Matriz Esparsa

Uma matriz esparsa é uma matriz em que nem todos os elementos estão realmente presentes ou são necessários. Portanto, nesse caso, a maioria das posições são preenchidas com zero e ocupam espaço na memória mesmo sendo nulas.

Pode-se economizar espaço significativo de memória se apenas os termos diferentes de zero forem armazenados e manuseados, de forma que a eficiência computacional será maior.

Uma matriz é considerada esparsa quando:

- Suas dimensões são relativamente grandes (1000 x 1000, por exemplo)
- Nem todas suas posições são usadas (o número de valores não nulos é da ordem do número de linhas ou de colunas, ou seja, de 1 milhão de posições usam-se cerca de 1000).

Aplicações:

- Em problemas de engenharia, física, eletrônica (método das malhas para resolução de circuitos elétricos)
- Em computação : armazenamento de dados – planilhas eletrônicas, busca de caminhos no google maps

As operações usuais sobre essas matrizes (somar, multiplicar, inverter, transpor) também podem ser feitas em tempo muito menor se não são armazenadas as posições que contém zeros.

Matrizes esparsas : representação

	1	2	3	4	5	6
1	0	0	0	**-3**	0	0
2	0	**1**	0	**2**	0	0
3	0	0	**3**	0	0	0
4	0	**4**	0	**5**	0	0
5	0	0	**7**	0	**1**	0
6	0	**-1**	0	0	0	0

É necessário usar uma representação que evite o armazenamento de tantas posições nulas

Ao invés de representar todos os valores, armazenam-se números de linha e coluna dos elementos com valor não nulo:

Para a matriz acima, armazenam-se 25 inteiros (linha, coluna, valor) ao invés de 36 inteiros

- { (1,4,-3), (2,2,1),(2,4,2),(3,3,3), (4,2,4), (4,4,5),(5,3,7),(5,5,1), (6,2,-1) }

Há quatro técnicas para representar uma matriz esparsa:

- Lista ligada
- Árvore binária
- Matriz de ponteiros
- Fragmentação ou *hashing*

Essas técnicas são realmente úteis para matrizes com uma porcentagem significativa de elementos nulos; se este não for o caso, o melhor é usar o método de armazenamento comum

A representação que usaremos neste projeto é a matriz esparsa com listas ligadas cruzadas.

Cada coluna e cada linha da matriz serão representadas por uma lista ligada circular com um nó cabeça.

Cada nó da estrutura, além dos nós cabeça, representará as células diferentes de zero da matriz e podem ter o seguinte tipo:

```
class Celula
{
  Celula direita;  // ponteiro para a o próximo elemento não nulo na mesma linha
  Celula abaixo;  // ponteiro para a o próximo elemento não nulo na mesma coluna
  int linha, coluna;
  double valor;
}
```

Cada nó é colocado na lista com os elementos inseridos em ordem baseada no índice da matriz, ou seja, o elemento A_{12} aponta com o atributo direita o elemento A_{13} na mesma linha.

