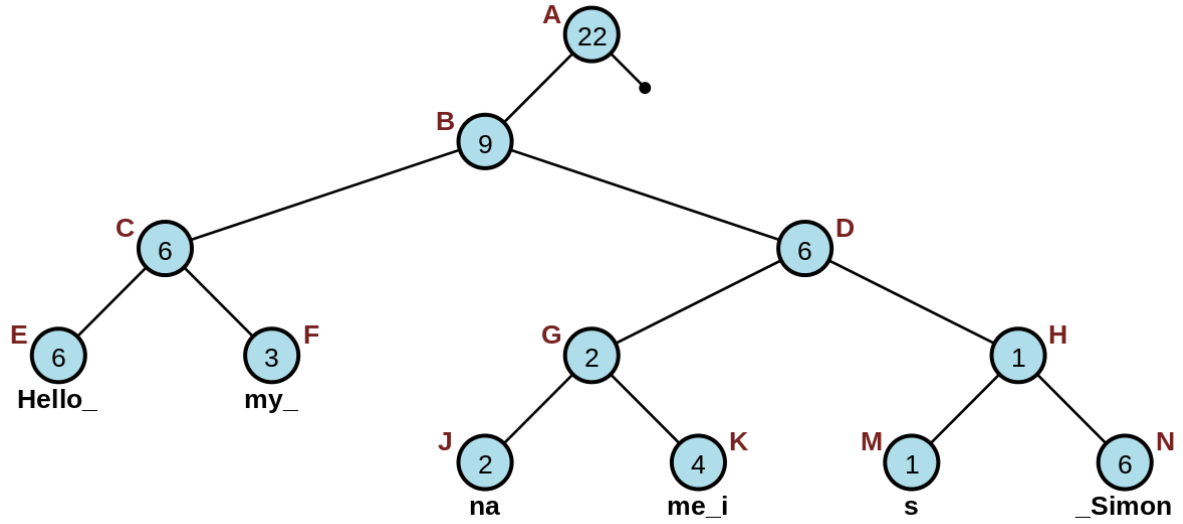
direction of where the gap is moving to. Furthermore, when deleting characters, the gap is “extended”, allowing for the empty spaces that have just been allocated free to be used up by new characters.

Overall, the gap buffer is the most popular choice due to its ease of implementation and use. It is good for small to medium sized text editors, but it requires more memory than the other two mentioned data structures.

Ropes data structure

The ropes data structure is a highly efficient and versatile data structure primarily used in text editing applications. It provides a way to represent and manipulate large, mutable strings by combining the benefits of both arrays and linked lists. This one-page technical explanation aims to delve into the intricacies of the ropes data structure and highlight its key features and advantages in text editing scenarios.

A rope is essentially a binary tree where each leaf node stores a small substring of the overall text. The internal nodes represent the concatenation of their child nodes, providing a hierarchical representation of the text. This design allows for efficient manipulation of large texts, as operations such as insertion, deletion, and substring extraction can be performed more quickly compared to traditional array-based or linked list-based approaches.



Some of the key features and benefits of using a ropes data structure are:

Efficient Concatenation:
One of the key advantages of ropes is the ability to concatenate two ropes together in logarithmic time complexity. When concatenating ropes, new internal nodes are created to represent the merged structure, while the original ropes remain unchanged. This process ensures that the concatenation operation remains efficient, even for very large texts, as it does not require copying or shifting large chunks of data.

Balanced Structure:
Ropes are designed to maintain a balanced binary tree structure, typically achieved through techniques like weight balancing or height balancing. This balance ensures that the depth of the tree remains logarithmic in relation to the length of the text, enabling efficient access to any specific index or range of characters within the text.

Efficient Splitting and Substring Extraction:
Ropes excel at extracting substrings or splitting a large text into smaller parts. By traversing the rope structure and intelligently selecting the appropriate nodes, substring extraction can be accomplished with optimal time complexity, often logarithmic or sub-logarithmic. This efficiency is particularly beneficial in scenarios where multiple small edits or operations on different parts of the text are required.

