INSTITUTO FEDERAL DO NORTE DE MINAS GERAIS

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GUSTAVO HENRIQUE ALVES ROCHA,
NATÃ TEIXEIRA SANTOS DE OLIVEIRA

RELATÓRIO – PRÁTICA 5

ORGANIZAÇÃO E SISTEMAS DE ARQUIVOS

Montes Claros, MG
2025

Sumário

1 Introdução **1**

2 Classes Utilizadas **1**

2.1 IndiceInvertido 1

2.2 IndiceInvertidoManager 1

3 Criação do Índice Invertido **2**

4 Consulta e Inserção de Registros **2**

5 CLI **3**

6 Testes Realizados **5**

7 Análise dos Resultados **6**

1 Introdução

O presente relatório tem como objetivo detalhar o desenvolvimento da Prática 5, cuja finalidade é implementar um sistema para a criação de um índice invertido, promovendo buscas eficientes em um arquivo binário que contém registros de livros. Para tal, será utilizado um arquivo de dados no formato CSV, contendo informações sobre livros, sendo essas: ID, título, autores, ano de publicação e gênero.

O processo inclui a construção do arquivo principal com registros de tamanho variável, com delimitador e descritor de tamanho, a criação de um índice primário, o qual é mantido em memória por meio de uma árvore binária de busca, e um índice invertido, que é mantido em memória através de um mapa, bem como a implementação de operações de consulta, atualização do arquivo principal e atualização de índices.

2 Classes Utilizadas

2.1 IndiceInvertido

Classe criada visando promover uma abstração em memória para um índice invertido. Um índice invertido possui como atributos sua palavra e um conjunto de IDs dos registros que possuem tal palavra em seu título.

2.2 IndiceInvertidoManager

Nas práticas anteriores, foi requerido que o sistema fosse implementado usando três classes: *Buffer*, Registro e Índice, o que por sua vez tornou a classe *Buffer* sobrecarregada de atributos e métodos. À luz disso, optamos por criar uma abstração à parte para centralizar o índice invertido e suas operações, sendo essa a classe IndiceInvertidoManager.

Essa classe possui os conjuntos de *stop words* e pontuações, o mapa contendo os índices invertidos e os métodos afins aos índices invertidos.

3 Criação do Índice Invertido

Tendo em vista que o processo de leitura do CSV, a criação dos arquivos binários e a criação do índice primário foram detalhados no relatório da prática anterior, tais detalhes serão omitidos.

Para a criação do índice invertido, primeiro populamos o mapa em memória. O processo é semelhante ao de criação do índice primário: enquanto cada registro é lido do CSV e escrito no arquivo binário, concomitantemente, seus respectivos índices, primário e invertido, são criados e armazenados em memória. O índice invertido é mantido em memória como um mapa não ordenado. Após o término da leitura do CSV, os índices são escritos nos respectivos arquivos de índices.

Durante a construção da *string* binária correspondente a um índice invertido, escrevemos primeiro o tamanho do registro, depois a palavra e, por fim, os IDs. Cada campo é separado por "||". Desse modo, os índices invertidos são salvos em arquivo como registros de tamanho variável, com descritor de tamanho e com delimitador de campo.

4 Consulta e Inserção de Registros

Para realizar a consulta de um registro, o usuário deve inserir o título ou as palavras-chave desejadas. Após inserido, as *stop words* e pontuações são removidas. Em seguida, para cada palavra no título, acessamos o mapa com os índices invertidos e obtemos o conjunto de IDs vinculados àquela palavra e o adicionamos a um vetor de conjuntos. Com o intuito de aprimorar a precisão da busca, caso uma ou mais palavras do título digitado pelo usuário não estejam no mapa, é retornado um conjunto vazio.

De posse do vetor de conjuntos, é realizada a interseção entre os conjuntos de modo a obter apenas os IDs que estão presentes em todos os conjuntos. Posteriormente, para cada ID, acessamos o respectivo índice primário na árvore, obtemos o *offset* e localizamos o registro referente ao livro. Os registros são então exibidos na interface de linha de comando. O funcionamento da inserção de novos registros se mantém o mesmo da Prática 4.