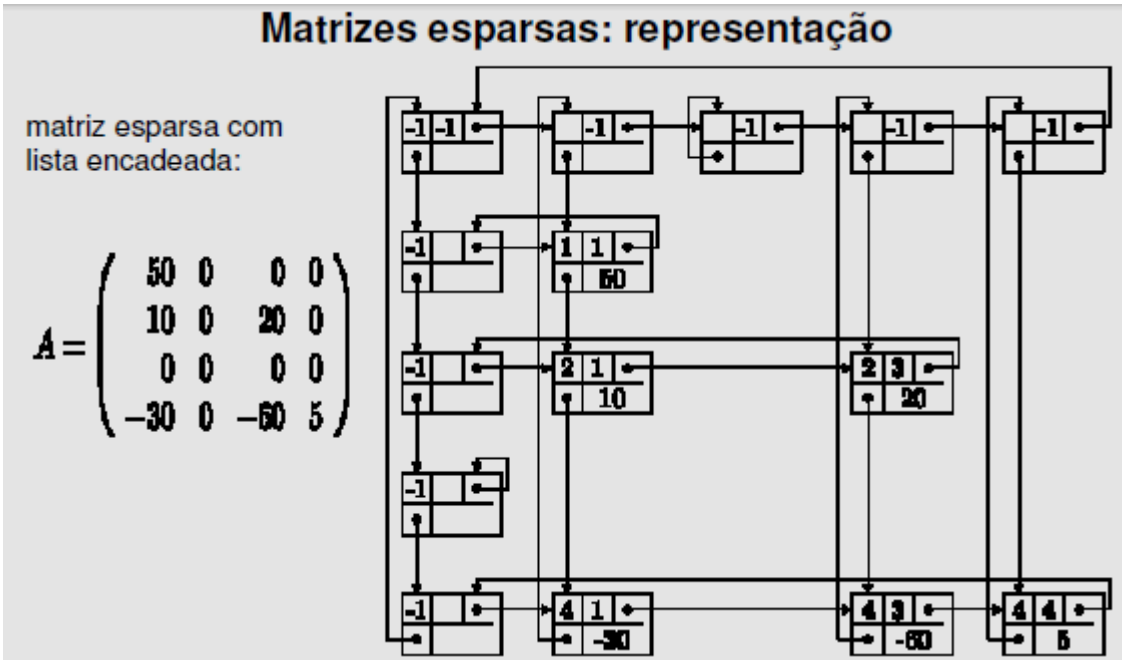
O atributo abaixo deve ser usado para apontar o próximo elemento diferente de zero na mesma coluna.

O atributo direita deve ser usado para apontar o próximo elemento diferente de zero na mesma linha.

Dada uma matriz A, para um valor Aij diferente de zero, deverá haver um nó Celula com:

- o atributo valor contendo o elemento Aij
- o atributo linha contendo i
- o atributo coluna contendo j
- esta Celula deverá pertencer à lista circular da linha i e também deverá pertencer à lista circular da coluna j
- cada Celula pertencerá a duas listas ao mesmo tempo.

Para diferencias as células cabeça, -1 é colocado nos atributos linha e coluna dessas células.

