Laboratorio didattico di matematica computazionale

Beatrice Meini

Lezione 8 - 2/5/2012

Attenzione!

La versione di `octave` presente sulle macchine dell'aula M è troppo vecchia per poter eseguire gli esercizi di oggi, infatti non supporta certe funzioni. Dunque occorre usare il proprio calcolatore, oppure `octave` (o in alternativa `qtoctave`) delle macchine dell'aula 4, ad esempio. Per far questo occorre collegarsi ad una lab[numero]:

`ssh -X [vostra login]@ssh1.dm.unipi.it`

e successivamente, per non lavorare tutti sulla stessa macchina,

`ssh -X lab[numero che vi pare]`

Dare il comando `xterm` per aprire un'altra shell.

1 Il "Pagerank" di Google

Il successo del motore di ricerca Google è dovuto alla capacità di ordinare, in ordine di importanza decrescente, le pagine web che rispondono ad una ricerca (query) dell'utente. Infatti, generalmente, solo le prime pagine web elencate sono quelle che interessano all'utente.

L'ordinamento delle pagine web si basa su un modello matematico di navigazione sul web. Secondo questo modello, un utente che naviga nel web "di norma" sceglie con distribuzione di probabilità uniforme un link dalla pagina che sta visitando; invece talvolta l'utente decide di scegliere, ancora con distribuzione di probabilità uniforme, una pagina web scelta tra tutte le pagine web esistenti. Secondo il modello di Google, la pagina web w_1 è *più importante* della pagina web w_2 se, facendo tendere il tempo di navigazione all'infinito, l'utente visita la pagina w_1 con probabilità maggiore rispetto alla pagina w_2.

Questo concetto può essere formalizzato matematicamente nel seguente modo. Sia W l'insieme delle pagine web che possono essere raggiunte seguendo link successivi a partire da qualche pagina fissata, e sia n la cardinalità di W. L'insieme W varia nel tempo; l'insieme utilizzato da Google è formato da circa $4 \cdot 10^9$ pagine! Sia $G = (g_{i,j})$ la matrice $n \times n$ che rappresenta le connessioni tra una pagina e l'altra: $g_{i,j} = 1$ se c'è un link dalla pagina j alla pagina i, e $g_{i,j} = 0$ altrimenti. La matrice G può avere enorme dimensione, ma è molto sparsa.

Sia c_j la somma degli elementi sulla j-esima colonna di G:

$$c_j = \sum_i g_{i,j}$$

Sia p la probablità che l'utente segua un link dalla pagina che sta visitando (dunque $1-p$ è la probablità che l'utente scelga una pagina arbitraria). Definiamo la matrice $A = (a_{i,j})$ di dimensione $n \times n$:

$$a_{i,j} = \begin{cases} p\frac{g_{i,j}}{c_j} + \frac{1-p}{n}, & se \;\; c_j \neq 0 \\ \frac{1}{n}, & se \;\; c_j = 0 \end{cases}$$

Si osservi che per costriure la matrice A si scala la matrice G mediante la somma degli elementi su ciascuna colonna. Se la somma degli elementi lungo la colonna j è nulla, significa che non ci sono link che partono dalla pagina web j; in questo caso assegnamo una probabilità uniforme, uguale a $1/n$, a tutti gli elementi della j-esima colonna.

Valori tipici usati nel calcolo del Pagerank di Google sono $n = 4 \cdot 10^9$ and $p = 0.85$; per questi valori $(1-p)/n = 3.75 \cdot 10^{-11}$.

I comandi

```
octave:2> n = 10;
octave:3> G = sprand(n,n,0.1);
```

generano una matrice 10×10 random G sparsa, dove 0.1 è la densità di sparsità (si veda l'help per il funzionamento). Per trasfomare G in una matrice con elementi uguali a 0 o 1 possiamo dare il comando

```
octave:4> G=(G!=0)
G =

Compressed Column Sparse (rows = 10, cols = 10, nnz = 10)

  (1, 1) -> 1
  (9, 2) -> 1
  (1, 4) -> 1
  (10, 4) -> 1
  (10, 7) -> 1
  (6, 8) -> 1
  (7, 8) -> 1
  (10, 8) -> 1
  (10, 9) -> 1
  (5, 10) -> 1
```

Le matrici sparse in octave sono rappresentare nnz da terne, dove nnz è il numero di elementi diversi da zero, e ciascuna terna è costituita da indice di riga, indice di colonna, e elemento corrispondente. In pratica, vengono memorizzati solo gli elementi diversi da zero.