5 CLI

Para facilitar a navegação entre os resultados das consultas, a CLI da Prática 4 foi estendida:

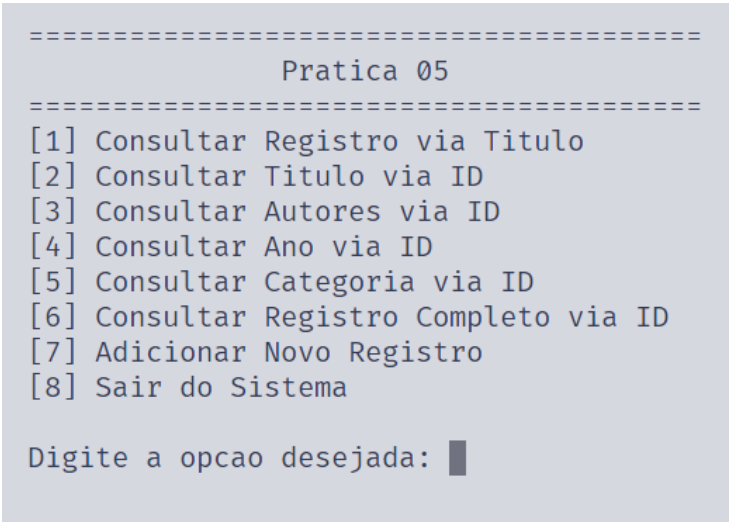

Figura 5.1 – Momento inicial da CLI

Após selecionar **Consultar Registro via Titulo**, o usuário é redirecionado para a seção de inserção do título a ser pesquisado. O usuário pode pesquisar por palavras-chave, por exemplo, "*Book*":

