Relatório Tabela Hash

Guilherme Ferraz Freire De Araújo

October 2024

1 Introdução

Este é um trabalho feito em Java sobre a Tabela Hash, onde deveríamos implementar a tabela e analisar seu desempenho em diferentes casos. O meu código ficou separado em 4 arquivos: Registro, No, TabelaHash e o Main.

2 Registro

A classe Registro irá gerar uma chave para a tabela, que terá o nome de código.

3 No

A classe No irá representar uma lista encadeada, e nela há 2 atributos: registro e próximo.

O registro será responsável por armazenar os dados do registro.

O próximo faz referência ao próximo No presente na lista, e definimos aqui que o valor do próximo será null.

4 Tabela Hash

Aqui será onde eu defini minha tabela hash.

Aqui teremos 3 funções de hash (hash1, hash2, hash3), cada uma das funções terá uma chave em um índice dentro do tamanho da tabela.

- O hash1 usará o operador módulo.
- O hash2 será usado para multiplicar.
- O hash3 será usado para realizar operações com bit.

O método Inserir será onde iremos inserir um novo registro à tabela, ele vai usar a função de hash para determinar onde o registro deverá estar. Caso um elemento esteja em uma determinada posição e for colocado um novo elemento na mesma, o novo elemento será mandado para o final da lista.

O Buscar irá buscar um registro com base no seu código, aqui o hash será usado para determinar onde o registro deverá estar, caso seja necessário, ela percorre a lista ligada nessa posição para encontrar o registro desejado.

O método contagem de colisões servirá para contar quantas vezes ocorrerão colisões na tabela, as colisões acontecem quando dois ou mais elementos são mapeados para a mesma posição na tabela.

O método calcularHash seleciona qual das três funções de hash será usada, com base na configuração inicial da tabela.

5 Main

5.1 Configuração Inicial

O código define algumas constantes no início:

- TAMANHOS: diferentes tamanhos para a tabela hash (10.000, 100.000, 1.000.000).
- QUANTIDADES: diferentes quantidades de dados a serem inseridos (1 milhão, 5 milhões, 20 milhões).
- NUM_FUNCOES_HASH: número de funções de hash a serem testadas (3).
- NUM_BUSCAS: número de buscas a serem realizadas em cada teste (5).

Aqui teremos um gerador aleatório de números com uma semente fixa (42) para garantir que os mesmos números sejam gerados em cada execução.

5.2 Execução

Ao longo da execução o programa executa vários loops aninhados para testar diferentes combinações:

- Para cada tamanho de tabela
- Para cada quantidade de dados
- Para cada função de hash

Para cada combinação, o programa vai gerar um conjunto de dados aleatórios, criar uma nova tabela hash, medir o tempo de inserção de todos os dados, contar o número de colisões e realizar buscas aleatórias e medir o tempo médio de busca.

Eu usei o System.nanoTime() para medir o tempo de execução em nanossegundos.

5.3 Resultados

Agora que tudo foi explicado, os resultados serão exibidos nessa ordem:

- O tamanho da tabela, quantidade de dados e função de hash usada.
- O tempo total de inserção e número de colisões.
- O tempo médio de busca.