Treball 1: Matrius disperses

Ismael El Habri, Marc Cané, Lluís Trilla

16 d'octubre de 2018

Índex

Capítol 1

Què són les matrius disperses?

Quan parlem de matrius disperses ens referim a matrius de gran tamany en la qual la majoria d'elements son zero. Direm que una matriu és disperrsa, quan hi hagi benefici en aplicar els mètodes propis d'aquestes.

Per identificar si una matriu és dispersa, podem usar el seguent:

> Una matriu $n \times n$ serà dispersa si el número de coeficients no nuls es $n^{\gamma+1}$, on $\gamma < 1$.

En funció del poblema, decidim el valor del paràmetre γ. Aquí hi ha els valors típics de γ:

- $\gamma = 0.2$ per problemes d'anàlisi de sistemes eléctics degeneració i de transpot d'enegía.
- $\gamma = 0.5$ per matrius en bandes associades a problemes d'anàlisi d'estructues.

Podem trobar dos tipus de matrius disperses:

- **Matrius estructurades:** matrius en les quals els elements diferents de zero formen un patró regular. Exemple: Les matrius banda.
- **Matrius no estructurades:** els elements diferents de zero es distribueixen de forma irregular.

Capítol 2

Formes d'emmagatzemar matrius disperses

Per Coordenades

És la primera aproximació que podríem pensar i és bastant intuïtiva. Per cada element no nul guardem una tupla amb el valor i les seves coordenades: (a_{ij}, i, j).

Exemple

$$\begin{pmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 3 & 0 & -2 & 0 \end{pmatrix} =$$

índex	tupla(a_{ij}, i, j)
0	$(1, 1, 1)$
1	$(2, 1, 4)$
2	$(1, 2, 2)$
3	$(3, 4, 1)$
4	$(-2, 4, 3)$

Per emmagatzemar això podem usar tres vectors de la mateixa mida (n_z, el nombre d'elements diferents de zero): Un amb els valors, un amb les files i un amb les columnes:

Vector	Coeficients				
valors	1	2	1	3	-2
files	1	1	2	4	4
columnes	1	4	2	1	3

A la realitat però, aquest mètode d'emmagatzemar les dades és poc eficient quan hem de fer operacions amb les matrius.

Per files

També conegut com a *Compressed Sparse Rows (CSR)*, *Compressed Row Storage (CRS)*, o format *Yale*. És el mètode més estès.

Consisteix en guardar els elements ordenats per files, guardar la columna on es troben, i la posició del primer element de cada fila en el vector de valors. Així ens quedaran tres vectors:

- **valors:** de mida n_z, conté tots els valors diferents.
- **columnes:** també de mida n_z, conté la columna on es troba cada un dels elements anteriors.
- **iniFiles:** de mida $m+1$, conté la posició on comença cada fila en els vectors valors i columnes, sent m el nombre de files de la matriu.

Exemple

$$\begin{pmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 3 & 0 & -2 & 0 \end{pmatrix} \quad = \quad \begin{array}{l|ccccc} \text{Vector} & \multicolumn{5}{c}{\text{Coeficients}} \\ \hline \text{índex} & 1 & 2 & 3 & 4 & 5 \\ \hline \text{valors} & 1 & 2 & 1 & 3 & -2 \\ \text{columnes} & 1 & 4 & 2 & 1 & 3 \\ \text{iniFiles} & 1 & 3 & 4 & 4 & 6 \end{array}$$

Si es canvien files per columnes, dona la implementació per columnes, o també anomenada *Compressed Sparse Columns (CSC)*.

Implementació del mètode CSR

Hem implementat un script Matlab amb una classe `CSRSparseMatrix` que guardi les dades necessàries. Aquestes les tenim en "l'atribut" `Matrix` dins del bloc `properties` (línia 10 del codi posterior). Aquestes dades consisteixen en el següent:

- `Matrix.nColumns`: número de columnes de la matriu, necessari per recrear les files posteriorment.
- `Matrix.values`: vector valors comentat anteriorment, amb els valors no nuls de la matriu.
- `Matrix.columns`: vector de columnes, amb la columna corresponent a cada valor amb el mateix índex.
- `Matrix.beginningRow`: vector amb els índex comença cada fila en el vector de valors i de columnes.

