MULTITHREADED DOCUMENT PROCESSING

DUȚU ALIN
CĂLIN

CONTENTS

1 INTRODUCTION

1.1 Context

Document processing is a technology concept encompassing a wide array of methods and tools designed to handle, manage, and manipulate documents in both digital and physical forms. This technology includes processes such as scanning, optical character recognition (OCR), data extraction, classification, and storage.

1.2 Objectives

This project aims to develop a program that makes use of multithreading technology to process multiple documents and compute meaningful metrics with which the documents can be classified and reordered. The following objectives are covered after implementing the client:

- Understand multithreading mechanism and its implementations
- Discover new paradigms for large document processing
- Understand the flow of document processing

2 PROPOSED SOLUTION

This project proposes a solution that classifies documents using the Map-Reduce paradigm, similar to what Google uses for processing big data in distributed systems.

Map-reduce[1] is a model used in parallel programming for processing big data by making use of the computational power of today's processors. The mechanism behind this model has 2 phases:

- Map - The map function, processes a single key/value input pair and produces a set of intermediate key/value pairs.
- Reduce - The reduce function, consists of taking all key/value pairs produced in the map phase that share the same intermediate key and producing zero, one, or more data items.

In the current context, the input is represented by several N documents with big-sized texts. The first phase is the Map phase and it consists of splitting all documents into multiple complete fragments and associating each fragment to one thread to be prepared for the next phase. The second and final phase is the Reduce phase in which the rank for each document is calculated based on the results from the previous phase and presented in descending order.

3 IMPLEMENTATION DETAILS

3.1 Proposed implementation

The proposed solution principally consists of an application implemented in Java that receives the number of documents, the maximum size of a fragment, and the paths to each document.

The program initializes a multithreading implementation used to handle the first phase, an Executor Service. The Executor Service is a platform designed for parallel programming in which the main thread can call multiple threads to execute one specific task and allows the workers to further create new tasks if necessary.

In the Map phase, the workers parse the provided documents into multiple fragments. As no document dimension is provided, the program uses, at first a number of workers equal to the number of documents processed.

One such worker parses the file until it has reached the maximum fragment length, but if the fragment ends in the middle of one word, the thread continues to parse until it reaches a character that is not a letter. Each worker returns a word frequency dictionary with the number of appearances for each processed word. Following the Map phase, the main thread is going to group the resulting dictionaries by the parsed document as a preparation for the following phase.

In the Reduced phase, the workers gather the dictionaries for one document, compute the maximum length a word can have, and make a list with the words that have the maximum length. In the final part of the phase, the document rank is calculated and returned using the following formula:

$$\frac{\sum_{k=1}^{n} F(lungime_k) \times numar\_aparitii_k}{numar\_cuv\_total}$$

When the Reduce phase is over, the main thread is going to sort all documents based on their ranking and write the result into an output file.

3.2 Testing

For testing the proposed solution, the following methods have been used:

- Checker - Used for testing general cases
- Manual testing - Used for testing corner cases

4 CONCLUSIONS

In conclusion, this project proposes a solution for processing and document classification using the MapReduce model, implemented in Java that demonstrates the efficiency and scalability in processing big volumes of data such as documents. By implementing a distributed architecture, the app permits faster parallel processing, reducing the necessary time for analyzing and classification compared to traditional methods.

Bibliography

[1] IBM. The MapReduce paradigm. https://www.ibm.com/docs/en/netezza?topic=guide-mapreduce-paradigm. Latest access: 27.07.2024.

[2] APD team @ Polytechnic University of Bucharest. Procesarea de documente folosind paradigma Map-Reduce. https://archive.curs.upb.ro/2021/pluginfile.php/406992/mod_resource/content/9/Tema%202%20-%20Enunt.pdf. Latest access: 27.07.2024.