In conclusion, the ropes data structure offers an elegant solution for efficient text editing operations, especially in scenarios involving large texts. By leveraging a balanced binary tree structure and intelligent concatenation, ropes provide fast and scalable operations such as concatenation, insertion, deletion, and substring extraction. Its memory efficiency and ability to handle complex text editing requirements make it a powerful choice for modern text editors and applications dealing with extensive textual content.

Piece table

The piece table data structure is a highly efficient and flexible data structure used in text editing applications. It provides a way to represent and manipulate text by storing the original content in a read-only buffer and tracking modifications using a separate structure called the "piece table." This one-page technical explanation aims to delve into the intricacies of the piece table data structure and highlight its key features and advantages in text editing scenarios.

The piece table data structure separates the original text into two main components: the original read-only buffer and the piece table itself. The read-only buffer stores the initial content and remains unmodified throughout editing operations. The piece table, on the other hand, keeps track of the modifications made to the text, such as insertions and deletions, by storing references to the original buffer and additional data for added or removed segments.

Given the following buffers and piece table:

Buffer	Content
Original file	`ipsum sit amet`
Add file	`Lorem deletedtext dolor`

Piece table

Which	Start Index	Length
Add	0	6
Original	0	6
Add	17	6
Original	6	8

Efficient Editing:

The piece table structure enables efficient editing operations. Insertions are performed by adding new segments to the piece table, which only requires appending or linking relatively small data structures. Similarly, deletions are handled by marking or removing segments in the piece table without modifying the original buffer. As

a result, editing operations can be completed quickly, even for large texts, without requiring expensive string manipulations or data shifts.

Undo/Redo Support:

The separation of the original buffer and the piece table simplifies the implementation of undo and redo functionality. Since modifications are tracked separately in the piece table, undoing or redoing an operation involves manipulating the references and metadata within the table, without affecting the original buffer. This design allows for efficient and seamless undo/redo operations, providing a valuable feature for text editing applications.

Version Control and Collaboration:

The piece table structure naturally lends itself to version control and collaboration scenarios. By maintaining the original buffer and tracking modifications in the piece table, it becomes easier to track changes, create different versions of the text, and merge or compare revisions. The ability to handle edits as separate pieces, rather than modifying the original content, simplifies the implementation of collaborative editing features and enhances the overall flexibility of the data structure.

Memory Efficiency:

The piece table approach optimizes memory usage by storing the original content in a read-only buffer and only tracking modifications separately. This design reduces memory overhead compared to maintaining a complete copy of the modified text. Additionally, the piece table can be implemented in a compact manner, using efficient data structures and pointers, further minimizing memory consumption.

What does Copycloud utilise within its system for text editing?

Quill.js, a popular JavaScript library for text editing, uses an optimized data structure known as a "Delta" to represent and manipulate the content of the document.
The Delta data structure in Quill.js is based on the Operational Transformation (OT) concept. It represents the document content as a sequence of operations or "deltas" that capture the changes made to the text. Each delta represents an atomic operation such as insertions, deletions, or formatting changes applied to the document.

The Delta structure consists of an array of individual delta objects. Each delta object contains two main properties:

1. "Insert": This property holds the actual content that is being inserted at a specific position in the document. It can include plain text, HTML markup, or even custom objects representing rich content.
2. "Attributes": This property stores any formatting or styling information associated with the inserted content. Examples include font styles, colors, sizes, and other text or paragraph-level formatting.

By using the Delta data structure, Quill.js can efficiently represent the document's content and apply changes in a granular and incremental manner. It allows for optimized rendering, manipulation, and collaboration capabilities. The Delta structure also facilitates easy transformation and synchronization of changes across multiple users in real time.

How are text collaboration tools created and what is their importance?

Text collaboration tools have revolutionised the way individuals and teams work together on written content. These tools enable multiple users to simultaneously collaborate on the same document, regardless of their geographical location. In this introduction, we will explore how text collaboration tools are created, the technologies behind them, and discuss the importance of these tools in modern workflows.