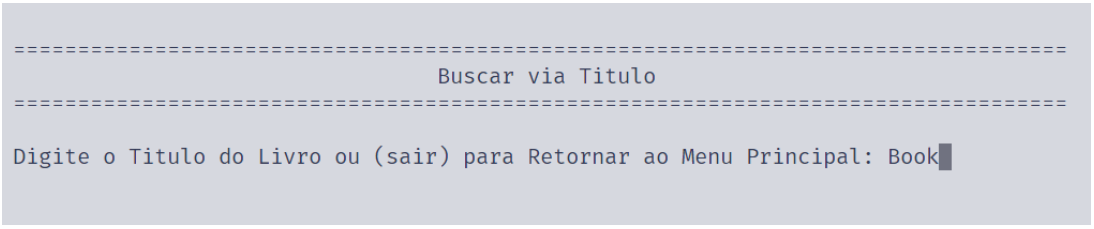

Figura 5.2 – Consulta por Livros com a Palavra *Book*

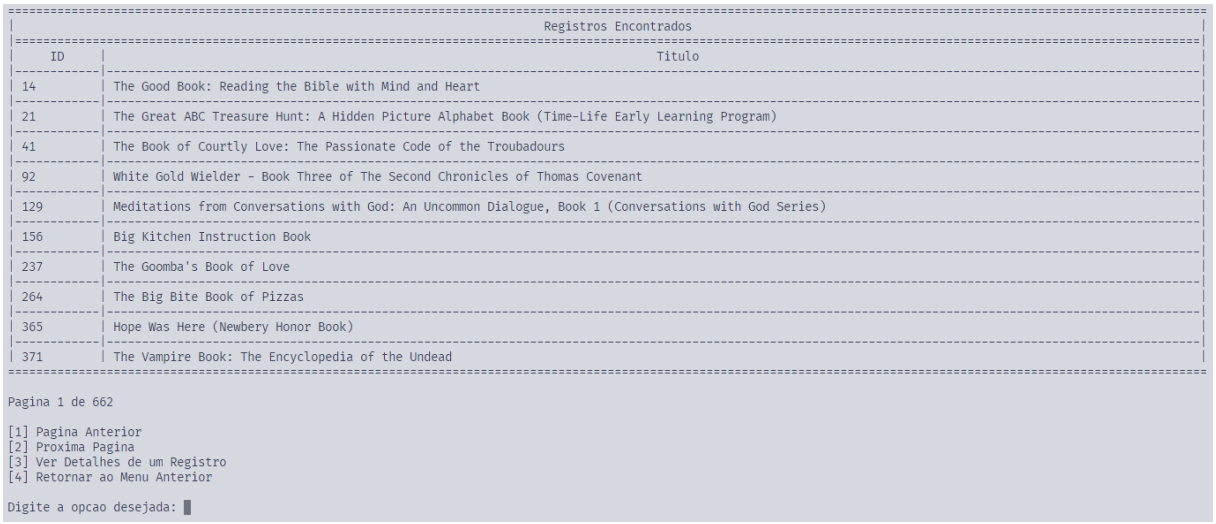

Figura 5.3 – Listagem dos Registros que contém "*Book*" no Título

Ou pesquisar por um registro específico:

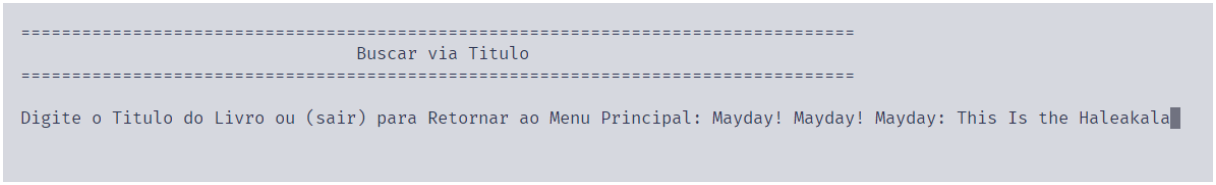

Figura 5.4 – Consultando um Registro Específico

Figura 5.5 – Listagem do Registro Específico

É possível ver os detalhes de um dos registros encontrados:

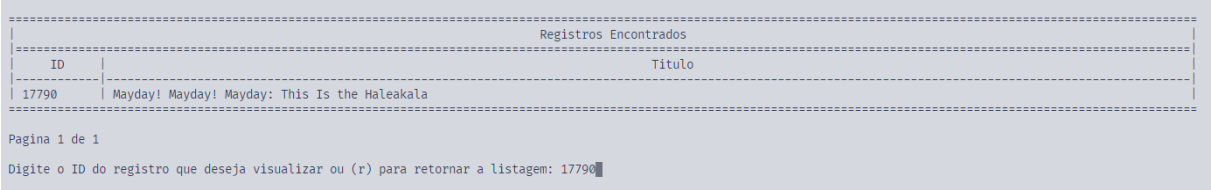

Figura 5.6 – Solicitando os Detalhes de Um Registro

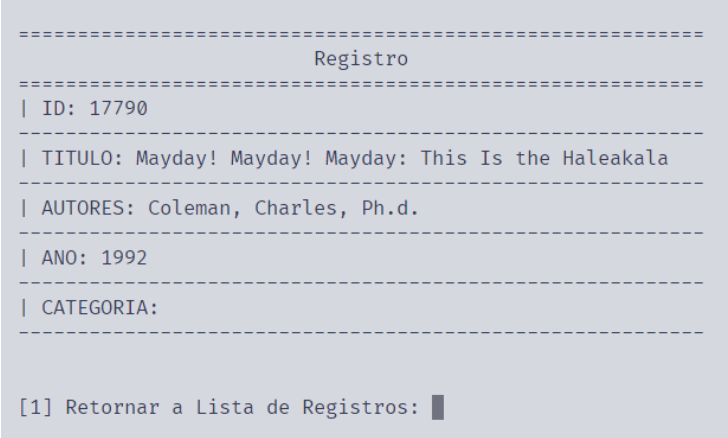

Figura 5.7 – Detalhes do Registro

6 Testes Realizados

Uma vez que o tempo amortizado da consulta ao mapa é $O(1)$ e não sabemos como a função que realiza a interseção dos conjuntos é implementada internamente, optamos por mensurar a performance do sistema por meio do tempo de execução em vez do número de operações.

