# Relatório: Tabelas de Dispersão

Daniel Campos da Silva

Felipe Gabriel Fraga Pinto

Thiago Monteles de Souza

Universidade Federal de Goiás - Instituto de Informática

## Introdução

Estruturas de dados podem ser descritas como meios de organização de dados para maior eficiência em determinadas operações. Enquanto árvores de busca binárias são, como o próprio nome diz, otimizadas para buscas, as tabelas de dispersão (ou, no inglês "*Hash Tables*") priorizam o acesso aos dados na memória.

Para isso temos como estrutura básica um vetor, ou seja, um segmento de memória particionada numa quantidade *M* de dados de mesmo tamanho em sequência. Assim, torna-se possível o acesso direto a um dado na posição *x* pela simples equação:

*endereço da posição x no vetor = endereço do dado na posição 0 do vetor + x*tamanho do dado*

Logo, como a operação para acessar um dado na tabela é dada por uma simples equação, temos que sua complexidade é *O(1)*, o que nos mostra a alta eficiência dessa estrutura.

A questão é: como determinar que a posição de um dado deve ser *x*? Em Listas Encadeadas e Árvores de Busca são utilizadas chaves para identificar cada dado (ou "nó", se preferir), porém eles eram alocados dinamicamente e posicionados de acordo com uma ordem relacionada a "comparação de menor e maior que", diferente do caso atual em que aloca-se um vetor com posições estáticas e bem definidas, todas dentro do intervalo fechado [0, M] (sendo M o tamanho da tabela). A solução está nas Funções de Dispersão (ou, no inglês "*Hashing Functions*"), que tratarão a chave original do dado para gerar uma nova dentro do intervalo descrito anteriormente. Não só isso, mas também há o interesse de fazer uma função que distribua as posições da forma mais uniforme possível, pois, segundo a Teoria da Probabilidade, no tópico de Distribuição Normal, valores aleatórios num determinado intervalo costumam se aglomerar num ponto próximo ao seu meio, e isso pode ocasionar um problema comum dessa estrutura: as colisões.

Uma colisão é definida como a tentativa de inserção de um dado numa posição já ocupada, assim, fica a cargo do programador definir qual solução será abordada para essa situação: deletar o dado presente e inserir o novo; deletar o novo; ou então preservar ambos e organizá-los de alguma forma.

Neste trabalho iremos abordar diferentes soluções para esse problema, denominando-as “Tratamento de Colisões”, e comparar funções de dispersão diferentes para estimar quais são as mais eficientes.

## Método

### Funções de Dispersão

Primeiro serão definidas e implementadas, para a construção de uma biblioteca, as seguintes funções de dispersão:

- Método da divisão: a chave gerada será o resto da chave original pelo tamanho da tabela.

- Método da divisão por primo: método da divisão, porém com o tamanho da tabela definido como o menor primo *p* maior ou igual ao tamanho da tabela.

- Método da dobra: somam-se os dígitos da chave original até a quantidade de dígitos ser equivalente à do tamanho da tabela. Como estamos considerando as chaves na base 10, o tamanho da tabela deve ser um potência de base 10.

- Método da multiplicação simples: a chave é multiplicada por uma constante entre 0 e 1, é descartada a parte inteira do resultado, e então multiplicamos ele pelo tamanho da tabela.

- Método da multiplicação quadrada: elevamos a chave ao quadrado, multiplicamos por uma constante entre 0 e 1, é descartada a parte inteira do resultado, e então multiplicamos ele pelo tamanho da tabela.

Em seguida, construiremos um programa para cada função de dispersão, em que será implementada uma função para ler os 10000 dados do arquivo de exemplo (enviado pelo professor), criar uma estrutura Pessoa para armazenar cada dado em memória, e então aplicar a função de dispersão para enfim inserir cada um na tabela. Guardaremos também as quantidades e em quais posições ocorreram colisões, para então comparar com as outras funções e definir qual é mais eficiente para evitar colisões (menor número total de colisões), além de qual distribuiu de

melhor os valores (menor diferença entre a quantidade de posições vazias e a maior quantidade de colisões ocorridas numa única posição).