Collaboration in the realm of written content has traditionally been constrained by physical barriers and limitations imposed by sequential editing. However, with the advent of text collaboration tools, the barriers have been shattered, opening up new avenues for real-time collaboration and enhancing productivity across industries.

The creation of text collaboration tools involves a combination of various technologies and design considerations. At its core, a text collaboration tool requires a robust infrastructure to handle concurrent editing, synchronisation, and conflict resolution. This infrastructure typically leverages technologies such as real-time communication protocols, distributed databases, version control systems, and collaborative editing algorithms.

Real-time communication protocols, such as WebSockets, facilitate instantaneous data transmission between users, enabling seamless collaboration without noticeable latency. Distributed databases play a crucial role in storing and managing the shared document, ensuring data consistency and availability across different users and devices.

Version control systems, commonly used in software development, find application in text collaboration tools to track changes made by different collaborators. These systems allow users to revert to previous versions, view revision history, and merge conflicting edits.

Furthermore, collaborative editing algorithms form the backbone of text collaboration tools. These algorithms govern how concurrent edits are handled, ensuring that conflicts are detected and resolved intelligently while preserving the integrity of the document. Techniques like Operational Transformation (OT) or Conflict-Free Replicated Data Types

(CRDTs) are employed to handle simultaneous edits and maintain document consistency.

The importance of text collaboration tools cannot be overstated. They foster seamless teamwork and enable real-time collaboration, enhancing productivity and efficiency across various domains. These tools empower teams to work together on documents, spreadsheets, code repositories, and more, irrespective of their physical location. Collaboration becomes effortless, allowing for instant feedback, parallel contributions, and collective decision-making, which ultimately accelerates project completion and drives innovation.

Text collaboration tools are particularly valuable for remote teams, distributed organisations, and global enterprises where collaboration across time zones and geographical boundaries is a necessity. They promote effective communication, foster knowledge sharing, and reduce coordination overhead, leading to enhanced collaboration and improved outcomes.

OT vs CRDT in collaboration

Operational Transformation (OT) and Conflict-Free Replicated Data Types (CRDTs) are two prominent approaches used in the field of collaborative editing to handle concurrent edits and ensure consistency across distributed systems. While both techniques aim to address the challenges of collaborative editing, they differ in their underlying principles and mechanisms. Let's explore the key characteristics of OT and CRDTs in the context of collaboration.

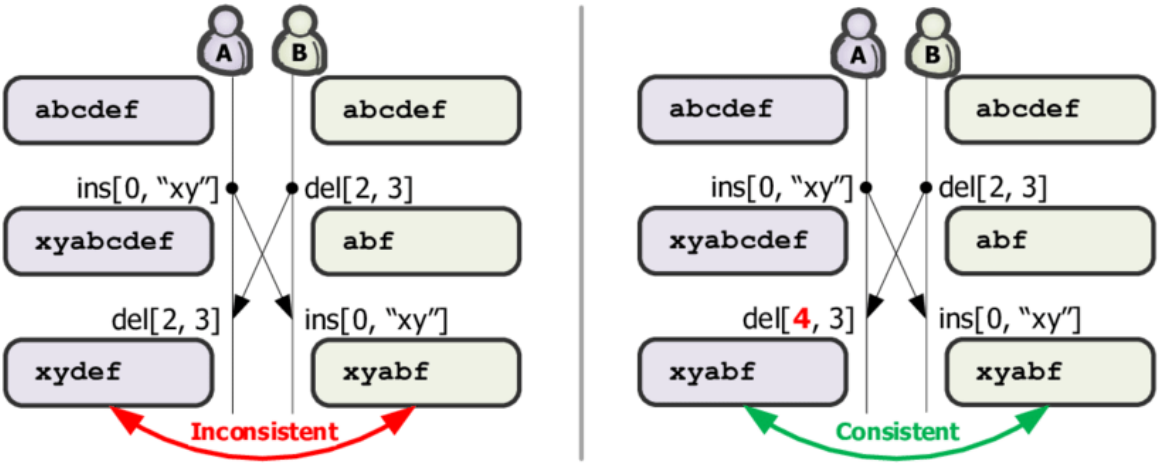
Operational Transformation (OT):
Operational Transformation is a technique that focuses on transforming operations to achieve consistency among collaborating users. OT works by decomposing editing operations into a sequence of primitive operations and applying transformations to resolve conflicts that arise due to concurrent edits.

