RELATÓRIO: Implementa-o-de-uma-Tabela-de-Espalhamento-Tabela-Hash

Introdução

Tem como objetivo criar um sistema que guarda vários nomes em uma tabela hash, essa deve ter 53 colunas que devem armazenar 100788 nomes diferentes. Para evitar colisão e também para possibilitar algum tipo de organização, as colunas foram criadas como listas encadeadas duplas.

Desenvolvimento

Primeiramente a criação da tabela hash, para tal se optou por criá-la como uma lista encadeada dupla de tamanho 53 com cada coluna tendo uma chave de 0 a 52 mantendo a possibilidade de mudar o tamanho se desejado.

O tratamento de colisão não é necessário já que estamos lidando com lista encadeada, porém seria possível implementar um método que manteria as tabelas mais uniformes porém esse destruiria a possibilidade de busca rápida e precisa.

A maior parte do trabalho relacionado a lista encadeada foi reutilizado. Para a leitura dos dados, utilizou-se o que o C disponibiliza “fopen” e “fgets”.

Para o hashing, foi feito alguns testes com números primos que multiplicam todos os caracteres e depois tirado o módulo de 53. Assim no final sobrando um número entre 0 e 52 que será usado como chave para saber em qual coluna da tabela o nome deverá ser colocado.

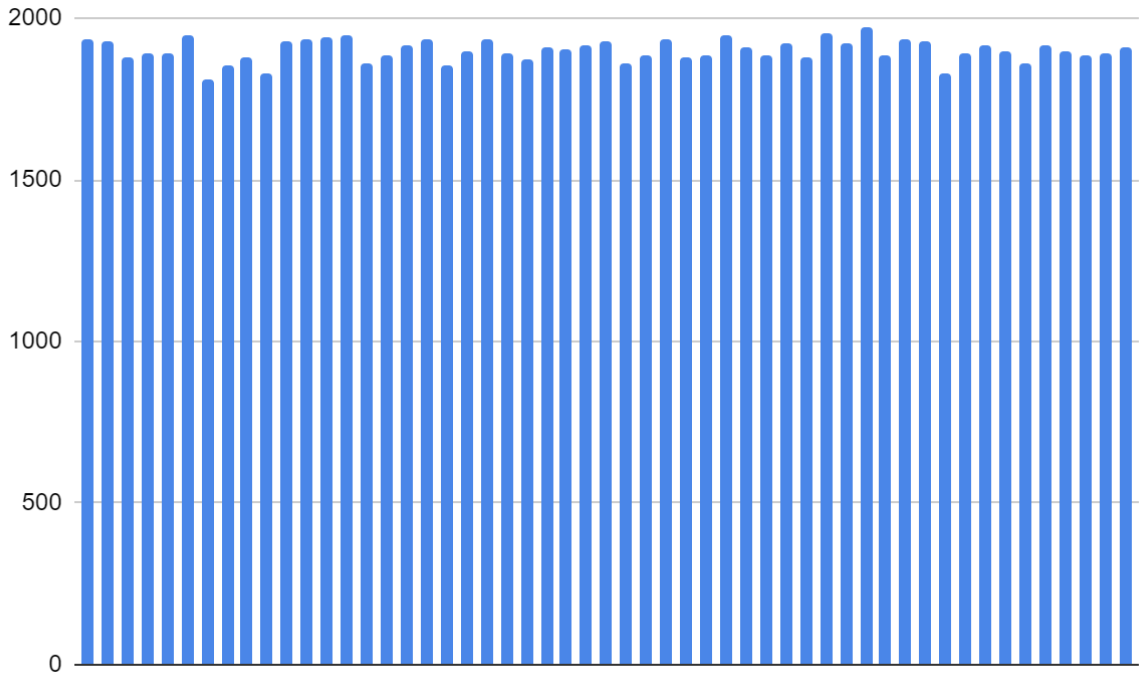
Os testes sempre foram visando o mais próximo do hashing uniforme, O hashing uniforme acaba sendo inalcançável, sendo que sempre existe a possibilidade de novos valores serem adicionados e acabaram quebrando esse hashing.

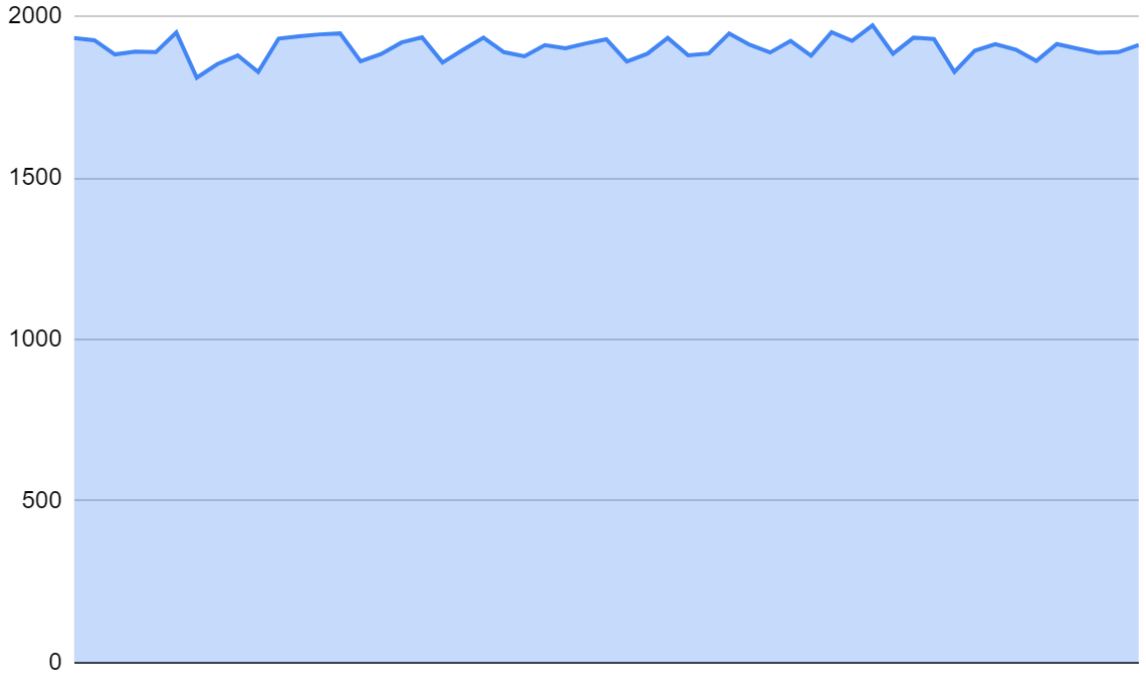
Resultado dos testes:

