Escola de Engenharia

Universidade do Minho

Laboratórios de Informática III

2023-2024

2ª Fase

Elaborado por:

Adélio José Ferreira Fernandes – a78778

Leonardo Gomes Alves – a104093

Salvador Duarte Magalhães Barreto – a104520

Índice

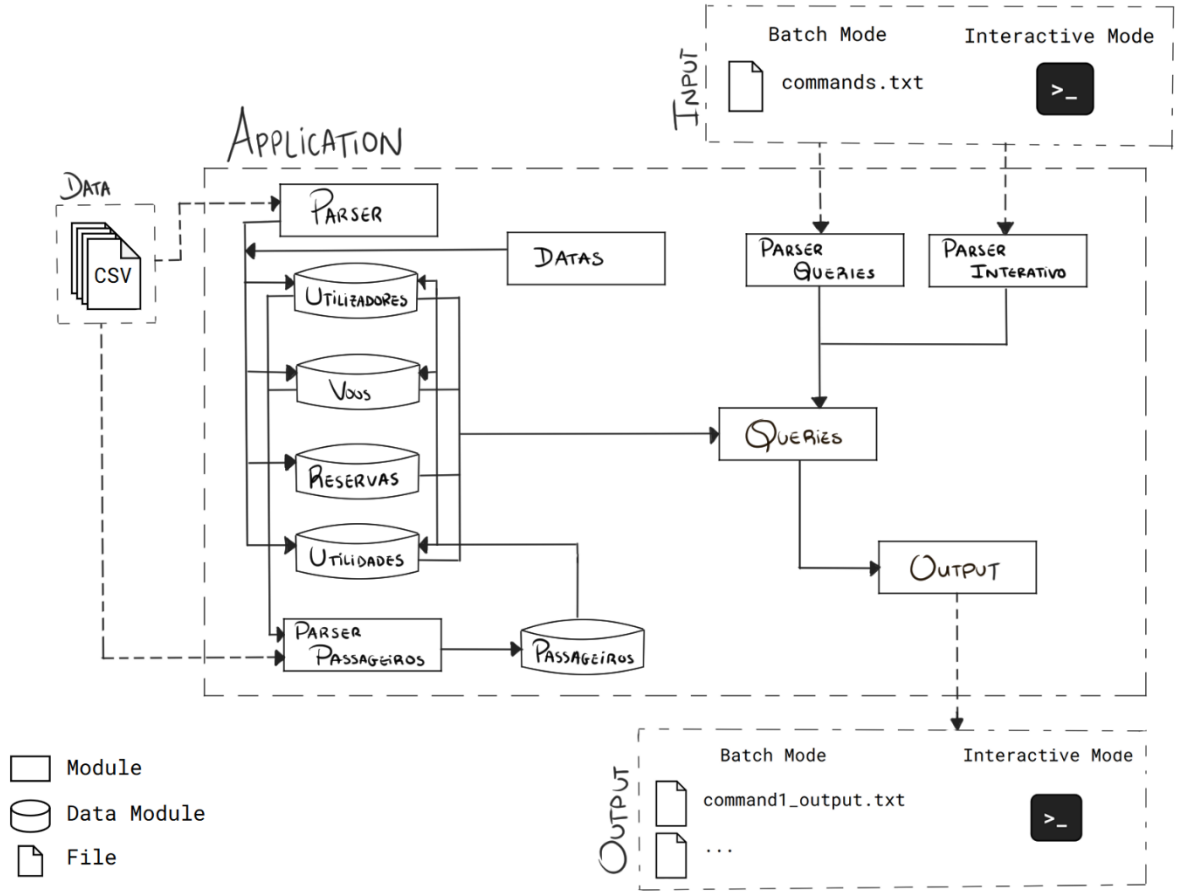
Introdução

Nesta segunda fase, foi nos imposto um dataset maior, que tornou as resoluções das queries mais desafiantes e com abordagens diferentes, tendo em conta o tempo de execução e o tamanho que o nosso programa ocupa.

Tivemos ainda que dividir o programa por módulos, e tratar do encapsulamento que, apesar de não ser difícil, é sempre trabalhoso, visto que temos que fazer funções de set e get, e aplicá-las no nosso código.