Key features of OT include:

Transformation Functions: OT employs transformation functions that are designed to modify operations based on their position and context within the editing history. These functions help ensure that operations are appropriately applied and integrated into the shared document, even when edits occur concurrently.

Centralised Approach: OT traditionally requires a centralised server that coordinates and manages the collaborative editing process. The server receives, transforms, and propagates operations to all participating clients, acting as a central authority to enforce consistency.

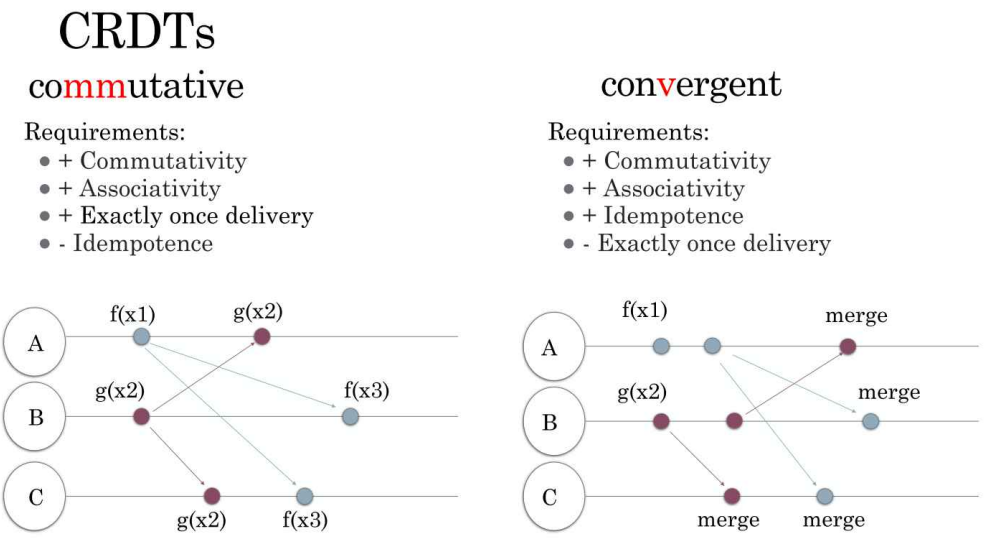
Convergence Mechanism: OT aims to achieve convergence, where all participants reach the same final document state, regardless of the order in which operations are applied. By transforming and resolving conflicts during the editing process, OT ensures that all collaborators eventually see a consistent view of the shared document.



Conflict-Free Replicated Data Types (CRDTs):
Conflict-Free Replicated Data Types are a family of data structures designed to be replicated across multiple nodes in a distributed system without requiring explicit coordination or conflict resolution. CRDTs operate on the principle of deterministic merging, ensuring that concurrent updates to the shared data structure always result in a globally consistent state.

Key features of CRDTs include:

1. Deterministic Operations: CRDTs employ operations that are inherently commutative and associative, meaning they can be applied in any order without affecting the final result. This property allows concurrent updates to be applied independently and merged consistently across replicas.
2. Decentralised Approach: CRDTs are designed for decentralised environments, where each replica can independently process operations without relying on a centralised server. Collaboration is achieved through a process of exchanging and merging updates across replicas, ensuring eventual consistency.
3. Convergence Guarantee: CRDTs provide strong eventual consistency guarantees, ensuring that all replicas eventually converge to the same state, regardless of the order in which updates are received and applied. This convergence is achieved through the deterministic nature of operations and the merge strategies employed by CRDTs.



In the context of Copycloud, the underlying technology behind the text collaboration capabilities involve the use of CRDTs through Codox. Codox is an abstraction over CRDT's which allows the ease of integration within any application.

