Relatório - Tabelas Hash em Java

1. Objetivo

O objetivo deste trabalho é implementar e avaliar o desempenho de diferentes estratégias de resolução de colisões em tabelas hash utilizando a linguagem Java, comparando as abordagens de encadeamento (chaining) e endereçamento aberto (rehashing). A análise envolve a medição e comparação de métricas como tempo médio de inserção e busca, quantidade de colisões, tamanho das listas encadeadas, gaps médios e máximos nas sondagens e, opcionalmente, o consumo de memória. Busca-se, ao final, identificar qual técnica apresenta o melhor equilíbrio entre desempenho, eficiência e distribuição dos dados em diferentes tamanhos de tabelas e conjuntos de entrada.

2. Estruturas e classes principais

O código desenvolvido foi organizado em classes independentes, de forma modular e coesa, visando a clareza da implementação e a facilidade de manutenção. Cada classe desempenha um papel específico dentro do sistema de experimentação e análise das tabelas hash:

- Registro - representa uma entrada da tabela hash, composta por um identificador numérico de 9 dígitos, utilizado como chave para as operações de inserção e busca.

- FuncaoHash - define uma interface genérica para implementação de diferentes funções hash, permitindo flexibilidade na escolha e substituição dos algoritmos de dispersão.

- HashUtils - contém as implementações das funções hash utilizadas no estudo, incluindo o hash multiplicativo de Knuth, além de métodos auxiliares para cálculo modular e normalização de valores.

- TabelaEncadeada - implementa a tabela hash com encadeamento, utilizando listas ligadas para o tratamento de colisões e registrando estatísticas como o tamanho máximo e médio das listas.

- TabelaEnderecAberto - representa a tabela hash com endereçamento aberto (rehashing), com suporte às estratégias de sondagem dupla e sondagem

quadrática, contabilizando colisões e gaps de sondagem.

- GeradorDados - responsável pela geração determinística dos conjuntos de dados (datasets), utilizando uma semente fixa (seed = 123456789L) para garantir reprodutibilidade dos experimentos.

- EscritorCSV - coleta e armazena todas as métricas de desempenho obtidas durante os testes, exportando-as para o arquivo resultados.csv para posterior análise e visualização gráfica.

Essa organização modular permite a execução isolada de cada componente, facilitando a comparação entre diferentes funções hash e estratégias de tratamento de colisões.

3. Configurações utilizadas no código

Para garantir a confiabilidade e a reprodutibilidade dos experimentos, foram definidas configurações fixas de tamanho de tabela, volume de dados e parâmetros das funções hash utilizadas.

- Tamanhos das tabelas hash (M):
  Foram escolhidos os valores 1009, 10007 e 100003, todos números primos, a fim de melhorar a dispersão dos índices e minimizar ciclos de endereçamento em operações de sondagem.

- Tamanhos dos conjuntos de dados (datasets):
  Foram gerados conjuntos contendo 100.000, 1.000.000 e 10.000.000 registros, permitindo avaliar o comportamento das estruturas sob diferentes volumes e fatores de carga.

- Seed do gerador aleatório:
  Utilizou-se uma semente fixa (123456789L) no gerador de números pseudoaleatórios, garantindo reprodutibilidade dos resultados em todas as execuções.

- Funções hash implementadas:

    1. Hash multiplicativo : utiliza o método de multiplicação com a constante, proporcionando boa dispersão com baixo custo computacional.

2. Hash duplo (Double Hashing): combinação de duas funções, reduzindo clustering e melhorando a distribuição.

3. Probing quadrático: utiliza equação com constantes, controlando o padrão de sondagem e limitando colisões primárias.

Essas configurações asseguram uma base experimental consistente, permitindo comparar o desempenho das diferentes estratégias de hashing sob condições controladas e variáveis de carga.

4. Justificativas das escolhas

As escolhas de funções hash, parâmetros e tamanhos das tabelas foram fundamentadas em princípios teóricos de dispersão uniforme e eficiência computacional:

- Função hash multiplicativa: foi adotada por apresentar baixo custo de processamento e boa uniformidade na distribuição das chaves, especialmente quando combinada com tamanhos de tabela primos. Esse método é amplamente recomendado pela literatura para aplicações de propósito geral.

- Hash duplo : escolhido por reduzir significativamente o problema de clustering, tanto primário quanto secundário, mantendo uma distribuição mais equilibrada dos elementos na tabela. Essa abordagem é especialmente eficiente em esquemas de endereçamento aberto.

- Probing quadrático: implementado como alternativa de sondagem que minimiza colisões primárias e evita sequências lineares de ocupação, melhorando o desempenho até fatores de carga intermediários.

- Tamanhos primos das tabelas: a utilização de números primos como tamanho da tabela evita ciclos de endereçamento e melhora a dispersão dos índices, reduzindo o risco de agrupamentos periódicos de chaves.

- Seed fixa : adotada para garantir reprodutibilidade dos experimentos, permitindo que todos os testes possam ser replicados de forma idêntica e os resultados comparados de maneira justa.

Essas decisões visam equilibrar desempenho computacional, estabilidade e consistência estatística, assegurando condições experimentais adequadas para a análise comparativa entre as abordagens de encadeamento e endereçamento aberto.

5. Metodologia de experimentação

O processo experimental foi estruturado de forma sistemática para permitir a comparação precisa entre as diferentes estratégias de tabelas hash. O programa foi desenvolvido para gerar automaticamente os conjuntos de dados (datasets) e executar operações de inserção e busca em todas as combinações possíveis de funções hash, tamanhos de tabela e tamanhos de dataset definidos.

Durante cada execução, o sistema coleta e registra métricas de desempenho essenciais, que servem de base para a análise comparativa:

- Tempo médio de inserção : mede o tempo necessário para inserir todos os elementos do conjunto na tabela hash, permitindo avaliar a eficiência da função hash e o impacto das colisões.

- Tempo médio de busca : avalia o tempo gasto para localizar elementos previamente inseridos, refletindo a eficácia da dispersão e a profundidade média de acesso.

- Número total de colisões: contabiliza quantas vezes duas ou mais chaves foram mapeadas para a mesma posição na tabela, sendo um indicador direto da qualidade da função hash.

- Tamanho das três maiores listas (encadeamento): aplicado às tabelas com encadeamento, indica o grau de concentração de chaves em determinados índices.

- Gap mínimo, máximo e médio (endereçamento aberto): representam a distância entre a posição ideal e a posição efetiva de armazenamento de cada chave, medindo a eficiência das sondagens e o efeito do clustering.

Todas as métricas coletadas são registradas em um arquivo CSV, possibilitando a geração de gráficos e análises comparativas posteriores. Essa metodologia garante consistência, reprodutibilidade e objetividade na avaliação dos resultados obtidos.