TDA MATRIZ DISPERSA

Ing. Mario Milton López Winnipeg

Capitulo 6 TDA MatrizDispersa

6 Matriz Dispersa

6.1 Descripción del TDA **Matriz Dispersa.**

6.2 Especificación del TDA **Matriz Dispersa**

6.3 Ejemplos de uso.

6.4 Implementaciones del TDA **Matriz Dispersa.**

6.4.1 Implementación con vectores.

6.4.2 Implementación con apuntadores

6.1 Descripción del TDA Matriz Dispersa

- Se define matriz dispersa a aquella matriz que tiene suficientes valores repetidos de forma que vale tener en cuenta esto.
  - Evitando operaciones sobre elementos repetidos
  - Ahorrando tiempo de computacion
  - No guardando valores repetidos

6.2 Especificación del TDA Matriz dispersa

Especificación Informal

`MatrizDispersa` = TDA con operaciones `crea`, poner, elemento, dimension_fila, fimension_columna, dimensionar, definir_valor_repetido

- **DESCRIPCIÓN:**
  - Los valores del TDA `Matriz Dispersa` son Elementos La maatriz dispersa es mutable cuando se pone , dimensiona o definir valor por defecto.

- **OPERACIONES:**
  - `crea()` `devuelve (M:Matriz Dispersa)`
  - `Efecto` : Devuelve la matriz de dimension 0x0 .

6.2 Especificación del TDA MatrizDispersa

- `Dimensionar(M: MatrizDispersa, df,dc:entero)`
  - efecto: Devuelve la matriz dimensionada de dfxdc

- `DimensionFila(M:MatrizDispersa)devuelve (Dimension fila de la matriz)`
  - requerimientos: Matriz creada.
  - efecto: Devuelve valor que indica el numero de filas

- `DimensionColumna(M:MatrizDispersa)devuelve (Dimension Columna de la matriz)`
  - requerimientos: Matriz creada.
  - efecto: Devuelve valor que indica el numero de columnas

6.2 Especificación del TDA MatrizDispersa

- `Poner( f, c : indice, valor: elemento)`
  - requerimientos: Matriz creada y Dimensionada
  - efecto: Modifica matriz ubicando el valor en la celda f,c

- `Elemento(f,c : indice )`
  - requerimientos: Matriz Creada y Dimensionada
  - efecto: Retorna el elemento de la celda f,c

- `Definir_valor_repetido( valor: elemento)`
  - requerimientos: Matriz creada
  - efecto: Llena toda la matriz con el valor repetido

6.2 Especificación del TDA MatrizDispersa

Especificación Formal

Tipo: `MatrizDispersa (Elemento)`

- *Sintaxis:*
- `crea → MatrizDispersa`
- `Dimensionar(MatrizDispersa,df,dc) → MatrizDispersa`
- `Dimension_Fila(MatrizDispersa)→ Numero`
- `Dimension_columna(MatrizDispersa)→ Numero`
- `Poner(f,c:indice ;valor : elemento)→MatrizDispersa`
- `Elemento(matrizdispersa;f,c:indice)→ Elemento`
- `Definir_valor_repetido(valor: elemento)`

`.....`

6.2 Especificación del TDA MatrizDispersa

La interface del TDA MatrizDispersa de acuerdo a esta especificación puede definirse de la siguiente forma:

```
publico interface MatrizDispersa
{

completar la presente lamina


}  // fin interface MatrizDispersa
```

6.3 Ejemplo de uso

```
publico entero SumaElemento( M: MatrizDispersa)
 inicio
     suma=0
     para cada i = 1 hasta M.dimension_Fila
        para cada j=1 hasta m.dimension_columna
        inicio
            suma=suma + m.elemento(i,j)
         fin
     retornar suma
  fin
```

6.4 Implementaciones del TDA MatrizDispersa

- En esta sección mostraremos tres implementaciones para el TDA MatrizDispersa:
  - Implementación con vectores
  - Implementación con apuntadores