The reason behind this design choice is that CRDT's are considerably more modern and tested. Furthermore, they are much easier to integrate and customise, which allows additional features above the text editing functionalities to be added.

Real-Time Collaboration Tools

Real-time collaboration tools have become essential in today's interconnected world, enabling individuals and teams to collaborate seamlessly regardless of their physical locations. These tools offer instant communication, simultaneous editing, and shared workspace functionalities. In this article, we will explore some popular real-time collaboration tools, their positives and negatives, and the challenges associated with latency.

Positives and Negatives

Positives

Real-time collaboration tools offer several advantages:

1. Enhanced Teamwork: These tools promote effective teamwork by allowing multiple users to collaborate on the same document or project simultaneously. This fosters better communication, coordination, and collaboration among team members.
2. Improved Efficiency: Real-time collaboration tools enable users to work together in real-time, eliminating the need for back-and-forth communication and file sharing. This reduces delays, improves efficiency, and streamlines workflows.
3. Version Control and Tracking: Many collaboration tools provide version control features, allowing users to track changes, revert to previous versions, and maintain an audit trail of edits. This enhances transparency, accountability, and simplifies collaboration among team members.
4. Flexibility and Accessibility: Real-time collaboration tools are often cloud-based, providing users with the flexibility to access and collaborate on documents from anywhere, using various devices. This allows for remote work, global team collaboration, and increased productivity.

Negatives

Alongside the benefits, real-time collaboration tools also have some potential drawbacks:

1. Learning Curve: Some collaboration tools may have a learning curve, requiring users to familiarize themselves with the platform's features and interface. This initial investment of time and effort can hinder immediate productivity.

2. Security and Privacy Concerns: As collaboration tools involve sharing and storing sensitive information, security and privacy become crucial considerations. Organizations must ensure that appropriate security measures, data encryption, and access controls are in place to protect sensitive data.
3. Dependency on Internet Connectivity: Real-time collaboration heavily relies on stable internet connectivity. In situations where the network connection is unreliable or bandwidth is limited, users may experience disruptions, delays, or even complete disconnection from the collaboration session.

Latency Issues

Latency, or the delay between an action and its response, can present challenges in real-time collaboration tools. Some factors contributing to latency include:

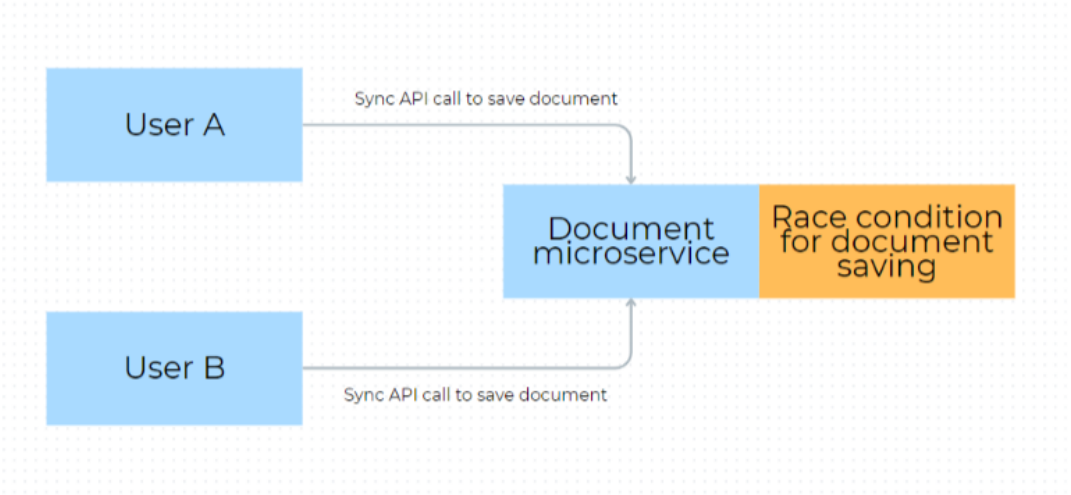
1. Network Latency: The time it takes for data to travel between users and the collaboration tool's servers can introduce latency. Factors such as distance, network congestion, and routing can impact the responsiveness of the collaboration tool.
2. Server Processing Time: The server hosting the collaboration tool may require time to process and distribute incoming data to all participants. Higher server loads or complex operations can lead to increased processing time and potentially impact real-time collaboration.
3. Device and Application Performance: The performance of the users' devices and the collaboration tool's application itself can influence latency. Older hardware, software inefficiencies, or resource-intensive operations can contribute to delays in real-time collaboration.

