UNIVERSIDADE FEDERAL DE MATO GROSSO
Campus Universitário do Araguaia
Instituto de Ciências Exatas e da Terra – ICET
Curso de Bacharelado em Ciência da Computação

Relatório de análise do DataSet de Músicas para a disciplina de Estrutura de Dados 1

Discentes: Fernanda Lima de Souza, Guilherme da Silva Ferraz

Docente: Dr. Ivairton Monteiro Santos

Barra do Garças - MT

2024

SUMÁRIO

Introdução

O trabalho "Análise do DataSet de Músicas" consiste numa análise comparativa de diferentes estruturas de dados implementadas em linguagem C para processamento de um dataset de letras de músicas entre os anos de 1950 e 2019, contendo aproximadamente 28 mil registros de letras de músicas. Este dataset contém 31 colunas de dados, incluindo informações como "danceability" e "genre".

Essa análise comparativa visa não apenas compreender a eficácia de diferentes estruturas de dados na manipulação desse vasto conjunto de informações, mas também aprimorar a compreensão sobre como as escolhas das estruturas de dados podem influenciar significativamente o desempenho e a eficiência de algoritmos de processamento de dados. Ao explorar diferentes abordagens de implementação, buscamos identificar as vantagens e desvantagens de cada estrutura em relação ao tempo de execução, uso da memória e facilidade de manipulação de dados.

Análise do Dataset de Músicas

O trabalho é estruturado em três partes fundamentais, cada uma delas desempenhando um papel crucial no cumprimento dos objetivos do trabalho:

Parte 1 – Carregamento da base de dados:

O processo de carregamento da base de dados é uma etapa crucial em qualquer projeto de análise de dados. Neste contexto, utilizamos um arquivo CSV que é lido completamente em memória, dessa forma, a estrutura escolhida foi a Lista Duplamente Encadeada (LDE). Essa escolha foi motivada pela eficiência na inserção de dados em qualquer posição da lista. Isso é particularmente útil, considerando que a ordem dos registros no dataset não é conhecida previamente, e é necessário inseri-los na ordem em que são lidos. Por isso, a forma de inserção dos registros levou conta a ordem original dos dados nos dataset e a inserção no início mostrou-se mais útil para este projeto – levando em conta que os ID's estão em ordem crescente dentro do dataset, a inserção no início mostrou-se ser a mais correta.

Este trecho do código lê cada tupla do arquivo CSV e processa-os individualmente. A cada processamento, os dados de cada coluna – separados por vírgula - são armazenados em uma estrutura de dados (Struct) chamada music_dataset que os guarda nos campos correspondentes a cada coluna nessa estrutura. A função sscanf é empregada para analisar cada tupla e extrair os dados de suas colunas, os quais são inseridos em seus respectivos campos no music_dataset. Além disso, o tempo de abertura e carregamento dos dados é medido utilizando as funções da biblioteca padrão do C para manipulação de tempo e data, time.h.

Parte 2 – Processamento dos dados:

Nesta etapa, é realizada uma análise em todas as tuplas da base de dados e os registros musicais com "danceability" superior ou igual a 0,5 são inseridos em novas estruturas de dados – Lista Circular Duplamente Encadeada (LCDE) e Lista Encadeada (LE) – e posteriormente, com os dados já nessas estruturas, é realizada uma contabilização dos registros com "genre" igual a "pop". Assim como o tempo de execução do carregamento de dados é calculado utilizando as funções da biblioteca padrão do C para manipulação de tempo e data, time.h, o tempo de execução desta segunda etapa do trabalho também é medido da mesma forma.

A escolha da LCDE para essa segunda parte do trabalho levou em consideração a eficiência dessa estrutura na manipulação contínua dos dados em um ciclo, eficiência na inserção dos

dados e a facilidade/flexibilidade de acesso de um determinado elemento de qualquer ponto da lista, tornando-a uma boa escolha para a análise de dados, assim como a equivalência a referência dada, a Lista Encadeada, já que as funções de Inserção das duas estruturas são muito semelhantes.