Formato Coordenado

$$A = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 \\ 3 & 4 & 0 & 5 & 0 \\ 6 & 0 & 7 & 8 & 9 \\ 0 & 0 & 10 & 11 & 0 \\ 0 & 0 & 0 & 0 & 12 \end{bmatrix}$$

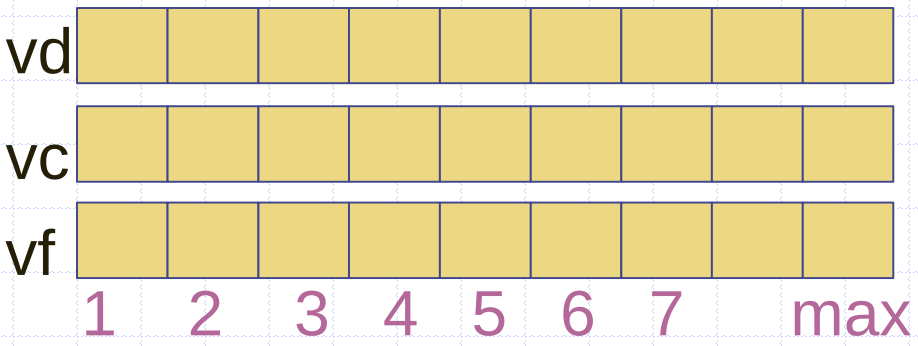
$$\text{vd} = \begin{bmatrix} 12 & 9 & 7 & 5 & 1 & 2 & 11 & 3 & 6 & 4 & 8 & 10 \end{bmatrix},$$
$$\text{vf} = \begin{bmatrix} 5 & 3 & 3 & 2 & 1 & 1 & 4 & 2 & 3 & 2 & 3 & 4 \end{bmatrix},$$
$$\text{vc} = \begin{bmatrix} 5 & 5 & 3 & 4 & 1 & 4 & 4 & 1 & 1 & 2 & 4 & 3 \end{bmatrix}.$$

6.4.1 Implementacion basada en vectores

Definición básica de la clase MatrizDispersa cuya implementacion es usando vectores

```
Clase MatrizDispersa
    Atributos
        Vf,                          // filas
        VC,                          // Columnas
        VD   :     Arreglo(MAX)      // elementos
        df,dc :    Entero           // Dimension
        repe   :   elemento
        nt     : Metodos
       Crear()
       dimensionar(df,dc:entero)
       dimension_Fila()
       dimension_columna()
       poner(f,c:indice; valor:elemento)
       Elemento(f,c:indice)
       definir_valor_repetido(valor:elemento)
   fin
Constructor  matrizdispersa.Crear
  inicio
      df=0 dc=0 repe=0
   fin
```