Efforts to minimize latency include optimizing network infrastructure, utilizing distributed servers for better geographical coverage, employing efficient data compression algorithms, and optimizing client-side rendering.

In conclusion, popular real-time collaboration tools offer numerous benefits for seamless teamwork and improved productivity. However, it's crucial to consider potential challenges such as a learning curve, security concerns, and the impact of latency on real-time collaboration. By understanding these aspects, organizations and users can leverage the positives and mitigate the negatives of real-time collaboration tools to maximize their effectivenes

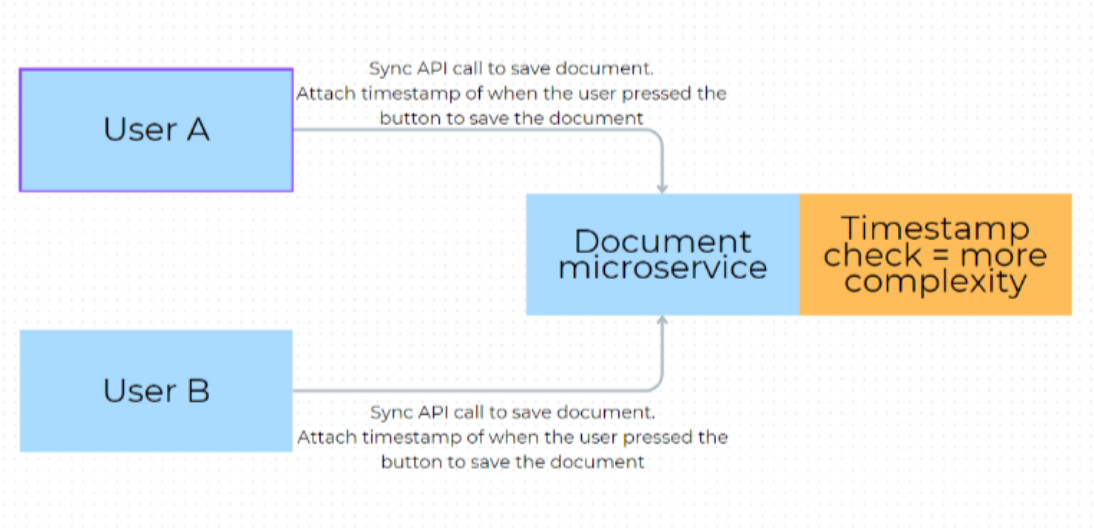
Issues with Copycloud

Example 1 - Race condition



Initially, Copycloud used a simple REST API structure, where two users could collaborate in a document and any of them could save the document in a synchronous manner. This, however, is inconsistent, as it leads to race conditions based on which user has less latency.

Example 2 - Timestamp check



The second proposal for fixing such an issue was to keep a synchronous call, but instead publish a message to a queue with a unique, globally centralized timestamp. This means that if a given situation happens where both user A and user B decide to save a collaborated document at the same time, a message would be published which would contain a globally synchronised time (converted to UTC) and make a saving decision based on that. The problem here, however, is not solved, but rather avoided. A synchronous call must still be made to a message queue, which ultimately leads to the same outcome as before.

In order to get to a proper fix to this problem, we take a closer look at how Google (Google Docs) deals with this issue.

