UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA

Trabalho Final de Estrutura de Dados

Gabriel Assis dos Santos
Maximus Borges da Rosa
Thiago Vito Paes de Farias

SETEMBRO DE 2023

1. Introdução

Esse trabalho tem como objetivo comparar duas estruturas de dados, a Árvore Binária de Pesquisa (ABP) e a Lista Simplesmente Encadeada (LSE). Testando com valores ordenados e não ordenados, iremos inserir 5.000, 10.000 e 100.000 de dados, que serão números inteiros, e com base nisso, colocaremos lado a lado os dados semelhantes das estruturas e analisaremos o tempo de inserção e o número de comparações utilizadas.

2. Estrutura de Dados e Geração de Dados

Lista Simplesmente Encadeada (LSE) é uma estrutura de dados do tipo Lista que armazena um conjunto de elementos em sequência, onde cada elemento tem uma marcação para o próximo, que não precisa necessariamente estar ao lado na memória, devido à alocação dinâmica de memória. Além disso, o número de elementos não é fixo, possibilitando adicionar e remover elementos a qualquer momento. Essa estrutura foi escolhida por ser de fácil manuseio.

Árvore Binária de Pesquisa (ABP) é uma estrutura de dados do tipo Árvore onde cada elemento é conhecido como Nó e pode ter até dois Filhos, que são nós diretamente ligados a ele. Essa estrutura utiliza uma série de comparações para adicionar novos elementos. Em caso de não haver elementos na lista, será inserido no topo. Os próximos elementos inseridos passaram recursivamente por teste para saber sua posição. Se forem maiores que o elemento da atual comparação, serão alocados à direita, e esquerda caso sejam menores, isso até não haver mais elementos passíveis de comparação, o que significa que o elemento inserido achou seu lugar na árvore. Essa estrutura foi escolhida por ser uma das mais simples de utilizar, facilitando a implementação do código e por ser de um tipo diferente da outra estrutura usada, a LSE, o que gera resultados bem interessantes.

Geração de Dados:

Para gerar os dados nas estruturas ordenadas, primeiramente alocamos as estruturas na memória, em seguida, utilizando laços de repetição, adicionamos números crescentes partindo do 1 até o número correspondente à quantidade de dados solicitados. Nas estruturas não ordenadas, também começamos as alocando na memória, porém, ao invés de gerarmos dados sequenciais crescentes, geramos números aleatórios utilizando a função “aleatorio”, localizada em utils.c, que gera números randômicos na faixa de dados solicitada. Após a inserção dos dados, será feita a análise, que mostrará dados como número de comparações, tempo de inserção dos elementos, consulta dos elementos e média de tempo de consulta e de comparações. Quando todos os dados tiverem sido mostrados, as estruturas serão liberadas da memória.

3. Metodologia

Sistema Operacional: Windows 10 Pro 64-bit

Processador: Intel Core i3-4005U CPU 1.70GHz

Memória (RAM): 8,00 GB (7,78 GB)

Execução: Para fazermos esse experimento, primeiramente, é necessário abrir o executável “main.exe”. Será solicitada as letras P, M e G para selecionar a quantidade de dados a serem inseridas, P para 5 mil, M para 10 mil, G para 100 mil. Após escolher a quantidade de dados desejada, será pedido um número de 1 a 6 para selecionar qual análise será feita, será possível solicitar:

1. Apenas as ordenadas .
2. Apenas as aleatórias.
3. LSE ordenada.
4. LSE aleatória.
5. ABP ordenada.
6. ABP aleatória.

Com isso, será mostrada a inserção dos elementos até o máximo solicitado, o tempo de inserção, o número de comparações realizadas, a consulta do valor primeiro, do meio e o último no caso das ordenadas, a consulta de dez valores randômicos e a média de comparações e do tempo de consulta no caso das estruturas aleatórias. Por fim, ao final da execução, será perguntado ao usuário se deseja continuar no programa ou sair, para sair é necessário apertar a letra N e dar Enter, já para continuar, é possível entrar com qualquer tecla.