Constructor

Entenent com s'emmagatzema la matriu, fer el constructor de la classe és trivial (a partir de la línia 20 del codi posterior): només és necessari recórrer la matriu per files i guardar al vector de valors els elements diferents de 0, guardar-nos la columna on es troba cada un d'aquests elements en el vector de columnes. A part, per cada fila, hem de actualitzar el vector d'índexs de files. Per fer-ho seguim el següent:

- `Matrix.beginningRow[1] = 0`.
- `Matrix.beginningRow[i] = Matrix.beginningRow[i-1]` + elements diferents de zero en la fila i.

Mètodes per obtenir una fila, una columna i un element

El mètode per obtenir una fila es troba a partir de la línia 46 del codi posterior. Com podem veure consisteix en primer crear una fila amb `Matrix.nColumns` zeros. Llavors, sabent que `Matrix.beginningRow[i]` ens indica on comença la fila i en el vector de valors i en el de columnes, i deduint que acaben a `Matrix.beginningRow[i+1]-1`; podem delimitar els elements de la fila. Només quedaria afegir aquests elements a les seves posicions corresponents (usant el vector de columnes).

El mètode per obtenir un element es troba a partir de la línia 62 del codi posterior. Aquest mètode, usa el que hem vist anteriorment per crear la fila determinada, llavors només cal obtenir l'element amb la columna.

El mètode per obtenir una columna es troba a partir de la línia 72 del codi posterior. Sabent quantes files tenim (recordem que `Matrix.beginningRow` té $m+1$ elements, on m és el nombre de files de la matriu), inicialitzem cada element a 0 i busquem els elements de cada posició de la columna, usant el mateix mètode usat per obtenir les files. Cal tenir en compte, que per cada fila, les columnes estan ordenades, així que no cal buscar més quan ja hem passat la columna que estem mirant.

Script Matlab

```
1  %===============CSRSparseMatrix=====================================================
2  % Classe que implementa el mètode d'emmagatzematge per files sobre matrius
3  %%% disperses
4  %
5  % El constructor rep una matriu i hi aplica el metode, tornant un objecte
6  %%% de tipus CSRSparseMatrix.
7  %
8  classdef CSRSparseMatrix
9    properties
10     Matrix
11     %%% La matriu (estructura de quatre elements:
12     %%%                 n: Matrix.nColumns,
13     %%%                 valors: Matrix.values,
```

```
14      %%%                 columnes: Matrix.columns,
15      %%%                 inici files: Matrix.beginningRow)
16    end
17
18    methods
19      %==============Constructor=============================================
20      function obj = CSRSparseMatrix(A)
21        [m,n] = size(A);
22        obj.Matrix.nColumns = n;
23        obj.Matrix.values = [];
24        obj.Matrix.columns = [];
25        obj.Matrix.beginningRow = [1];
26        for i = 1:m
27          nonZero=obj.Matrix.beginningRow(i);
28          nonZeroThisRow=0;
29          for j = 1:n
30            if(A(i,j) !=0)
31              obj.Matrix.values = [obj.Matrix.values, A(i,j)];
32              obj.Matrix.columns = [obj.Matrix.columns, j];
33              nonZeroThisRow=nonZeroThisRow+1;
34            end
35          end
36          obj.Matrix.beginningRow = [obj.Matrix.beginningRow, nonZero+nonZeroThisRow];
37        end
38      end
39      %=============FI CONSTRUCTOR ============================================
40
41      %============getRow=====================================================
42      %
43      %%% Donats obj, sent el objecte actual, i x la fila que volem obtenir;
44      %%%% retorna la fila x de la matriu obj
45      %
46      function row = getRow(obj, x)
47        row = 0;
48        for i= 1:obj.Matrix.nColumns
49          row(i)=0;
50        end
51        for i = obj.Matrix.beginningRow(x):obj.Matrix.beginningRow(x+1)-1
52          row(obj.Matrix.columns(i)) = obj.Matrix.values(i);
53        end
54      end
55
56      %============getElem====================================================
57      %
58      %%% Donats obj, sent el objecte actual i (x,y) les coordenades del element
59      %%%% que volem obtenir;
60      %%%% retorna l'element x,y de la matriu obj
61      %
```

```
62      function elem = getElem(obj, x, y)
63        row = obj.getRow(x);
64        elem = row(y);
65      end
66
67      %============getRow===================================================================
68      %
69      %%% Donats obj, sent el objecte actual, i y la columna que volem obtenir;
70      %%%% retorna la columna y de la matriu obj
71      %
72      function column = getColumn(obj, y)
73        column = 0;
74        m=size(obj.Matrix.beginningRow,2)-1;
75        for j=1:m
76          column(j)=0;
77          for i=obj.Matrix.beginningRow(j):obj.Matrix.beginningRow(j+1)-1
78            if obj.Matrix.columns(i)==y column(j) = obj.Matrix.values(i), break
79            elseif obj.Matrix.columns(i)> y break
80            end
81          end
82        end
83      end
84
85    end
86  end
```