**Tratamentos de Colisão**

Implementamos conforme orientado pelas instruções do trabalho os três métodos de tratamento de colisão:

- Método de Encadeamento Externo (considerando o 1º registro): Cada primeiro registro é um membro da tabela hash e é um ponteiro para uma lista encadeada de membros que tem o mesmo valor de hash.

- Método de Encadeamento Externo (nenhum registo considerado): Cada posição da tabela hash é um ponteiro para uma lista encadeada independente de primeiro registro.

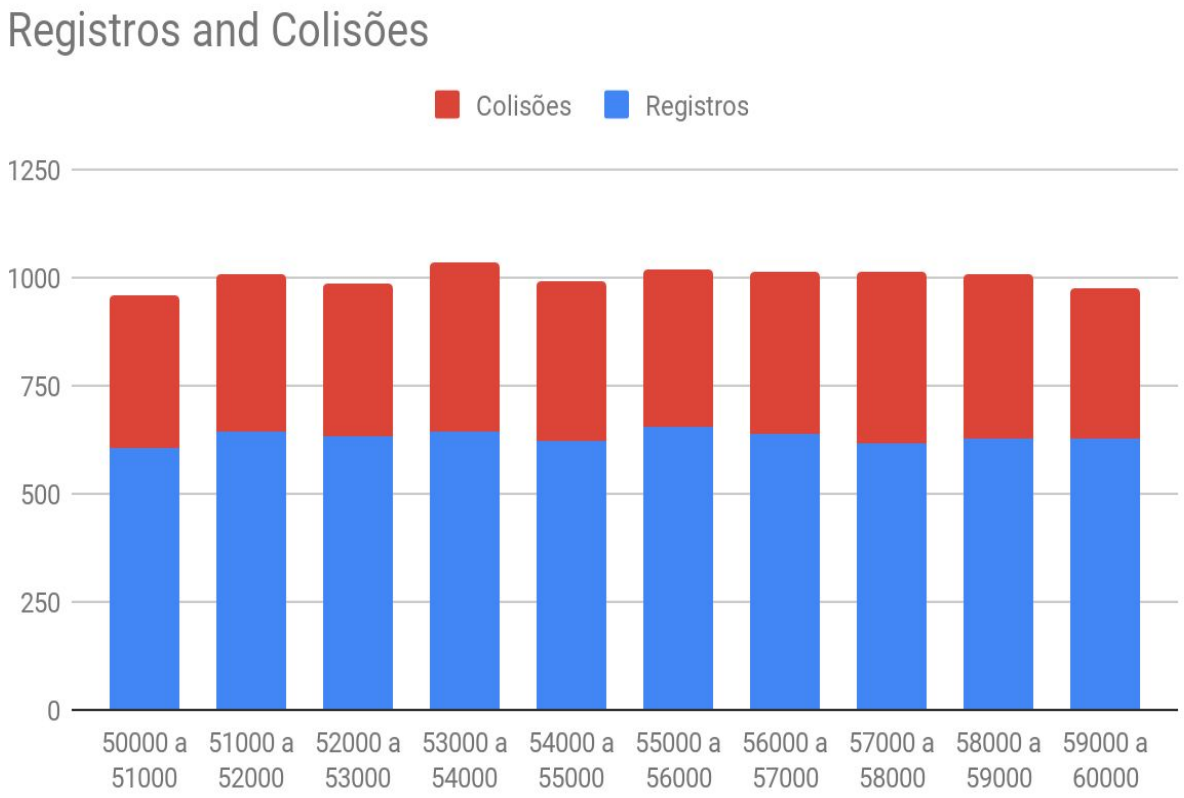
- Método da Zona de Colisão: Parte do arquivo é destinado a tabela hash e o restante é destinado a zona de colisão, essa que por sua vez pode ser vista como uma lista encadeada ou grande vetor onde operamos linearmente para buscar, deletar e adicionar.