Por fim, fizemos o modo interativo para o nosso programa, utilizando a biblioteca do *ncurses*, e também o programa de testes que compara os outputs das queries com os resultados reais dessa certa query.

Arquitetura da Aplicação



1. Tempo de execução e memória

Com a entrada do *large dataset* no site de testes, o nosso programa ocupava mais de 5 GB para executar, e como tal, não era possível.

Com uma análise geral no código, verificámos que as nossas estruturas auxiliares, que utilizávamos para responder às queries, gastavam bastante memória, devido ao uso excessivo de **strdup**, que foi facilmente resolvido guardando apenas o apontador para onde os dados ficam guardados no parser, pois estes dados nunca seriam alterados.

Outro método que utilizámos para reduzir memória foi mudar nas estruturas de dados de char* para char[], pois desta forma não seria alocado espaço dinamicamente para uma string e poupar-nos-ia espaço.

No entanto, já diminuído o espaço, um dos problemas era o tempo de execução, que também era superior ao máximo permitido no site testes.

Utilizando a ferramente *gprof*, verificámos que grande parte dos nossos problemas eram relacionados com ordenação de arrays. Em alguns casos o problema era o tipo de algoritmo utilizado, em que utilizávamos o *Bubble-Sort*, e passámos a utilizar o Quick Sort, que tem uma complexidade de tempo muito inferior. Noutros casos, o problema era a reordenação de arrays já ordenados, que foi rapidamente resolvido ao retirar os que estavam a mais.

2. Modularização e encapsulamento

Um dos problemas do nosso código relativo à fase 1 do projeto, é a inexistência de encapsulamento em todos os módulos. No entanto, devido à utilização das nossas estruturas de dados (hashtables e arrays), foi simples aplicar funções de get e set. Conforme fomos verificando o código, fomos criando as funções necessárias para ser possível manter o encapsulamento e a abstração do tipo de estruturas utilizadas que eram partilhadas nos diferentes módulos.

No âmbito da modularização, já tínhamos dividido praticamente tudo na primeira fase e apenas tivemos que mudar algumas funções de sítio, pois já não funcionavam nesse módulo devido ao encapsulamento. Criámos ainda mais módulos para melhorar a modularização e também pela necessidade nomeadamente do modo interativo.

3. Queries

- Query 5

A query 5 é responsável por listar os voos com origem num dado aeroporto entre duas datas, ordenados por data de partida ou por ordem alfabética do identificador do voo. Para tal, decidimos criar uma nova estrutura de aeroportos que guarda todos os voos que passam por um dado aeroporto. Desta forma, a query 5 é responsável por ordenar os voos (utilizando o **Quick Sort**) do dado aeroporto no input.

- Query 6

A query 6 é responsável por listar o top N aeroportos com mais passageiros num dado ano. Voltámos a utilizar a estrutura dos aeroportos, que agora tem um parâmetro que conta o número de passageiros que passam por cada aeroporto dado um certo ano. Desta forma, dado o ano, o array de aeroportos é ordenado por ordem decrescente de passageiros e é chamada a função que imprime os resultados.

- Query 8

A query 8 é responsável por apresentar a receita total de um dado hotel entre duas datas. Para tal, temos a estrutura dos hotéis, que guarda todas as reservas que nele acontecem. Assim, ordenámos o array das reservas por data e basta verificarmos se uma dada reserva se encontra na data estipulada do input.

- Query 10

A query 10 é responsável por apresentar várias métricas do programa. Desta forma, à medida que os dados são processados no parsing, as estruturas de estatística são atualizadas conforme a data de ocorrência. Inicialmente, a forma que utilizamos para calcular a métrica de passageiros únicos por dia era bastante ineficiente, devido ao elevado número de estruturas em cada dia. Assim, decidimos utilizar estruturas que já estavam definidas para a utilização noutras queries.

4. Modo interativo

O modo interativo é responsável por fornecer uma interface gráfica ao programa, em que é possível ao utilizador do programa, escolher o seu input.