Nt=0
Df =0
Dc=0
Repe=0



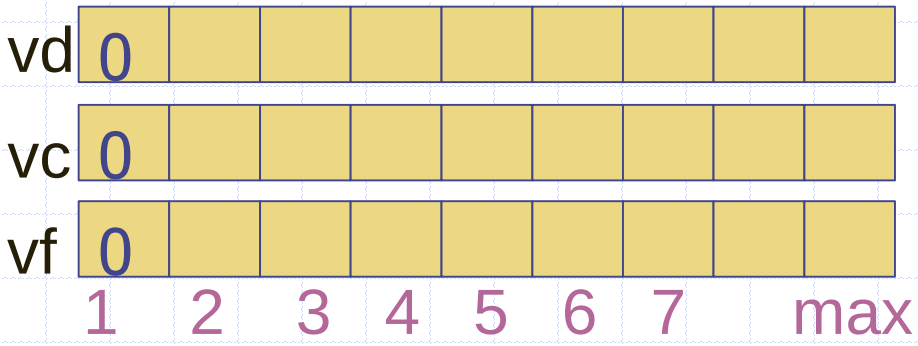
6.4.1 Implementacion basada en vectores

Definición básica de la clase MatrizDispersa cuya implementacion es usando vectores

```
Publico matrizdispersa.poner(f,c: entero; e: Elemento)
Inicio
   Lug = // Buscar en vector vf,vc los valores f y c y retornar indice
   si lug>0 entonces vd[ lug ] = e
            si vd[lug]=rep entonces // desplazar
            caso contrario
               si nt< MAX entoces
                           nt = nt +1
                           vd[ nt ] = e vf[ nt ] = f vc[ nt ] = c
                         caso contrario
                           // error no existe espacio
fin
```

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Nt=1
Df =1000
Dc=1000
Rep=0

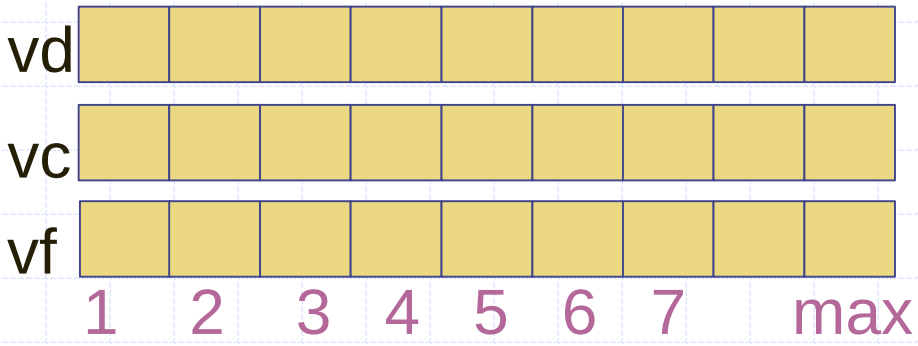


6.4.1 Implementacion basada en vectores

```
Publico entero matrizdispersa.elemento(f,c: entero)
Inicio
     si (f>=1 y f<= df)  y ( c>=1 y c<=dc) entoces
                  lug = // buscar f,c en vectores vc,vf y retornar lugar
                  si lug>0 entoces
                                retornar vd[lug]
                           caso contrario
                                retornar repe
               caso contrario
                 // Error fuera de rango indices
fin
```

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Nt=1
Df =1000
Dc=1000
Rep=0



Formato CSR (Compressed Sparsed Row)

$$A = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 \\ 3 & 4 & 0 & 5 & 0 \\ 6 & 0 & 7 & 8 & 9 \\ 0 & 0 & 10 & 11 & 0 \\ 0 & 0 & 0 & 0 & 12 \end{bmatrix}$$

$$\begin{aligned} AA &= \begin{bmatrix} 12 & 9 & 7 & 5 & 1 & 2 & 11 & 3 & 6 & 4 & 8 & 10 \end{bmatrix}, \\ IA &= \begin{bmatrix} 5 & 3 & 3 & 2 & 1 & 1 & 4 & 2 & 3 & 2 & 3 & 4 \end{bmatrix}, \\ JA &= \begin{bmatrix} 5 & 5 & 3 & 4 & 1 & 4 & 4 & 1 & 1 & 2 & 4 & 3 \end{bmatrix}. \end{aligned}$$

$$\begin{aligned} IA(1) &= 1\,, \\ IA(i+1) - IA(i) &= \text{numero de elementos no nulos en la fila } i\,. \end{aligned}$$

Así, la matriz A en el formato CRS se representa por

$$\begin{aligned} \text{Vd / } AA &= \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{bmatrix}, \\ \text{Vc / } JA &= \begin{bmatrix} 1 & 4 & 1 & 2 & 4 & 1 & 3 & 4 & 5 & 3 & 4 & 5 \end{bmatrix}, \\ \text{Vf / } IA &= \begin{bmatrix} 1 & 3 & 6 & 10 & 12 & 13 \end{bmatrix}. \end{aligned}$$