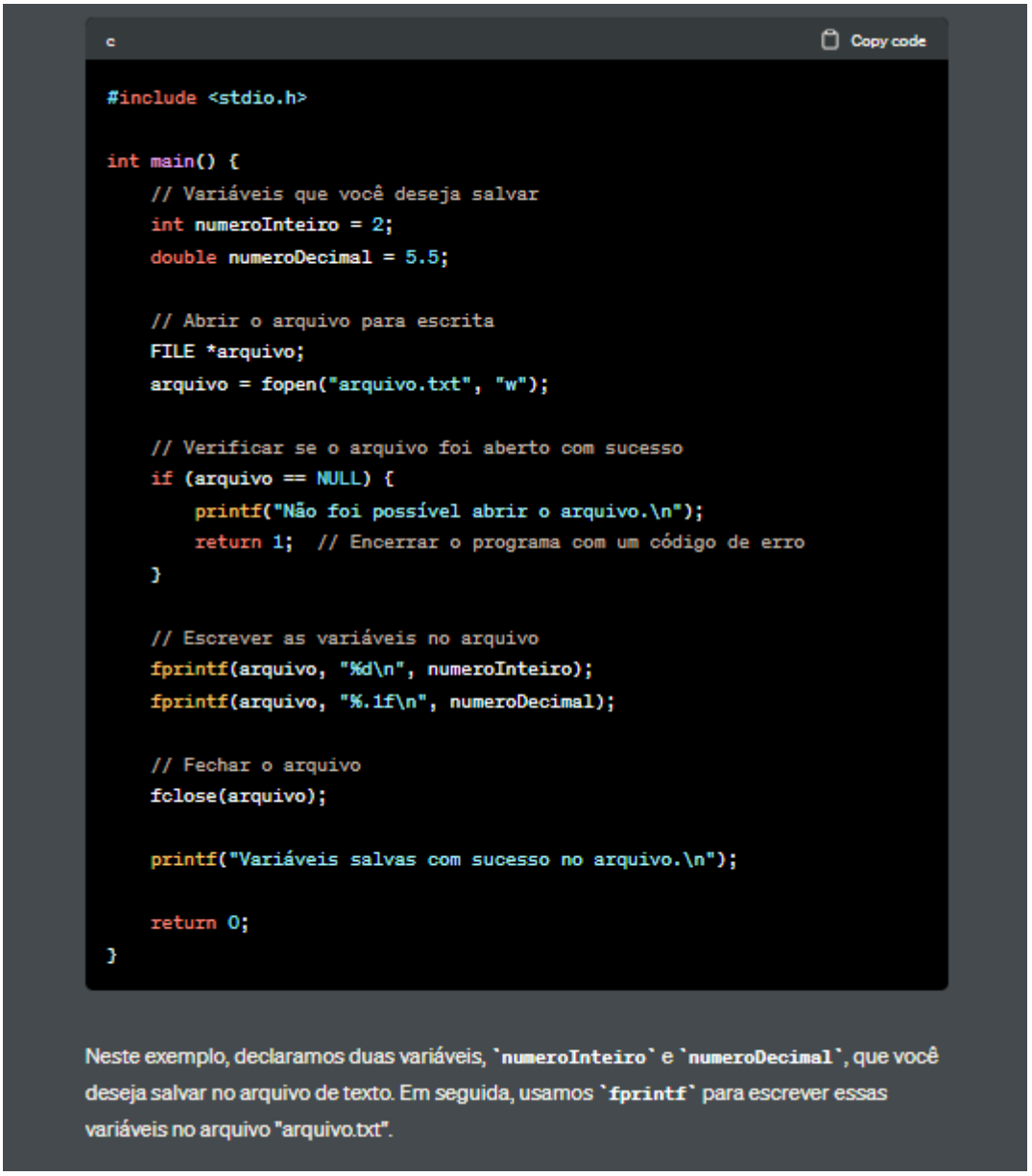
4. Análise dos Resultados

ABP x LSE Aleatórias Comparações: perto das comparações da LSE de dados aleatórios, a ABP parece que nem faz comparações, o que prova a eficiência da ABP nesse caso. No geral, a ABP se mostrou melhor do que a LSE na questão de tempo, sendo equivalentes até chegar em cerca de 10 mil dados, porém, ao passar disso, a ABP continua estável, já a LSE fica extremamente lenta.

ABP x LSE Ordenadas Comparações: as quantidades de comparações usadas nas estruturas que utilizam dados ordenados são extremamente similares.

5. OpenIA

Utilizamos o ChatGPT para tirarmos dúvidas sobre assuntos diversos. Segue abaixo um exemplo de uso para relembrarmos o uso de arquivos em C:



c Copy code

```c
#include <stdio.h>

int main() {
    // Variáveis que você deseja salvar
    int numeroInteiro = 2;
    double numeroDecimal = 5.5;

    // Abrir o arquivo para escrita
    FILE *arquivo;
    arquivo = fopen("arquivo.txt", "w");

    // Verificar se o arquivo foi aberto com sucesso
    if (arquivo == NULL) {
        printf("Não foi possível abrir o arquivo.\n");
        return 1;  // Encerrar o programa com um código de erro
    }

    // Escrever as variáveis no arquivo
    fprintf(arquivo, "%d\n", numeroInteiro);
    fprintf(arquivo, "%.1f\n", numeroDecimal);

    // Fechar o arquivo
    fclose(arquivo);

    printf("Variáveis salvas com sucesso no arquivo.\n");

    return 0;
}
```

Neste exemplo, declaramos duas variáveis, `numeroInteiro` e `numeroDecimal`, que você deseja salvar no arquivo de texto. Em seguida, usamos `fprintf` para escrever essas variáveis no arquivo "arquivo.txt".

6. Conclusão

No geral, em termos de eficiência, uma Árvore de Busca Binária de Procura (ABP) é geralmente preferível em escalas maiores, devido à sua capacidade de manter um desempenho razoável em operações de busca, inserção e remoção à medida que a quantidade de dados aumenta. No entanto, a escolha entre LSE e ABP depende das necessidades específicas do seu problema e das operações que você planeja realizar. Para pequenas escalas ou quando a simplicidade é crucial, uma Lista Simplesmente Encadeada (LSE) pode ser suficiente.

Dificuldades

Tivemos dificuldades em organizarmos nosso tempo para fazer o trabalho, pois estamos cursando outras cadeiras juntamente com Estrutura de Dados, além da dificuldade de assimilarmos nosso tempo disponível em equipe.

Utilizamos a IDE CLion para programar e nos organizar, pois existe a função Code With Me para programar com a sua equipe em tempo real com a possibilidade de compilar o código.

Lições aprendidas

Aprendemos a importância de utilizar as estruturas de dados corretamente. Na prática, toda a parte teórica se tornou muito mais visual, podendo ser facilmente observada nos gráficos das análises de dados vistas anteriormente.