No documento do trabalho, é solicitado que o sistema seja eficiente e lide bem com arquivos grandes. À luz disso, realizamos os testes em 3 cópias do CSV original, além do CSV em si. Os arquivos possuem, respectivamente, as primeiras 1.000, 10.000 e 50.000 linhas. O arquivo original possui 103.064 registros.

Foram realizados 3 testes: Leitura do CSV e criação dos índices, busca por uma palavra (a palavra utilizada na busca foi "*Book*") e busca por um título específico ("*The Gluten Free Gourmet: Living Well Without Wheat*"). A seguir, estão os tempos de execução dos testes realizados:

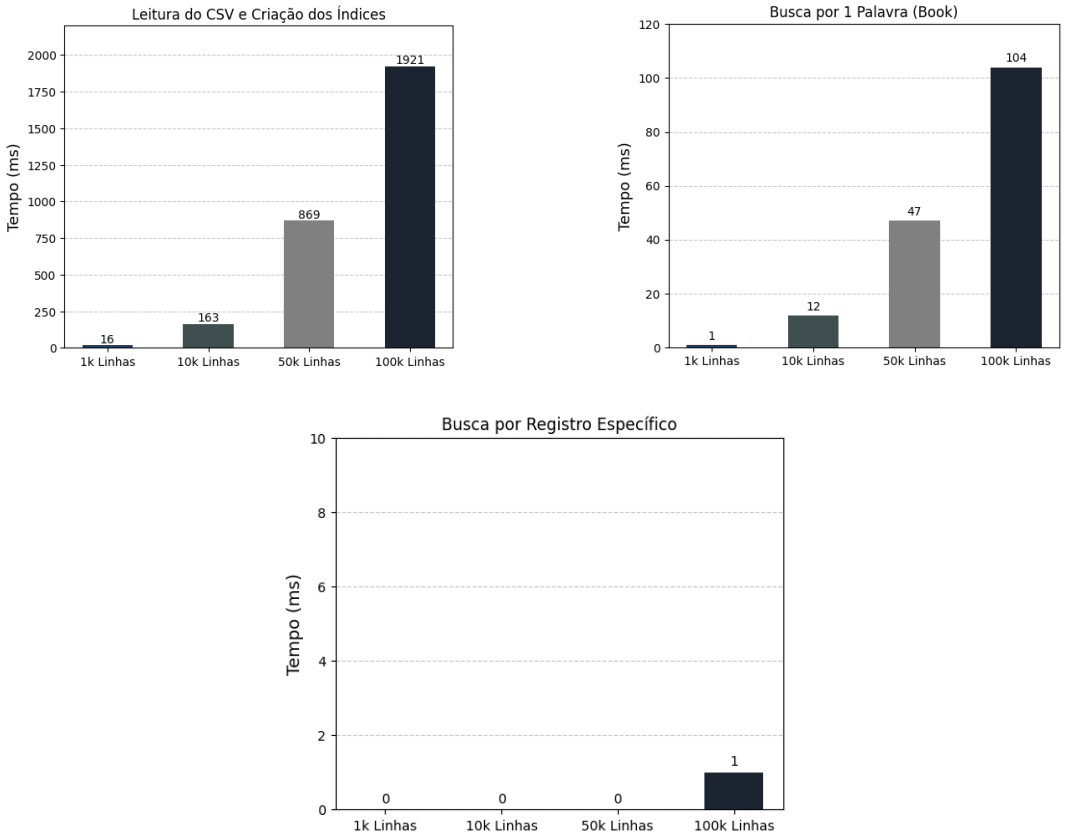

Figura 6.1 – Performance nos Testes

7 Análise dos Resultados

À luz dos resultados obtidos, percebe-se que a utilização de índices invertidos é de grande importância no que diz respeito a tornar eficiente o processo de busca por registros em grandes volumes de dados. A performance nos testes comprovou a eficiência da implementação, visto que, mesmo com o aumento do tamanho do arquivo, o tempo de execução das buscas não cresceu demasiadamente. Ademais, a utilização de um mapa não ordenado, o qual possui tempo de busca amortizado $O(1)$, corroborou a alta performance do sistema. Tais resultados atestam a eficiência da implementação e, por conseguinte, atestam sua corretude.