Desta forma, utilizamos a biblioteca do *ncurses.* Para o funcionamento do programa, é pedido ao utilizador para inserir o diretório de onde se situa o dataset a ser utilizado e, posto isto, o utilizador tem acesso ao menu principal. Este menu é constituído por um menu de exemplos, o menu das queries e um botão de sair do programa.

O menu de exemplos é um menu interativo, em que o utilizador escolhe qual query deseja ver, e este é responsável por mostrar os outputs esperados de uma dada query, tanto com, ou sem a opção de field.

O menu das queries fornece ao utilizador a opção de escrever a query que deseja ser executada pelo programa. Após executada a query, o programa mostra o seu output no terminal e é possível navegar pelas diversas páginas (caso existam), utilizando o teclado.

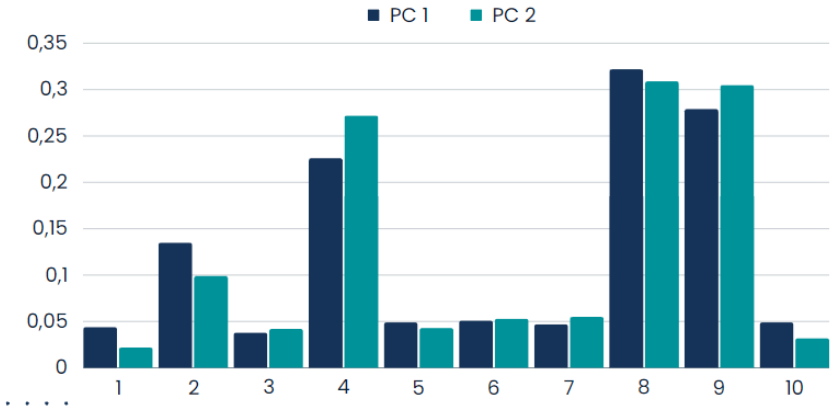
5. Análise de desempenho

	Computador 1	Computador 2
CPU	AMD Ryzen 7 5800X3D 3.4 GHz	Intel I5-1135G7 2.4 GHz
RAM	32 GB DDR4 3200 MHz	8 GB 2133 MHz
GPU	GTX 1660 6GB	Mesa Intel Xe Graphic (Integrada)

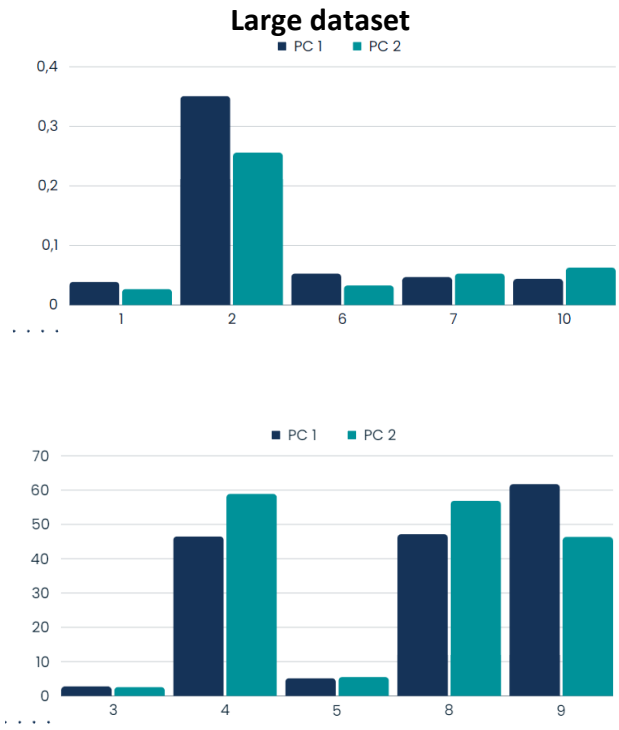
De modo a medir a performance de cada query, decidimos medir o tempo de execução de cada uma, tanto para o regular dataset, como para o large dataset. Para isso, utilizamos 5 queries de cada tipo, executando o programa 5 vezes, e fizemos a sua média.