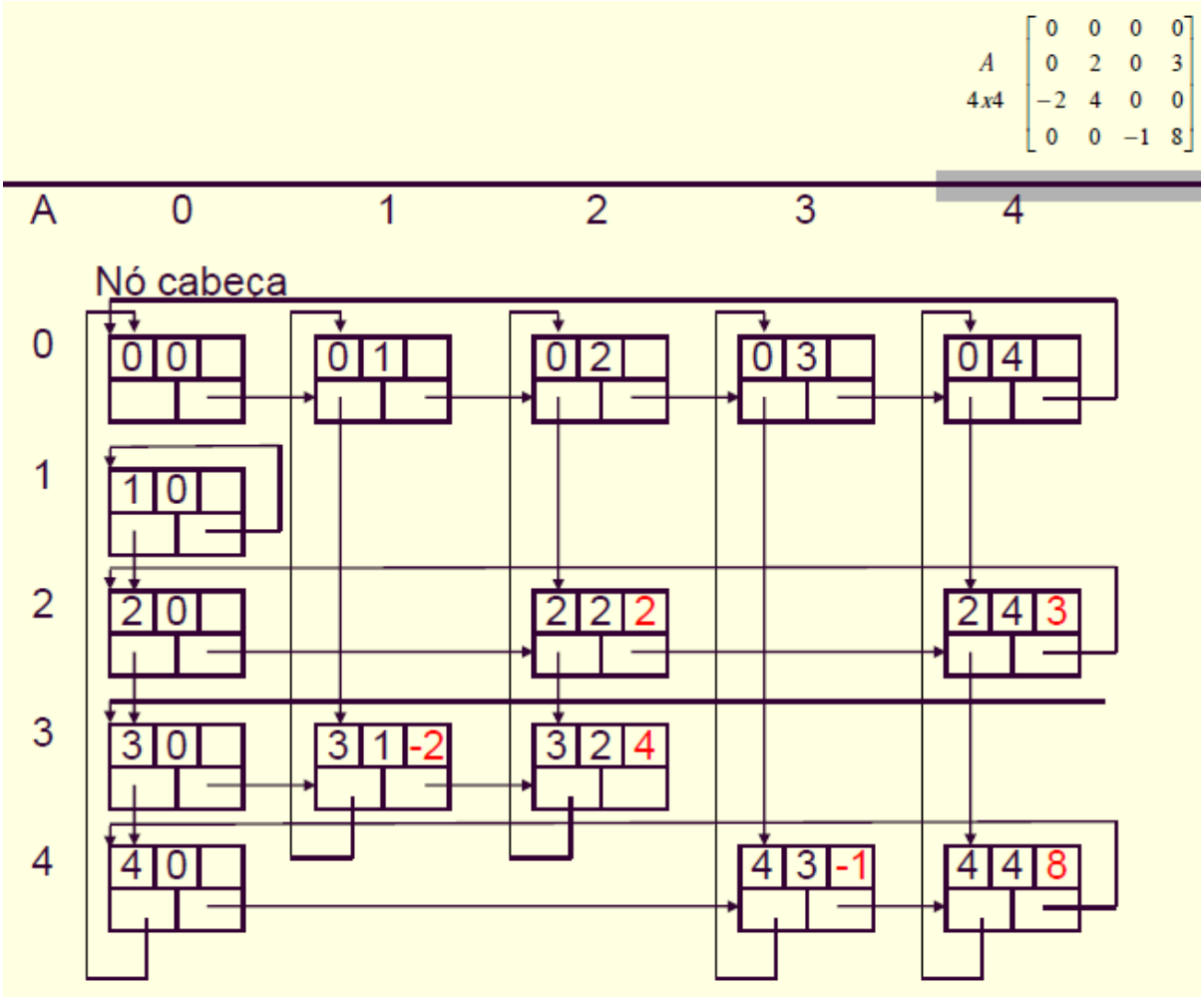


Além de armazenar matrizes esparsas, as aplicações normalmente exigem a realização de operações sobre essas matrizes como, por exemplo:

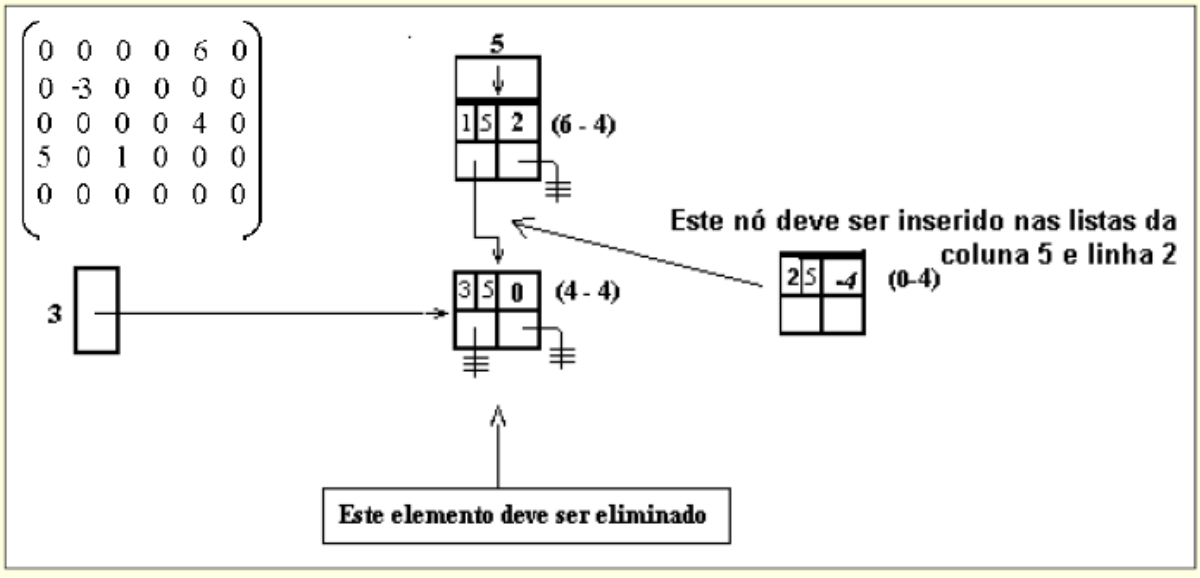
- montar matriz esparsa a partir de dados em um arquivo texto ou teclado
- acessar um elemento na matriz, devolvendo seu conteúdo
- multiplicar uma dada linha ou coluna por uma constante
- somar uma constante a todos os elementos de uma linha ou coluna
- somas duas matrizes esparsas de igual dimensão
- multiplicar matrizes compatíveis
- inverter ou transpor matrizes esparsas