Per perfil

Aquest mètode és una manera eficient de guardar un tipus concret de matrius, les matrius banda.

Matrius banda

Com vam veure a classe, una matriu $n \times n$ és banda si existeixen p i q naturals, tals que $1 < p, q < n$ i $a_{ij} = 0$ sempre que $p \leq j - i$ o $q \leq i - j$.

Anomenem ample de banda de la matriu a $p + q - 1$.

L'**envoltant** de una matriu banda consisteix en tots els elements de cada fila des de el primer no nul fins al ultim no nul, incloent els elements nuls que hi pugui haver entre mig. Aquí la definició formal:

L'envoltant de una matriu banda A, $env(A)$ es defineix com el conjunt $env(A) = \{i, j\} : f_i \leq j \leq l_i, 1 \leq i \leq n$ on f_i és on comencen els valors no nuls de la fila i, i l_i on acaben els valors no nuls de la fila i.

El mètode

Consisteix en guardar els elements de l'envoltant de la matriu, la columna on comença l'envoltant en cada fila, i la posició en el vector de valors on comença cada fila.

Doncs, ens quedarien tres vectors:

- **valors:** amb els valors de l'envoltant.
- **columnInici:** amb el valor f_i de cada fila (la columna on comença l'envoltant en cada fila.
- **iniFiles:** amb la posició on comença cada fila en el vector valors.

Exemple

$$\begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 2 \\ 0 & 0 & 0 & -2 \end{pmatrix} \quad = \quad$$

Vector	Coeficients						
índex	1	2	3	4	5	6	7
valors	1	2	1	2	0	2	−2
columnInici	1	2	2	4			
iniFiles	1	3	4	7			

Altres mètodes

Diccionari de claus

Conegut també com a *Dictionary of Keys (DOK)*, consisteix en tenir un diccionari que mapeja parelles de (fila, columna) amb el valor de cada element. Els elements que no estan en el diccionari es poden considerar zero. Aquest mètode es bo per construir la matriu de manera incremental en un ordre aleatori, però es dolent al iterar pels elements diferents de zero en un ordre lexicogràfic. El seu ús més habitual es usar aquest format per construir la matriu, per després convertir-la en un format més eficient de processar.

Llista de llistes

Guarda una llista per fila, en la qual cada entrada és una parella de valors (valor, columna). S'acostumen a ordenar per número de columna per motius d'eficiència. Aquest mètode també es bo per construir la matriu de forma incremental.

Esquema DIA

Aquest esquema s'usa quan els valors no nuls estan restringits a un reduït nombre de diagonals. Consisteix en guardar una matriu de dades que conté els valors no nuls i un vector amb els *offsets*, que guarda el desplaçament de cada diagonal respecte la diagonal principal.

A la diagonal principal li correspon l'*offset* 0. A les diagonals superiors, els hi assignem valors positius; a les inferiors, valors negatius.

Exemple

Donada la matriu:

$$\begin{pmatrix} 1 & 7 & 0 & 0 \\ 0 & 2 & 8 & 0 \\ 5 & 0 & 3 & 9 \\ 0 & 6 & 0 & 4 \end{pmatrix}$$

Necessitaríem guardar el següent:

$$dat = \begin{pmatrix} * & 1 & 7 \\ * & 2 & 8 \\ 5 & 3 & 9 \\ 6 & 4 & * \end{pmatrix}, off = (-2, 0, 1)$$