Per visualizzare tutti gli elementi della matrice G occorre dare il comando

```
octave:7> full(G)
ans =

   1   0   0   1   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0
```

```
   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   1
   0   0   0   0   0   0   0   1   0   0
   0   0   0   0   0   0   0   1   0   0
   0   0   0   0   0   0   0   0   0   0
   0   1   0   0   0   0   0   0   0   0
   0   0   0   1   0   0   1   1   1   0
```

Il comando

```
octave:8> spy(G)
```

mostra graficamente gli elementi diversi da zero.

Esercizio 1. Si costruisca una matrice G sparsa di dimensione 100, con elementi 0 o 1, e si costrisca la matrice A, scegliendo $p = 0.85$. Si verifichi sperimentalmente che la matrice A ha elementi non negativi, ed ha somma degli elementi uguali a 1 su ogni colonna. Per calcolare la somma degli elementi su ogni colonna si utilizzi l'istruzione `sum`.

È possibile dimostrare che esiste un unico vettore x, con elementi non negativi, tale che

$$x = Ax, \quad \sum_{i=1}^{n} x_i = 1.$$

Il vettore x rappresenta il Pagerank di Google: la pagina corrispondente all'indice j è più importante della pagina corrispondente a i se $x_j > x_i$. Infatti, x_j è la probabilità che, per il tempo che tende a infinito, l'utente visiti la pagina j.

Un metodo per calcolare il vettore x è generare la successione di vettori:

$$x^{(0)} = [1/n, \ldots, 1/n]^T, \quad x^{(k+1)} = Ax^{(k)} \quad k \geq 0. \tag{1}$$

Infatti la successione $\{x^{(k)}\}$ converge a x (se volete saperne di più cercate con Google "power method").

Esercizio 2. Definire la `function x=pow(A,maxit,tol)` che prende in input la matrice A, un numero massimo di iterazioni `maxit` e una tolleranza `tol`, e che restituisce in output una approssimazione del vettore x ottenuta mediante l'iterazione (1), dove il calcolo della successione è interrotto se $k =$`maxit` oppure $||x^{(k)} - x^{(k-1)}||_1 <$`tol`.

2 Un esempio di piccola rete

Salvare il file `www.dm.unipi.it/meini/LDMC12/harvard500.mat` e dare il comando `load harvard500.mat`. La matrice G è la matrice dei link di una sottorete 500×500 ottenuta partendo dal sito `www.harvard.edu`; il vettore U è un array di stringhe, che contiene i nomi delle pagine web.

Dare il comando `spy(G)` per vedere la struttura di G.

Il grosso difetto della function `pow` è che non sfrutta la sparsità della matrice G. La successione $x^{(k)}$ infatti può essere calcolata non calcolando esplicitamente la matrice A. Infatti la matrice A può essere scritta come

$$A = pGD + ez^T$$

dove e è il vettore con tutte le componenti uguali a 1, D è la matrice diagonale con elementi diagonali

$$d_{j,j} = \begin{cases} 1/c_j & se \quad c_j \neq 0 \\ 0 & se \quad c_j = 0 \end{cases}$$

e z è il vettore con componenti

$$z_j = \begin{cases} (1-p)/n & se \quad c_j \neq 0 \\ 1/n & se \quad c_j = 0 \end{cases}$$

Dunque si ottiene che

$$x^{(k+1)} = pGDx^{(k)} + e(z^T x^{(k)}). \tag{2}$$

In questo modo per il calcolo di $x^{(k+1)}$ viene sfruttata la sparsità di G.

Le istruzioni

```
octave:8> c=sum(G,1);
octave:9> k=find(c!=0);
octave:10> D = sparse(k,k,1./c(k),n,n);
octave:11> e = ones(n,1);
octave:12> p=0.85;
octave:13> Gs=p*G*D;
octave:14> z = ((1-p)*(c!=0) + (c==0))/n;
```

costruiscono la matrice D sparsa, la matrice $Gs = pGD$ sparsa e il vettore z.

Esercizio 3. Definire la `function x=pow2(G,maxit,tol)`, modificando la `pow`, in modo che la successione di vettori sia calcolata mediante la formula (2). Scegliere n abbastanza grande, e confrontare i risultati di `pow` e `pow2`.

Esercizio 4. Calcolare il vettore x mediante `pow2`. Successivamente visualizzare il Pagerank e le pagine corrispondenti mediante i comandi:

```
bar(x)
title('Page Rank')

% Print URLs in page rank order.

   [ignore,q] = sort(-x);
   disp('      page-rank   url')
   k = 1;
   while (k <= n) & (x(q(k)) >= .005)
      j = q(k);
      disp(sprintf(' %3.0f %8.4f  %s', j,x(j),U{j}))
      k = k+1;
   end
```