RA03 - Relatório de Análise de Comparações em Operações de Inserção e Busca

Thiago Grasso

November 13, 2024

1 Introdução

Este relatório visa analisar a eficiência das operações de inserção e busca em uma estrutura de dados. Para esta análise, utilizamos um arquivo contendo dados sobre operações realizadas, registrando o número de comparações necessárias para cada inserção e busca. O objetivo é compreender como o número de comparações varia entre essas operações e o que isso pode nos dizer sobre o desempenho do sistema.

2 Escolhas de Implementação

Para a análise das operações de inserção e busca em tabelas hash, foi necessário definir alguns parâmetros importantes: o tamanho do vetor, a função hash utilizada e o conjunto de dados. Abaixo estão as escolhas feitas e suas respectivas justificativas.

2.1 Tamanhos do Vetor

Escolhi tamanhos de vetor de $N = 100$, $N = 500$ e $N = 1000$ para verificar como a quantidade de posições na tabela influencia o número de comparações. Valores maiores oferecem uma perspectiva de comportamento em tabelas mais densas, enquanto tamanhos menores permitem uma análise com menor número de colisões.

2.2 Funções Hash

Escolhi as variações de função hash Divisão, Multiplicação e Dobramento; Estas funções são simples e representam bem o trabalho de funções hash em um programa, além de fornecerem resultados diretos e compreensíveis.

2.3 Conjunto de Dados

Para as chaves, usamos números inteiros sequenciais de 1 a 100, simulando uma carga típica de operações com valores que aumentam gradualmente. Esse conjunto de dados permite observar o comportamento em relação ao preenchimento gradual da tabela e o impacto nas comparações.

Essas escolhas foram feitas para facilitar a compreensão dos resultados e garantir que as operações refletissem cenários práticos. A escolha da função hash e dos tamanhos de vetor buscou minimizar o efeito de colisões ao mesmo tempo em que evidencia o comportamento das comparações em tabelas de diferentes tamanhos.

3 Descrição dos Dados

O arquivo de dados utilizado, denominado `dados_convertidos.csv`, contém as seguintes colunas:

- **Operação**: Indica o tipo de operação realizada, sendo "Inserir" ou "Buscar".
- **Chave**: Identificador único para cada item inserido ou buscado.
- **Comparações**: Número de comparações realizadas para completar a operação.

Os dados foram extraídos e organizados para facilitar a visualização do desempenho das operações em uma tabela hash, permitindo uma comparação direta entre o número de comparações em inserções e buscas.

4 Objetivo da Análise

O objetivo desta análise é avaliar a quantidade de comparações necessárias para realizar operações de inserção e busca. Isso nos ajuda a entender a eficiência dessas operações em termos de tempo de execução, uma vez que o número de comparações pode impactar diretamente o desempenho da estrutura de dados.

5 Gráfico de Comparações

Abaixo, apresentamos um gráfico que mostra o número de comparações realizadas em cada operação de inserção e busca ao longo de diversas chaves.

5.1 Interpretação do Gráfico

Observando o gráfico, percebemos que:

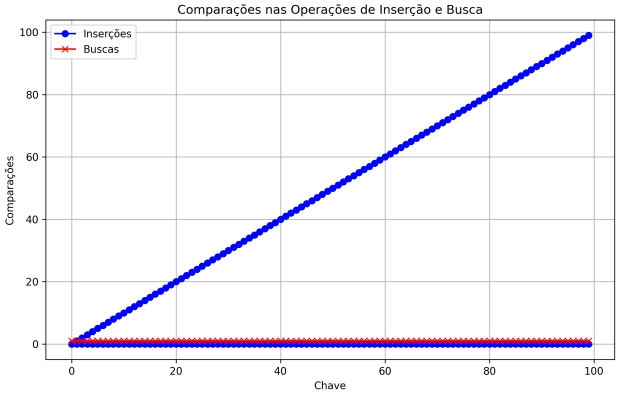

Figure 1: Comparações nas Operações de Inserção e Busca

- O número de comparações para **inserções** aumenta linearmente conforme o número de operações cresce, indicando uma maior complexidade conforme a tabela hash se preenche. Esse comportamento sugere que o número de comparações na inserção depende do estado atual da estrutura de dados (número de elementos já inseridos).
- Para **buscas**, o número de comparações permanece praticamente constante, o que sugere uma eficiência maior para esta operação, possivelmente devido ao uso de uma estratégia de busca que mantém o número de comparações baixo, mesmo com o aumento de chaves.