Usamos o método da dobra para criar o Hash em todos os métodos e no tópico a seguir temos os nossos resultados.

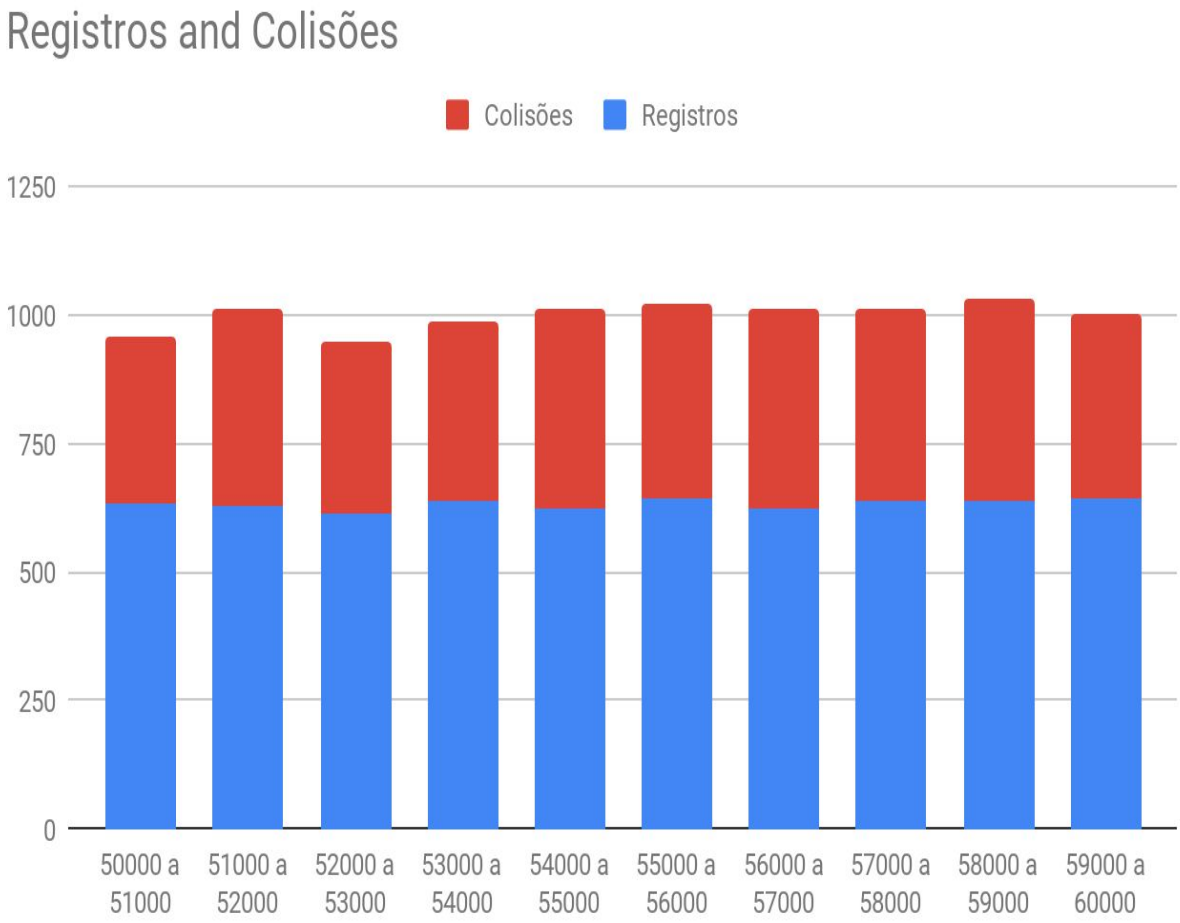
**Resultados**

Após os testes obtivemos os seguintes resultados:

- Usando o método de Hash Dobra contabilizando apenas os primeiros registros e não inserindo as colisões (colisões foram contabilizadas apenas para fins comparativos.).
    - ➔ Foram processados 10.000 registros sendo 6296 registrados e 3704 colisões.