Como consequência dessas operações, alguns elementos podem deixar de ser nulos, enquanto que outros podem se tornar nulos. Portanto, devemos lembrar que:

- quando um elemento da matriz deixar de ser nulo, há inserção na estrutura
- quando um elemento da matriz se torna nulo, há sua remoção da estrutura.



- Por exemplo, ao se somar -4 a coluna 5 do exemplo



a entrada para estes procedimentos consiste dos valores de *m* e *n* (número de linhas e colunas da matriz) seguidos de triplas (*i, j, valor*) para os elementos diferentes de zero da matriz

- para a matriz do exemplo, a entrada seria:
  - 4 4
  - 1 1 50.0
  - 2 1 10.0
  - 2 3 20.0
  - 4 1 -30.0
  - 4 3 -60.0
  - 4 4 5

$$A = \begin{pmatrix} 50 & 0 & 0 & 0 \\ 10 & 0 & 20 & 0 \\ 0 & 0 & 0 & 0 \\ -30 & 0 & -60 & 5 \end{pmatrix}$$

Usando Visual Studio e C#, implemente as classes Lista Ligada Cruzada e Celula conforme descrito acima. Crie um programa Windows Forms em C#, para, para realizar operações sobre matrizes esparsas, armazenada na lista ligada cruzada. O programa deve ter as seguintes operações:

- Criar a estrutura básica da matriz esparsa com dimensão M x N
- Inserir um novo elemento em uma posição (l, c) da matriz
- Ler um arquivo texto com as coordenadas e os valores não nulos, armazenando-os na matriz
- Retornar o valor de uma posição (l, c) da matriz
- Exibir a matriz na tela, em um gridView
- Liberar todas as posições alocadas da matriz, incluindo sua estrutura básica
- Remover o elemento (l,c) da matriz
- Somar a constante K a todos os elementos da coluna c da matriz
- Somar duas matrizes esparsas, cada uma representada em uma estrutura própria e ambas exibidas em seu próprio gridView. O resultado deve gerar uma nova estrutura de matriz esparsa e exibido em um gridview próprio
- Multiplicar duas matrizes esparsas, cada uma representada em uma estrutura própria e ambas exibidas em seu próprio gridView. O resultado deve gerar uma nova estrutura de matriz esparsa e exibido em um gridview próprio.

IMPORTANTE

- Trabalho feito **em dupla;**
- Desenvolver em C# no Visual Studio – não serão aceitos programas feitos em outros ambientes de desenvolvimento nem em outra linguagem;
- Comentar adequadamente o programa e o código programado;
- Nomear os identificadores de forma adequada;
- No início dos arquivos fonte, digitar comentário com os RAs e nomes dos alunos;
- **Relatório de desenvolvimento** deve ser feito num arquivo cujo nome é: RA1_RA2_RelatorioProjeto1ED.PDF (exemplo: 18101_18192_RelatorioProjeto1ED.pdf);
- O relatório **deve** ser entregue em formato PDF;
- Entrega : **11/04/2019**, pela área da disciplina no AVA Google Classroom;
- Material a ser entregue: arquivos **do projeto e PDF compactados em um único arquivo, cujo nome será** RA1_RA2_Projeto1ED.rar (18101_18192_Projeto1ED.rar, por exemplo).

Fontes de consulta:

http://www2.inatel.br/docentes/rosanna/downloads/c421-c-20061-s186383-1/65-me-s972142-1

https://pt.scribd.com/document/337947689/Matrizes-esparsas-RAFR