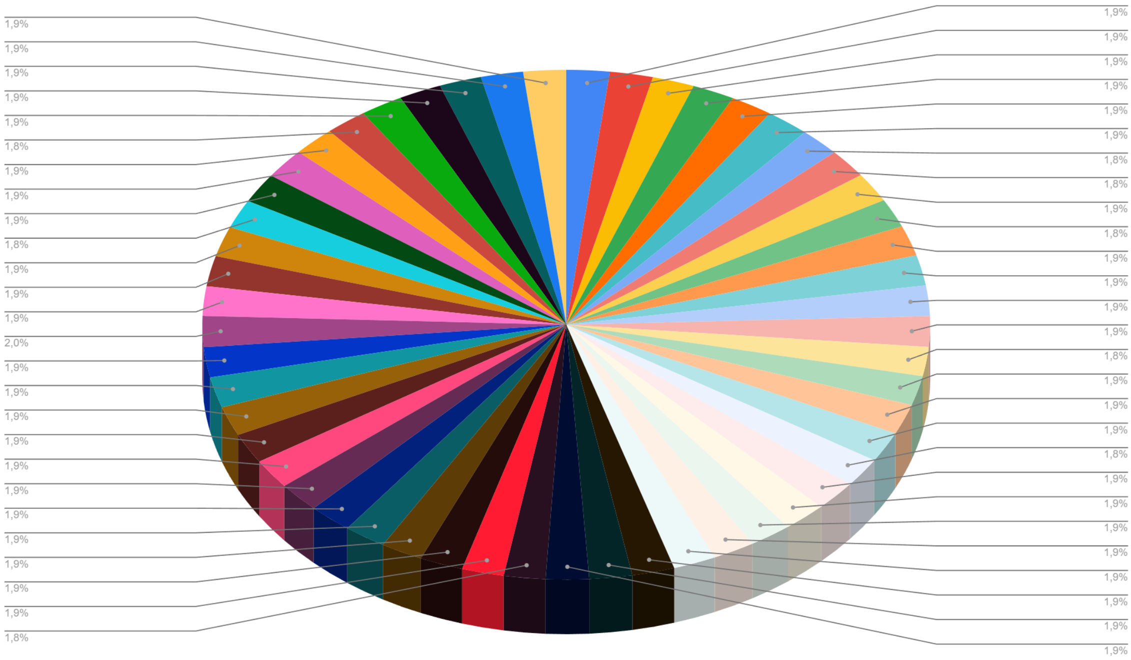
n	max %	min %	diff %	frequência
31	1,968488312	1,772036354	0,1964519586	1,110862262
61	1,985355399	1,74524745	0,2401079494	1,137578169
113	1,95459777	1,791879986	0,1627177839	1,090808416
127	1,970472675	1,796840894	0,1736317816	1,096631695
149	1,957574314	1,795848712	0,1617256023	1,090055249
181	1,97940231	1,815692344	0,1637099655	1,090163934
467	1,956582133	1,795848712	0,1607334206	1,089502762
367	1,988331944	1,809739255	0,1785926896	1,098684211

Observando a diferença entre o maior e o menor, notou-se que o número 467 tem uma boa frequência (1972/1810 = 1,089).

Alguns gráficos que ajudam nessa análise, cada coluna/fatia representa uma chave de 0 a 52.







Sistema de organização da tabela foi o mais trabalhoso e mais complexo, para criá-lo inicialmente se visava uma quicksort de Hoare, porém devido ao alto nível de complexidade que esse tomou devido ao fato da dificuldade em achar o pivô no centro da partição e várias verificações para evitar um endereço nulo.

Por causa de tantos problemas gerados pelo modelo equivalente ao de Hoare, utilizou um modelo em que o último elemento é o pivô até a primeira partição, isso causa a possibilidade do pior caso ser igual a um bubble sort.

Além da função básica de adicionar mais nomes, também é possível remover nomes e buscar tais nomes, uma restrição que a tabela coloca, só é possível encontrar a coluna que contém o nome desejado se inserir todos os caracteres do nome.

Conclusão

Tabela hash é um ótimo método para organizar itens de grande variedade e ajuda na busca de tais, porém quando há falta de informações a busca se torna horrível, tendo que buscar de forma linear em todas as colunas.