Strings of Pearls
Seminario feito a partir da 15° coluna do livro progamming pearls

Felipe Stein
Arthur da Matta
2026

Sumário

1 Problema a ser resolvido

O problema central abordado por Bentley não é a dificuldade técnica de manipular textos, mas sim a ineficiência sistêmica causada pela escolha de estruturas de dados inadequadas para o volume e o objetivo do processamento. O autor argumenta que, no desenvolvimento de software moderno, existe uma tendência perigosa de tratar "strings"como objetos abstratos e simples, ignorando o custo computacional de operações em larga escala.

2 Contagem e Frequência

O objetivo aqui é ler um documento e contar quantas vezes cada palavra aparece.

2.1 Abordagem STL (C++)

O autor começa usando **set** e **map** da biblioteca padrão. É elegante e funciona, mas pode ser lento para arquivos gigantescos como a Bíblia.

2.2 Aprimoramento com Hash Table

Para ganhar velocidade, ele implementa sua própria Tabela Hash em C.

2.3 Resultado

A implementação customizada em C chegou a ser **uma ordem de magnitude (10x) mais rápida** que as bibliotecas padrão, processando milhares de palavras em menos de um segundo.

2.4 Lição

Árvores de busca (usadas no STL) mantêm tudo ordenado, mas se você só quer contar, o Hashing é imbatível.

3 O Sufixo e a Repetição

Aqui o problema muda: "Qual é a maior frase que se repete em um livro?"(Ex: No livro Ilíada, de Homero).

3.1 O Problema da Força Bruta

Comparar todas as combinações de frases levaria um tempo absurdo ($O(n^2)$).

3.2 A Solução: Suffix Array (Vetor de Sufixos)

Esta é a **"pérola"do capítulo**. Em vez de copiar o texto, criamos um array de ponteiros onde cada ponteiro aponta para uma posição do texto original (cada posição é o início de um "sufixo").

3.3 A Técnica

Você ordena esses ponteiros alfabeticamente. Ao ordenar, frases repetidas ficam vizinhas uma da outra no array. Basta comparar os vizinhos para achar a maior repetição.

3.4 Eficiência

Isso transforma um problema impossível em algo que roda em segundos ($O(n \log n)$).

4 Geração de Texto e Cadeias de Markov

Ao usar cálculos aproximados, lembre-se do famoso conselho de Einstein.

4.1 O Conceito

O autor usa **Cadeias de Markov**. A ideia é: dada uma sequência de k palavras, qual a probabilidade da próxima palavra aparecer?

4.1.1 Implementação

Ele usa a mesma estrutura de **Suffix Array** da parte anterior para encontrar rapidamente todas as ocorrências de uma frase e escolher a próxima palavra aleatoriamente entre as opções reais do texto original.

4.2 Níveis de Ordem

4.2.1 Ordem-1

Escolhe a próxima letra/palavra baseada apenas na anterior (gera um texto sem sentido).

4.2.2 Ordem-2

Ele olha as duas últimas palavras para decidir a terceira.

4.2.3 Ordem-3

Ele olha as três últimas para decidir a quarta.

5 Referências

Bentley, Jon (2000). Programming Pearls. 2ª ed. Addison-Wesley. isbn: 9780201657883.