Projeto: Matrizes Esparsas

Antônio Joabe Alves Morais
Iarley Natã Lopes Souza

1 Descrição do Problema

Uma matriz esparsa é uma matriz que tem mais valores nulos do que não nulos, ou seja, mais "zerados"do que "setados", ao contrário da matriz densa.

Quanto as aplicações da matriz esparsa, pode-se citar: [1, 3, 4]

- Machine Learning;
- Codificação de dados (Data Encoding);
- Otimização de algoritmos;
- Sistemas computacionais baseados em IA;
- Entre várias outras aplicações.

Para representar uma matriz esparsa na programação, diversas técnicas e estruturas de dados podem ser aplicadas, nesse caso, no entanto, usaremos listas simplesmente encadeadas circulares.

2 Decisões Tomadas

Não fizemos muitas coisas que ficam fora do que foi proposto no documento inicial, o que implementamos foram 3 funções extras na `SparseMatrix.cpp`:

- `void getHead`:
  retorna o atributo `head`, declarado na `SparseMatrix.hpp`;
- `void getLineQty`:
  retorna o número de linhas da matriz;
- `void getColQty`:
  retorna o número de colunas da matriz.

Também optamos por implementar uma `main()` interativa, devidamente documentada na seção 5.

3 Divisão

A divisão do projeto foi sendo decidida no decorrer do projeto:

Joabe ficou responsável pelas funções `SparseMatrix()`(construtor), `insert()`, `print()` e `readSparseMatrix()` [2, 6]; pela análise assintótica; e pela escrita deste documento [5].

Iarley ficou responsável pelas funções `~SparseMatrix()`(destrutor), `get()`, `sum()` e `multiply()`; e pela `main()` interativa.

4 Dificuldades

Dentre as dificuldades que enfrentamos estão: saber implementar cada função da matriz; pensar em cada possibilidade de erro que uma certa implementação pode gerar; saber como ligar cada nó em diferentes situações; e comunicar de forma concisa e organizada cada caso e conceito, o que pode ser bem confuso.

5 Testes Executados

Como dito na seção 2, foi implementado uma `main()` interativa, que será documentada a seguir:

Comandos da Main Interativa	
`create m n`	Cria uma matriz com m linhas e n colunas
`insert i j x a`	Insere um valor x na posição (i, j) na matriz a
`get i j a`	Retorna um valor inserido na posição (i, j) na matriz a
`show a`	Mostra no terminal a matriz a
`showAll`	Mostra no terminal todas as matrizes existentes
`sum a b`	Soma as matrizes a e b e cria uma matriz resultado
`mult a b`	Multiplica as matrizes a e b e cria uma matriz resultado
`read s`*	Lê um arquivo com nome s (uma string, incluindo .txt) e cria uma matriz com os dados desse arquivo.
`exit`	Desaloca todas as matrizes e encerra o programa

*Quanto aos testes envolvendo arquivos, já existe um arquivo preenchido na pasta de projeto, chamado *A.txt*. Se o usuário quiser, ele pode modificá-lo e/ou até criar um novo arquivo, só tendo o cuidado de passar o nome correto ao chamar o comando `read`.

6 Análise de Complexidade

Nessa análise na notação *Big O*, vamos focar somente nas complexidades não constantes ($O(n), O(n^2)...$), pois as linhas com complexidade constante não influenciam no resultado final, além de serem numerosas no programa.

- `get()`:

  Na linha 185, temos um laço `while`, que percorre a lista até encontrar a linha passada por parâmetro. $O(n)$.

  Depois, na linha 190, um laço `while` percorre a lista até encontrar a coluna. $O(n)$.

  $$O(n) + O(n) = 2 \times O(n)$$

  Ignorando as constantes, temos $O(n)$.

- `insert()`:

  Nessa função, o pior caso vai acontecer quando for preciso desalocar um nó, que demanda percorrer a lista várias vezes para ajustar ponteiros e, finalmente, deletar o nó.

  Primeiramente, temos o primeiro $O(n)$, na linha 94, um laço `for` usado para encontrar a linha desejada.

  Depois vamos para a linha 100, que faz chamada da função `get()`, já analisada anteriormente. $O(n)$.

  Ao entrar nessa condição, vamos para o `else`, da linha 106, já que queremos desalocar, a fim de obter o pior caso.

  Temos na linha 109 um laço while que acha o nó a ser desalocado. $O(n)$.

  Na linha 113, temos um laço `while`, que percorre a matrix até encontrar o nó anterior ao nó a ser desalocado. $O(n)$.

  Na linha 124, um laço `for` encontra a coluna passado por parâmetro. $O(n)$.

  E o laço `while` da linha 129 é análogo ao da linha 113. $O(n)$.

  $$O(n) + O(n) + O(n) + O(n) + O(n) + O(n) = 6 \times O(n)$$

  Portanto, temos $O(n)$.

- `sum()`:

  Nessa função, ocorre um aninhamento de complexidades não constantes. Por isso, essa análise será um pouco diferente das anteriores.

  Vamos começar do "núcleo"do primeiro while (linha 103): a linha 106.

  Nela, ocorre a chamada da função `get()`: $O(n)$, dentro da função `insert()`: $O(n)$. O que dá a essa linha uma complexidade $O(n^2)$.

  A linha 106 é executada n vezes, pois está inserida num laço `while`, o que eleva o nível de complexidade para $O(n^3)$.

  E tudo isso está dentro de outro laço `while`, que é executado n vezes, o que eleva a complexidade para $O(n^4)$.

  Depois a linha 113 faz um processo semelhante ao descrito anterior, executando a soma em si, ao contrário do bloco anterior, que executa uma cópia. $O(n^4)$.

  $$O(n^4) + O(n^4) = 2 \times O(n^4)$$

  Portanto, temos $O(n^4)$.

Referências

[1] Jason Brownlee. *A Gentle Introduction to Sparse Matrices for Machine Learning*. Acessado em: 07/11/2022. 2018. URL: https://machinelearningmastery.com/sparse-matrices-for-machine-learning/.

[2] Bruno P. Campos. *Curso de C++ #51 - Operações com arquivos (ifstream) - Parte 2*. Acessado em: 04/11/2022. 2017. URL: https://www.youtube.com/watch?v=Tczymt0OkYo&ab_channel=CFBCursos.

[3] Universidade Virtual do Estado de São Paulo. *Estrutura de Dados - Aula 14 - Matriz esparsa*. Accessed: 24/10/2022. 2016. URL: https://www.youtube.com/watch?v=C_ePgrEbLs0&t=689s&ab_channel=UNIVESP.

[4] Argonne National Laboratory. *Argonne National Laboratory Deploys Cerebras CS-1, the World's Fastest Artificial Intelligence Computer*. Acessado em: 07/11/2022. 2019. URL: https://www.anl.gov/articles/argonne-national-laboratory-deploys-cerebras-cs-1-worlds-fastest-artificial-intelligence-computer.

[5] Overleaf. *Learn LaTeX in 30 minutes - Overleaf, Online LaTeX Editor*. Accessed: 04/11/2022. 2022. URL: https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes.

[6] Shmeowlex. *C++ File Input and Output*. Acessado em: 04/11/2022. 2021. URL: https://www.youtube.com/watch?v=LIxwbvKRFAg&t=354s&ab_channel=Shmeowlex.