O eixo das ordenadas corresponde ao tempo de execução, sendo medido em milissegundos, e o eixo das abcissas corresponde ao número da query.

Regular dataset



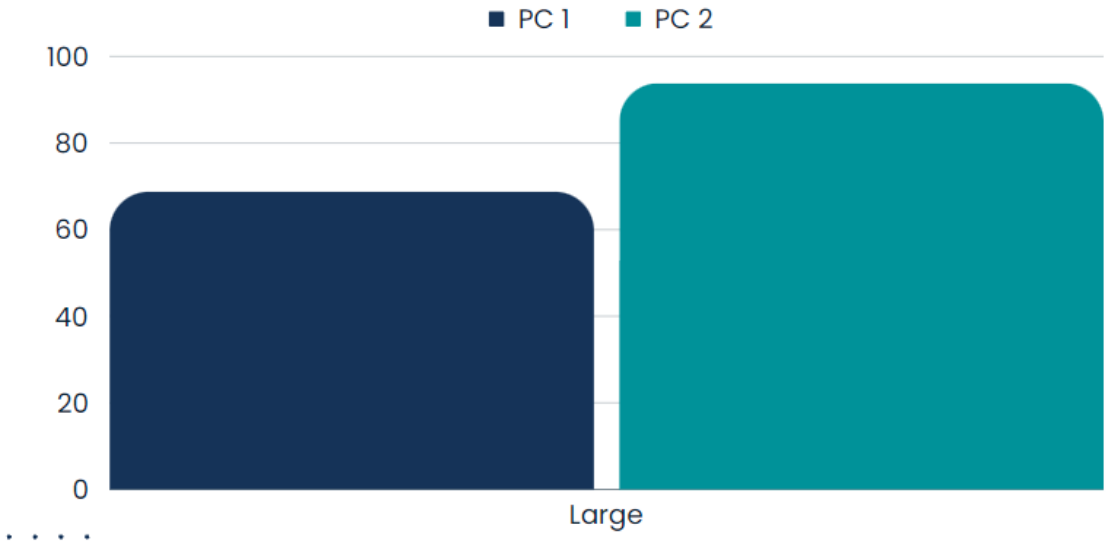
Ainda que o computador 1 seja substancialmente melhor, podemos observar que o computador 2 obtém uns tempos muito melhores no processamento das queries.



No large dataset, já podemos observar que o computador 1 executa mais rápido nalgumas das queries, o que pode ter a ver com a sua capacidade de processamento de maior número de dados.

No entanto, uma das grandes diferenças entre os computadores, é que o computador 1 consegue executar o programa num tempo muito menor que o computador 2. Tal fato, pode estar relacionado com a velocidade do processador e

com a quantidade de cache. É de realçar agora que os valores se encontram em segundos.



Ainda que a diferença no regular dataset não seja muita, (computador 1 – 0.241s; computador 2 – 0.266s), a diferença é notória no large dataset, com um total de 25,08s (computador 1 – 68.73s; computador 2 – 93.81s).

Assim, podemos concluir que, de um modo geral, o tempo de execução do programa é bastante influenciável pelo hardware da máquina e, até podemos comparar os tempos da plataforma de sites com os destas duas máquinas e veremos que a diferença de tempos é ainda mais notável.

6. Conclusão

Com este projeto, conseguimos aprofundar os nossos conhecimentos em C e nomeadamente na manipulação de datasets de elevado tamanho.

Comparativamente à 1ª fase, conseguimos melhorar diversos aspetos do nosso programa, como a utilização de memória e a sua gestão, resolvemos os erros notificados pelo **Valgrind** , ainda que estes não aparecessem na compilação, e diminuímos substancialmente o tempo de execução.

No entanto, surgiu-nos uma dificuldade relativa à query 10 que, foi responsável por aumentar o tempo de execução e a utilização de memória, devido ao parâmetro dos passageiros únicos por dia. Mesmo assim, conseguimos resolver a query por completo.

Por fim, achamos que concluímos os objetivos essenciais deste trabalho e aprendemos a dominar um novo tipo de estruturas que não estávamos familiarizados, que serão certamente úteis para o nosso futuro.