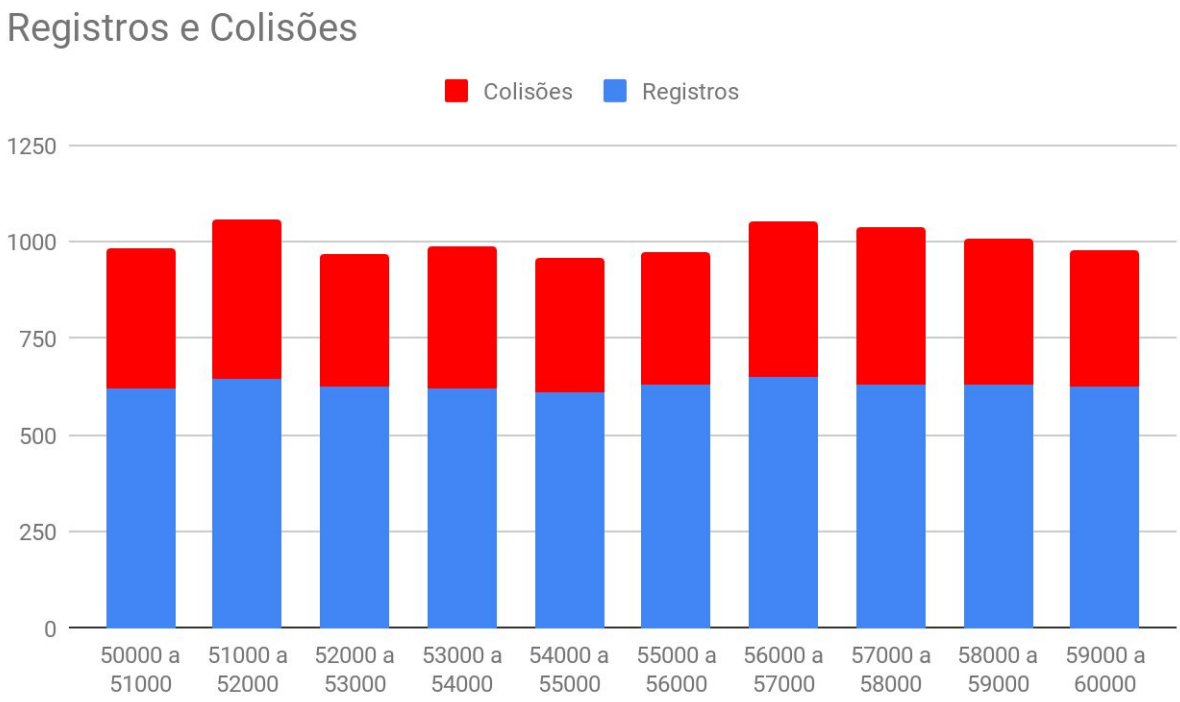


- Usando o método do resto considerando o tamanho padrão da tabela:
  - ➔ Foram processados 10.000 registros sendo 6.321 registrados e 3.679 colisões.



