CS 201: Data Structures II
Finger Tree Implementation

L4 Group 9 - All Roped Up

Spring 2023

1 Group Members

1. Simal Anjum
2. Muhammad Talha Salani
3. Dua e Sameen

2 Data Structure

A Rope data structure is a tree data structure that is used to store or manipulate large strings in a more efficient manner. It was first introduced by Hans-Juergen Boehm and Russ Atkinson in their 1995 paper 'Ropes: An Alternative to Strings'.

Ropes are an alternative to traditional string data structures that offer better performance and scalability for large text documents. Ropes data structure is a tree-based data structure used for the efficient handling of long strings by breaking the string into smaller pieces.

A rope is a binary tree that has strings and their lengths associated with each of its leaf nodes. Each non-leaf node stores the sum of the lengths of all its left subtree's leaves. Thus, a non-leaf node with two children divides the complete string into two parts, with the left subtree holding the first part of the string and the right subtree holding the second part. The weight of a non-leaf node is the length of its left subtree's string. In the context of rope operations, the strings stored in nodes are considered as unchanging and fixed objects in the usual non-destructive scenario. This allows for some copy-on-write behavior, which means that the data is not modified directly, but instead, a new copy is made only when necessary.

3 Application

The rope data structure is a specialized data structure designed for efficient manipulation of large strings. It can be used in various applications that involve text processing and editing, such as:

Some of these applications are as follows:

- Text editors: Ropes are used in text editors to provide efficient text manipulation operations, such as insertion, deletion, and substring extraction, on large documents. Examples of text editors that use ropes include Emacs, Eclipse, and Visual Studio Code.

- Version control systems: Version control systems, such as Git and SVN, use ropes to represent the differences between different versions of a file or a document. Ropes are used to efficiently compute and store the changes between two versions of a document.s
- Compiler front-ends: Ropes are used in compiler front-ends to represent the source code of a program. This allows efficient parsing and manipulation of the program code during compilation.
- DNA sequencing: Ropes are used in bioinformatics to store and manipulate DNA sequences. Since DNA sequences can be very large, ropes provide an efficient data structure for storing and processing them.
- Web browsers: Ropes are used in web browsers to efficiently represent and manipulate HTML documents. This allows for fast and efficient rendering of web pages, even when the pages contain large amounts of text and markup.

Overall, ropes are a useful data structure for any application that involves manipulating large strings or documents, and they offer a number of advantages over other data structures, such as arrays and linked lists.

4 Functionality

4.1 Interface

The Interface would provide the following functions:

1- init(rope): Initializes a new Rope object with a string s

2- empty(): Returns an empty rope.

3- concat(left, right): Concatenates two Rope instances left and right into a new Rope instance..

4- Searching(indexing): returns the character at ith position, we search recursively beginning at the root node.

5- Splitting(self, i): Splits the Rope at the specified index i.

6- Insertion(rope, x): takes a Rope instance rope, a string s, and an index i, and inserts the string s into the Rope at the specified index i.

7- Deletion(rope): This delete function takes a Rope instance rope, a start index start, and an end index end, and removes the characters from the Rope between the start and end indices.

4.2 Amortized Analysis

When analyzing the ropes data structure using amortized analysis, we can consider two main operations: split and concatenate.

Split operation: When splitting a rope into two parts at a specified index, the worst-case time complexity is O(log n), where n is the length of the rope. However, on average, each character in the rope is split only once, so the total time complexity of all split operations on a rope of length n is O(n).

Concatenate operation: When concatenating two ropes, the worst-case time complexity is O(log n), where n is the total length of the two ropes. However, the average time complexity of concatenating two ropes of length n1 and n2 is O(log(n1) + log(n2)), which is O(log(n1n2)).

Therefore, using amortized analysis, we can conclude that the overall time complexity of a sequence of split and concatenate operations on a rope of length n is O(n + m log(n)), where m is the total number of concatenate operations performed.

In practical applications, the ropes data structure has been shown to provide efficient performance for large string manipulation, making it a useful tool in text editors and other software applications that deal with large amounts of text.

5 Datasets

Haven't come across a relevant dataset yet. Will add it later when we'll find a suitable one.

6 Work Distribution

Fill in the table which indicates the work distribution of each member.

Item	Activity	ID
1	Implementation	ds07138
2	Testing	sa07716
3	Debugging	ms07725

7 Attribution

References

[1] OpenAI. *ChatGPT*. [Online]. Available: https://openai.com/chat . Last accessed: 18 Apr 2023. Prompt: Applications of Ropes data structure

[2] @onlinerope-data-structure, author = Vivek Kumar, title = Rope Data Structure, year = 2020, url = https://medium.com/underrated-data-structures-and-algorithms/rope-data-structure-e623d7862137, urldate = 2023-04-18