In Google Docs, if User A and User B make simultaneous edits to the same part of a document, Google employs an operational transformation (OT) algorithm to intelligently resolve conflicts and determine the final state of the document. The goal is to preserve the intent of each user's edit while maintaining document consistency.
When multiple users make edits simultaneously, their changes are treated as separate operations. These operations are associated with a specific location in the document, such as an insertion or deletion at a particular position. Google Docs tracks the order in which these operations occur to resolve conflicts accurately.
When a conflict arises, the OT algorithm analyzes the operations and their order to determine how to merge the changes. It considers factors such as the position of the edit, the type of operation (insertion or deletion), and the timestamps associated with each operation.

The OT algorithm follows a set of rules to resolve conflicts:

1. Timestamp Comparison: Each operation is associated with a timestamp indicating when it was performed. The algorithm compares these timestamps to establish the chronological order of the edits.
2. Operation Transformation: The algorithm transforms the conflicting operations based on their types and positions in the document. For example, if User A inserts text at a specific location while User B deletes the same text, the algorithm may choose to preserve User A's insertion and discard User B's deletion. The algorithm applies transformations to ensure that the order and intent of the edits are preserved.

3. Preserving Intent: The OT algorithm aims to preserve the intent of each user's edit as much as possible. It prioritizes retaining the overall meaning and purpose behind the changes while resolving conflicts.
4. Context Awareness: The algorithm takes into account the context of the edits to make informed decisions. It considers the neighbouring content, formatting, and structure of the document to minimise disruption and maintain document coherence.

By applying these rules, the OT algorithm resolves conflicts in a way that balances the edits of all collaborators, ensuring a coherent and consistent document state. This approach allows Google Docs to handle concurrent edits gracefully and provide a seamless collaborative editing experience.

From the above, if we draw a comparison between the collaboration and latency issues of Copycloud and Google, we see that Google Docs also utilises a globally centralised timestamp check, but additionally applies the operational transformation data structure, as well as taking into context what has been edited. Furthermore, Google keeps track of previous edits, allowing for data to be restored in the case of latency/network issues that cause data loss.

Conclusion

Text editing and collaboration - how is it achieved?

Text editing is something that most people take for granted and is often overlooked on a technical level.

On a higher level, text editing is just the manipulation of a set of given characters, usually by utilising a 3rd party tool. Looking deeper into technical details, however, we discover that there is much more to think about than what is obvious. A lot of techniques and memory utilisation tricks are used by some of the most popular text editing tools. Data consistency and performant editing is achieved with specialised data structures that all differ in their complexity, implementation difficulty and memory usage.

Text collaboration, on the other hand, has its own set of challenges that have to be thought of ahead of time. Developers must deal with latency issues, data consistency, security and performance overhead. Operational transformations (OT) and CRDT's come into play for addressing the issue of real time collaboration and the manipulation of the same piece of information at the same time. Furthermore, context preservation and timestamp checks are used to address the saving of such collaborative environments.

In the context of Copycloud, this research paper addresses what techniques are used internally within the system for the actual text editing part of the application, as well as how collaboration (of up to 3 people) is achieved. Furthermore, it goes over some of the hurdles that appear when developing a collaborative environment.

Overall, understanding the intricacies of text editing and collaboration is essential for building efficient and reliable applications. By employing appropriate data structures, synchronization mechanisms, and conflict resolution strategies, developers can create powerful text editing and collaboration tools that enhance productivity and enable seamless collaboration among users.

References

Piece table | https://en.wikipedia.org/wiki/Piece_table

Operational Transformations |
https://en.wikipedia.org/wiki/Operational_transformation
https://medium.com/coinmonks/operational-transformations-as-an-algorithm-for-automatic-conflict-resolution-3bf8920ea447

Gap buffer | https://en.wikipedia.org/wiki/Gap_buffer

Dynamic array | https://en.wikipedia.org/wiki/Dynamic_array

Google Docs and Collaboration |
https://edu.gcfglobal.org/en/googledocuments/sharing-and-collaborating/1/

Quill JS | https://quilljs.com/docs/quickstart/

CRDTS and their use in collaboration | https://zed.dev/blog/crdts