Trabalho Prático II - Classificação e Pesquisa de Dados

Grupo: Davi Santos Ferrarez
Rhuan Lucas Barbosa Fernandes
Curso: Engenharia de Computação

Experimento 1)

- **Introdução**

O experimento 1 consiste em comparar o tempo de execução para inserção e busca em diferentes tipos de algoritmos de transformação de chave (hashing). A tabela hash é a generalização de um vetor com associação entre chaves e valores, seu objetivo é fazer uma busca de forma rápida através de uma chave simples e encontrar o valor desejado.
Foram utilizados dois tipos de tratamento de colisões: com listas encadeadas onde o primeiro elemento com a respectiva chave é inserido no vetor e, caso outros elementos possuam a mesma chave, são inseridos de forma encadeada com esta posição do vetor apontando para o início da lista encadeada. Já com endereçamento aberto, o vetor deve ser maior que o número de elementos a serem inseridos, pois elementos com a mesma chave são inseridos na própria lista em uma posição livre. Na sondagem para inserção linear é procurado a primeira posição livre e na sondagem quadrática a busca é feita utilizando o quadrado perfeito da posição onde ocorreu a colisão.

- **Tabela (tempo em segundos) e discussão dos resultados**

Teste 1:
Tamanho do vetor: 100 mil
Tamanho vetor endereçamento aberto: 200 mil
Tamanho vetor lista encadeada: 5 mil

	Sondagem Linear	Sondagem Quadrática	Lista Encadeada
Tempo de Inserção	0.004	0.002	0.012
Tempo de Busca	0.005	0.002	0.019

Teste 2:
Tamanho do vetor: 100 mil elementos
Tamanho vetor endereçamento aberto: 100001
Tamanho vetor lista encadeada: 99999

	Sondagem Linear	Sondagem Quadrática	Lista Encadeada
Tempo de Inserção	0.005	0.003	0.012
Tempo de Busca	0.005	0.002	0.020

Teste 3:

Tamanho do vetor: 100 mil elementos
Tamanho vetor endereçamento aberto: 150 mil
Tamanho vetor lista encadeada: 50 mil

	Sondagem Linear	Sondagem Quadrática	Lista Encadeada
Tempo de Inserção	0.004	0.002	0.011
Tempo de Busca	0.004	0.001	0.020

Todos os testes foram feitos com o vetor que será inserido contendo 100 mil elementos em ordem aleatória e os vetores das tabelas hash possuem tamanhos diferentes. O experimento consiste em retornar o tempo gasto tanto para inserir quanto para buscar todos os elementos do vetor aleatório.
Percebemos através dos três testes que o hashing de endereçamento aberto com sondagem quadrática é a estrutura que realizou o trabalho com o melhor tempo em todas as tentativas, isso acontece porque a sondagem quadrática encontra uma posição livre no vetor mais rápido que a linear e como o hashing com lista encadeada percorre toda a lista referente a chave, o tempo também é maior.

- **Conclusão**

Podemos concluir que para realizar operações com um tamanho fixo de elementos, a melhor opção é o algoritmo de transformação de chave com sondagem quadrática pois consome menos processamento e tempo que a sondagem linear. Mas para a inserção sem um tamanho fixo de elementos, é recomendado utilizar a tabela hash com lista encadeada, pois está estrutura de dados não possui um tamanho máximo de elementos a serem inseridos.

Experimento 2)

- **Introdução**

  Para a realização do experimento 2, foram utilizados dois algoritmos: a Árvore Binária de Busca (ABB) e a Árvore AVL. Em uma ABB, cada nó é um objeto e, além da chave (ou dados), cada nó contém os atributos left, right e p que apontam para os nós correspondentes da esquerda, da direita e do pai, respectivamente. Já em uma AVL, que é balanceada e obedece a todas as propriedades da árvore binária, cada nó apresenta diferença de altura entre as subárvores direita e esquerda de 1, 0 ou -1. Caso as diferenças de altura sejam maiores que 1 ou menores que -1, a árvore está desbalanceada. Esse fator é conhecido como fator de balanceamento.

- **Tabela (tempo em segundos) e discussão dos resultados**

	Tempo de inserção	Tempo de busca	Tempo de remoção
ABB	0.002	0.001	0.227
AVL	0.002	0.001	0.259

Usando-se um arredondamento de 3 casas decimais tanto na ABB quanto na AVL, mostra-se visível que o tempo de inserção e o tempo de busca em ambos os algoritmos são aproximadamente iguais, e a AVL possui um tempo de remoção menor em comparação com a ABB. O tempo de busca em ambos os algoritmos é mais rápido que o tempo de inserção, e o tempo de remoção em ambas as árvores é bastante demorado em relação aos outros dois casos, estando mais perto de 1 segundo do que o tempo de inserção e o tempo de busca. É importante destacar que foi utilizado um vetor de 10000 posições com números aleatórios para gerar cada uma das árvores.

- **Conclusão**

Neste trabalho, mais precisamente no experimento 2, foram utilizados dois algoritmos importantes na manipulação de dados, que são a Árvore Binária de Busca e a Árvore AVL. Com o uso de um vetor de 10000 posições para gerar ambas as árvores e com os resultados, sendo satisfatórios na busca e na inserção, expostos na tabela e discutidos posteriormente, o trabalho apresentou um grande aprendizado e mecanismos que podem ser usados ao longo do curso de Engenharia de Computação.

Link do GitHub para acessar o trabalho:

https://github.com/RhuanLucass/cpd-trabalho02

Link com o experimento 1 implementado:

https://github.com/RhuanLucass/cpd-trabalho02/tree/master/Experimento1

Link com o experimento 2 implementado:

https://github.com/RhuanLucass/cpd-trabalho02/blob/master/Experimento2/main.c