6 Análise de tabelas

Função Hash	Tempo Inserção	Colisões	Tempo Busca	Comparações
Multiplicação	215032900	0	5920	0
Divisão	93725400	0	5780	0
Dobramento	231163700	0	2660	0

Table 1: Tamanho da Tabela: 10 e Quantidade de Dados: 1000000

Função Hash	Tempo Inserção	Colisões	Tempo Busca	Comparações
Multiplicação	1103823900	0	5400	0
Divisão	340051000	0	200	0
Dobramento	780977700	0	320	0

Table 2: Tamanho da Tabela: 10 e Quantidade de Dados: 5000000

Função Hash	Tempo Inserção	Colisões	Tempo Busca	Comparações
Multiplicação	4188294600	0	380	0
Divisão	1299823200	0	160	0
Dobramento	3647690500	0	300	0

Table 3: Tamanho da Tabela: 10 e Quantidade de Dados: 20000000

Função Hash	Tempo Inserção	Colisões	Tempo Busca	Comparações
Multiplicação	165185300	0	240	0
Divisão	52023800	0	100	0
Dobramento	154320000	0	2380	0

Table 4: Tamanho da Tabela: 100 e Quantidade de Dados: 1000000

Função Hash	Tempo Inserção	Colisões	Tempo Busca	Comparações
Multiplicação	931130700	0	9100	0
Divisão	308104200	0	240	0
Dobramento	691001000	0	200	0

Table 5: Tamanho da Tabela: 100 e Quantidade de Dados: 5000000

7 Análise dos Resultados

Os resultados das tabelas mostram o desempenho de três funções hash diferentes (Multiplicação, Divisão e Dobramento) em termos de tempo de inserção, colisões, tempo de busca e comparações, para diferentes tamanhos de tabela e quantidades de dados.

7.1 Desempenho Geral

De maneira geral, a função hash de Divisão apresentou o melhor desempenho em termos de tempo de inserção e tempo de busca, independentemente do tamanho da tabela e da quantidade de dados. Isso pode ser observado pelos menores tempos de inserção e busca em todas as tabelas.

7.2 Colisões

Nenhuma das funções hash apresentou colisões nos testes realizados, o que indica que todas as funções foram eficazes em distribuir os dados uniformemente nas tabelas.

7.3 Comparações

O número de comparações foi zero para todas as funções hash em todos os cenários, sugerindo que as buscas foram realizadas de maneira eficiente, sem a necessidade de comparações adicionais.

7.4 Análise por Tamanho da Tabela e Quantidade de Dados

- **Tamanho da Tabela: 10** - Para 1.000.000 de dados, a função de Divisão teve o menor tempo de inserção (93.725.400) e busca (5.780). - Para 5.000.000 de dados, a função de Divisão novamente se destacou com o menor tempo de inserção (340.051.000) e busca (200). - Para 20.000.000 de dados, a função de Divisão manteve o melhor desempenho com o menor tempo de inserção (1.299.823.200) e busca (160).

- **Tamanho da Tabela: 100** - Para 1.000.000 de dados, a função de Divisão teve o menor tempo de inserção (52.023.800) e busca (100). - Para 5.000.000 de dados, a função de Divisão continuou a apresentar o menor tempo de inserção (308.104.200) e busca (240).

7.5 Conclusão

A função hash de Divisão demonstrou ser a mais eficiente em termos de tempo de inserção e busca, independentemente do tamanho da tabela e da quantidade de dados. As funções de Multiplicação e Dobramento, embora eficazes, apresentaram tempos de inserção e busca maiores em comparação com a função de Divisão.

Esses resultados sugerem que, para aplicações que requerem alta eficiência em inserções e buscas, a função hash de Divisão é a mais recomendada.