6.4.1 Implementacion basada en vectores

Definición básica de la clase MatrizDispersa cuya implementacion es usando vectores

```
Publico entero matrizdispersa.elemento(f,c : entero)
Inicio
   si No((f>=1 y f<= df)   y ( c>=1 y c<=dc)) entoces // error fuera de
      rango
   lug_antes=0
   para cada I = 1 hasta (f-1)
        lug_antes=lug_antes + (vf(i+1) - vf( i ))

   max_elem_fila=(vf(f+1)-vf(f))

   para cada i=1 hasta max_elem
        si vc(lug_antes+i)=c entoces retornar vd(lug_antes+i)
   retornar repe
Fin
Publico matrizdispera.definir_valor_repetido( e : elemento)
Inicio
      // si la matriz no tiene elementos
      // solo se realiza lo siguiente repe= e

      // si la matriz tiene elementos  y ninguno de sus elementos
      // es igual a e entoces repe= e

      // cualquier otra situacion deberia redefinirse los valores de
      // los vectores
Fin
```

6.4.2 Implementación con apuntadores

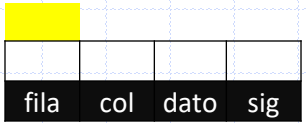
Definición básica de la clase Matrizdispersa implementada usando punteros:

```
Tipo de dato
 Nodo
     fila      Entero,
     col       Entero,
     dato      Entero,
     Sig       Puntero a Nodo
 // fin definicion
 Direccion Puntero a espacio de memoria de tipo Nodo
Clase Matrizdispersa
  Atributos
     Nt          Entero // Numero de Terminos
     PtrMatD     Direccion
     rep, dimf,dimc  Entero
 Metodos
```

6.4.2 Implementación con apuntadores

```
publico matrizdispersa.Crear()
    inicio
        ptrmatd=nulo
        dimf= 0
        dimc= 0
        rep=0
    fin
```

PtrMatD	-1
Nt	0
Rep	0
DimF	0
DimC	0



DIR	DATO	ID	LINK
1			2
2			3
3			4
4			5
5			6
6			7
7			8
8			9
9			10
10			11
11			12
12			13
13			14
14			15
15			-1

LIBRE=1

6.4.2 Implementación con apuntadores

```
publico matrizdispersa.poner(f,c,e)
   inicio
     dir= buscar si existe f,c en los nodos
      si dir=nulo entonces
          x = new_espacio('fila,col,dato,sig')
          si x<>null entonces
           poner_dato(x,'->fila',f)
           poner_dato(x,'->col',c)
           poner_dato(x,'->dato',e)
           poner_dato(x,'->sig',ptrmatd)
           ptrmatd=x
           nt=nt +1
           caso contrario
              // error no existe eespacio memoria
       fin si
       caso contraio
        poner_dato(dir,'->dato',e)
        si e=rep entices
             //eliminar nodo
             nt = nt -1
        fin si
     fin si
   fin
```