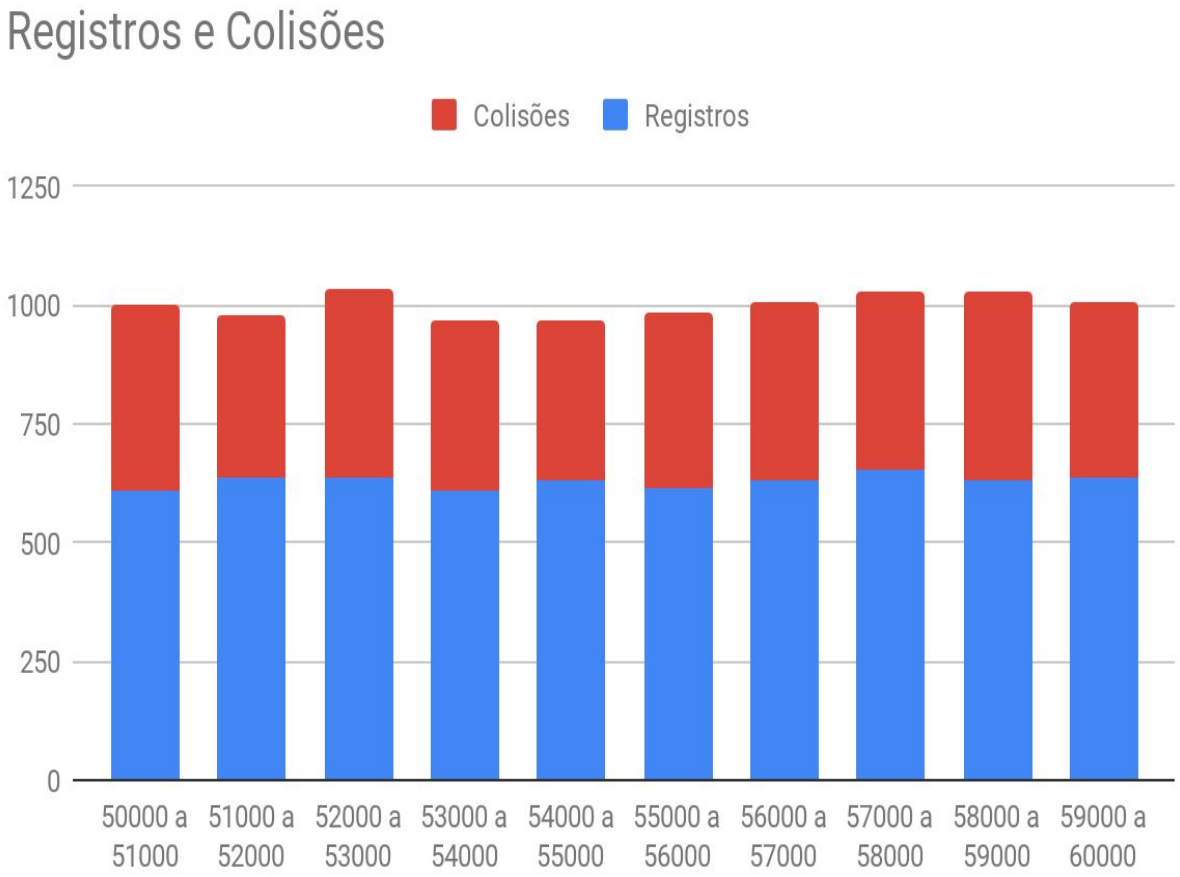
- Usando o método do tamanho baseado em um número primo e fazer o rehash:

  ➔ Foram processados 10.007 registros (primeiro número primo maior que 10.000) sendo 6.298 registrados e 3702 colisões.