Parte 3 – Análise:

Durante a execução do código é medida a duração das partes 1 (Carregamento do dataset na LDE) e 2 (Análise das tuplas da base de dados e contabilização dos “genre” igual a “pop”, bem como o tempo total de execução. Além disso, o consumo total de memória utilizado pelas estruturas de dados é medido. Por fim, são apresentados os totais de letras musicais com "danceability" igual ou maior que 0,5 e gênero "pop".

Dados obtidos:

• Tempo:

- Tempo da Parte 1 = 0.130000 segundos/minutos
- Tempo da Parte 2 = 0.150000 segundos/minutos
- Tempo total = 0.28 segundos/minutos

• Memória:

- Memória total para o contexto 1 (Lista encadeada) = 0,00029496 Mbytes
- Memória total para o contexto 2 (Lista circular duplamente encadeada) = 0,000295092 MBytes

• Contabilidade:

- Total de letras musicais com “danceability” igual ou maior que 0,5 = 16557
- Total de letras musicais com “danceability” (>= 0,5) e gênero pop = 4043

A ferramenta utilizada para a análise de consumo da memória foi o Valgrind. A suíte necessita ser instalada em sua máquina, dito isso, é compilado normalmente o programa, logo após, usando a ferramenta com a opção $ valgrind --leak-check=full. */nomedoprog*, então, *produzirá* uma saída detalhada no terminal, indicando possíveis vazamentos de memória, alocações de memória.

Para executar o:

- Compila normal o arquivo:

  $ gcc estruturas_de_dados.c main.c -o <nome_arquivo>

  $ valgrind --leak-check=full ./<nome_arquivo>

Ademais, foi desenvolvido um arquivo MakeFile. O emprego de tal arquivo proporciona diversas vantagens no contexto do desenvolvimento de trabalho estabelecido, tais como automação do processo de compilação, gerenciamento eficiente de dependências entre os diferentes componentes do projeto, flexibilidade na configuração das tarefas de compilação, garantia de reprodução em diferentes ambientes e plataformas, facilitação da manutenção do código fonte e sua integração harmoniosa com outras ferramentas de desenvolvimento disponíveis.

No Makefile criado, as especificações, variáveis e regras utilizadas nos arquivos são:

Figura 1 – Arquivo Make File

- **CC** é definido como gcc, o compilador C padrão.
- **OPTIMIZATION_FLAG**: Define opções de otimização para o compilador.

- **all**: define o alvo padrão "all" que compila o executável "dataset”.
- **dataset:** compila o executável "dataset" a partir dos arquivos objeto e arquivos fonte.
- **estruturas_de_dados.o**: compila o arquivo objeto para as estruturas de dados.
- **main.o**: compila o arquivo objeto para o programa principal.
- **valgrind**: Executa o programa "dataset" com Valgrind para detectar vazamentos de memória.
- **clean**: Remove os arquivos objeto e o executável "dataset".

Os resultados obtidos revelam informações importantes sobre o dataset analisado, fornecendo dados como tempo de execução, consumo de memória e contabilidade dos registros processados. Essas informações são cruciais para compreender a eficiência das estruturas de dados utilizadas e podem guiar futuras análises.

Conclusão

Em conclusão, as análises comparativas das estruturas de dados usadas para processar os registros de músicas entre os anos 1950 e 2019 mostrou uma compreensão mais profunda de como a escolha da estrutura de dados pode impactar significativamente o desempenho dos algoritmos de processamento de dados.

Durante este trabalho, avaliamos a adequação e eficiência de várias abordagens, incluindo listas encadeadas, listas duplamente encadeadas e listas circulares duplamente encadeadas, em diferentes contextos e ao lidar com um grande volume de dados.

Por fim, a escolha de qual estrutura de dados se adequa ao projeto deve levar em conta a quantidade de dados a serem processados, o uso de memória, os requisitos do projeto – Por exemplo, o tipo de dados que irá ser armazenado e as operações a serem realizadas, a flexibilidade, a ordem dos dados a serem processados e a complexidade do projeto.