PtrMatD	-1
Nt	0
Rep	0
DimF	10000
DimC	10000

DIR	DATO	ID	LINK
1			2
2			3
3			4
4			5
5			6
6			7
7			8
8			9
9			10
10			11
11			12
12			13
13			14
14			15
15			-1

LIBRE=9

Implementación con Nodos por Fila

Vrep=0

Nf=4

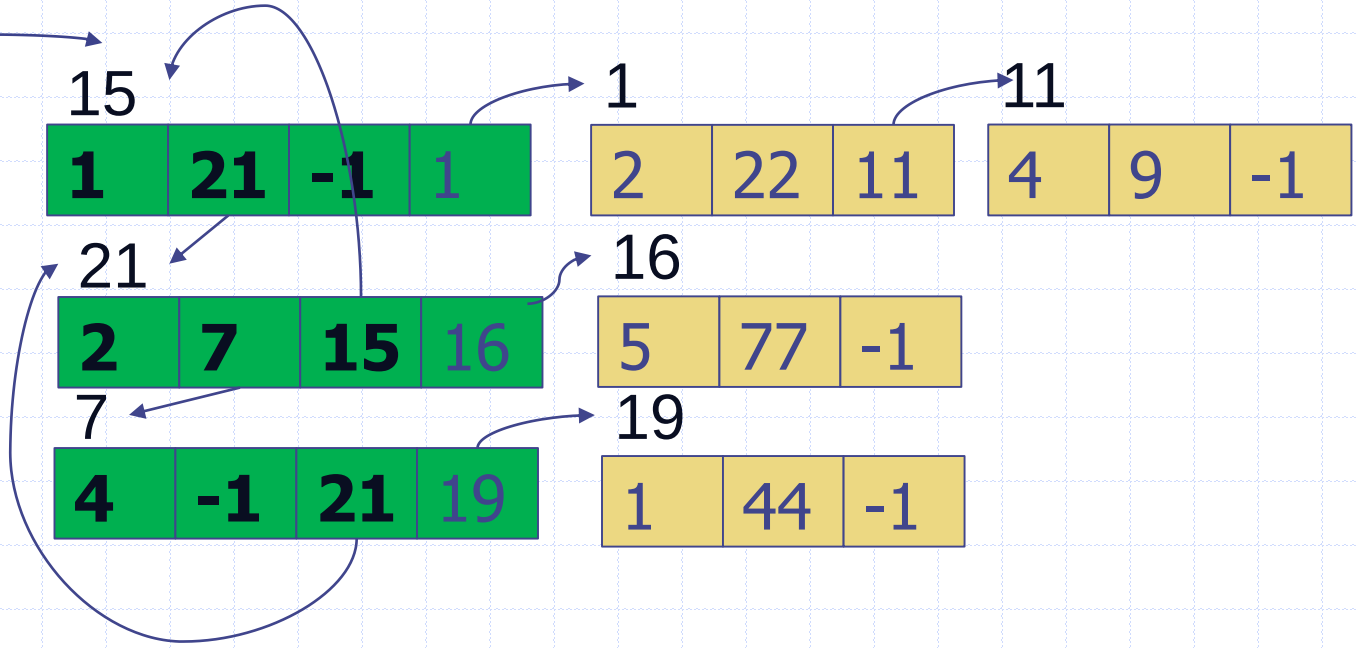
Nc=5

PtrFil=15

NodoF (Fil,SigF,AntF,PtrCol)

NodoC(Col,Dato,SigCol)

0	**22**	0	**9**	0
0	0	0	0	**77**
0	0	0	0	0
44	0	0	0	0



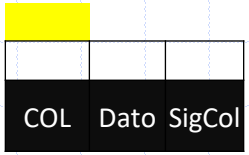
Implementación con Nodos por Fila

rep=0

Nf=4

Nc=5

PtrFil= 9



0	**55**	0	**77**	0
0	0	0	0	**100**
0	0	99	0	0
22	0	0	0	0

Elaborar la aplicación 1 y 2 de forma separada

1) Crear Memoria
2) Pedir espacio
3) Liberar Espacio
4) Crear Matriz
5) Dimensionar Matriz
6) Asignar valor por defecto
7) Poner Dato
8) Mostrar Matriz
9) Salir

Implementada con

- Un tipo de nodo
- Dos tipos nodos Filas y Columnas

(Dos proyectos)

1) Crear Matriz
2) Dimensionar Matriz
3) Asignar valor por defecto
4) Poner Dato
5) Mostrar Matriz
6) Salir

Implementada con

- 3 vectores sin comprimir
- 3 vectores comprimir vector fila
- Punteros de un solo tipo de nodo
- Punteros con dos tipos de nodos filas y columnas

(cuatro proyectos)