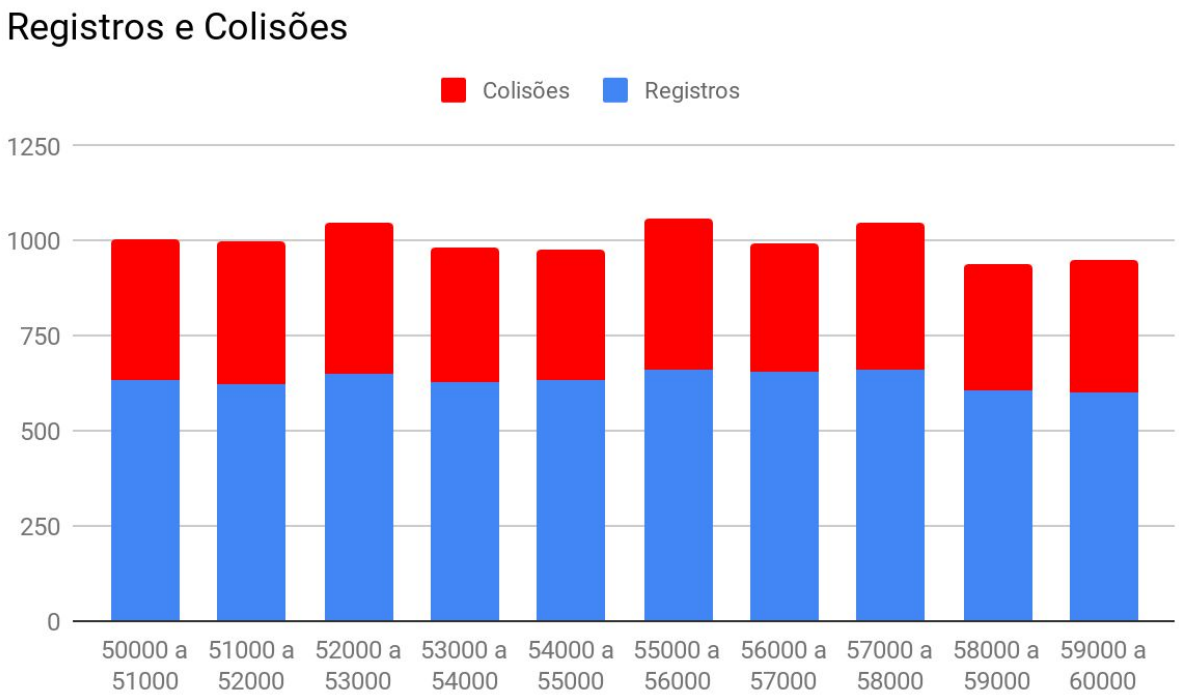
- Usando o método da multiplicação simples:

  ➔ Foram processados 10.000 registros sendo 6293 registrados e 3707 colisões.



- Usando o método da multiplicação quadrática:

  ➔ Forem processados 10.000 registros sendo 6354 registrados e 3646 colisões.




- Os resultados utilizando os métodos de colisão foram

  1. Método de Encadeamento Externo (considerando o 1º registro)
     a. A média de operações é 1,344;
  2. Método de Encadeamento Externo (nenhum registo considerado)
     a. A média de operações é 2,338;
  3. Método da Zona de Colisão
     a. A média de operações é 965,867981;

## Referências

DISTRIBUIÇÃO Normal. **Instituto de Matemática e Estatística | IME-USP - Instituto de Matemática e Estatística**, 2012. Disponível em: <https://www.ime.usp.br/~hbolfar/aula_2013/Aula6-A12012.pdf>. Acesso em: 12 de junho de 2019.

IVAN L. M. RICARTE. **DCA | FEEC - Faculdade de Engenharia Elétrica e de Computação - UNICAMP**, 2003. Página de texto. Disponível em: <http://www.dca.fee.unicamp.br/cursos/EA876/apostila/HTML/node26.html>. Acesso em: 12 de junho de 2019.

BOERES, Cristina. Hashing (Tabela de Dispersão). **Instituto de Computação - UFF**, sem ano. Disponível em: <http://www2.ic.uff.br/~boeres/slides_ed/ed_TabelaHash.pdf>. Acesso em: 12 de junho de 2019.

SOUZA, Jairo Francisco de. Hashing. **UFJF - Universidade Federal de Juiz de Fora**, 2012. Disponível em: <http://www.ufjf.br/jairo_souza/files/2012/11/4-Hashing-Fun%C3%A7%C3%B5es.pdf>. Acesso em: 12 de junho de 2019.

Soares, Fabrizzio UFG - Universidade Federal de Goiás, Instituto de Informática <https://github.com/Professor-Fabrizzio/AED2/tree/master/Trabalho1>. Acesso em: Quase todos os dias de produção do trabalho.