Sparse Vectors & Matrices

Jyh-Shing Roger Jang (張智星)
CSIE Dept, National Taiwan University

Array Representation of Polynomials

- A polynomial of order n
  - $a_n x^n + a_{n-1} x^{n-1} + \ldots + a_2 x^2 + a_1 x + a_0$
- Straightforward representation
  - Unique exponent arranged in increasing order
  - For instance: $7\,x^5 + 4\,x^3 + 2\,x^2 + 3$

Representation:

p.degree = 5

p.coef =

3	0	2	4	0	7							

- Characteristics
  - Easy to implement addition & subtraction
  - Waste of space to represent a sparse polynomial   x^1000+x+1
  - Complexity of addition: O(max(m, n))

Order of a(x).

Order of b(x).

Sparse Array Representation of Polynomials

- A polynomial of order n
  - $a_n x^n + a_{n-1} x^{n-1} + \ldots + a_2 x^2 + a_1 x + a_0$
- Representation
  - Keep non-zero terms only
  - For instance, see the right figure.
- How to add two polynomials
  - Traverse each polynomials
  - Add terms of the same exponent
- Characteristics
  - Takes only necessary memory
  - Complexity of addition & subtraction: O(m+n)

No. of terms in a(x).

No. of terms in b(x).

$a(x) = 2x^{1000} + 1$

$b(x) = x^4 + 8x^3 + 3x^2 + 1$

Representation of a(x):

coef:	2	1	~~*~~	~~*~~	~~*~~	~~*~~	~~*~~	...
exp:	1000	0	~~-1~~	~~-1~~	~~-1~~	~~-1~~	~~-1~~	...

Representation of b(x):

coef:	1	8	3	1	~~*~~	~~*~~	~~*~~	~~...~~
exp:	4	3	2	0	~~-1~~	~~-1~~	~~-1~~	~~...~~

Sparse Matrices

- Matrix representation
  - Dense:

15	0	0	22	0	-15
0	11	3	0	0	0
0	0	0	-6	0	0
0	0	0	0	0	0
91	0	0	0	0	0
0	0	28	0	0	0

  Sparse:

6	6	8
0	0	15
0	3	22
0	5	-15
1	1	11
1	2	3
2	3	-6
. . .	. . .	. . .

  - Sparse format
    - The first row stores the numbers of rows, columns, and non-zero elements, respectively.
    - Elements are sorted by row first, by column second.

Complexity of Operations on Dense Matrices

- Operations
  - c=transpose ➔ O(mn)

```
for (i=0; i<m; i++)
        for (j=0; j<n; j++)
                c[j][i]=a[i][j];
```

  - c=add(a, b) ➔ O(mn)

```
for (i=0; i<m; i++)
        for (j=0; j<n; j++)
                c[i][j]=a[i][j]+b[i][j];
```

  - c=multiply(a, b) ➔ O(pqr)

```
for (i=0; i<p; i++)
        for (j=0; j<r; j++){
                sum=0;
                for (k=0; k<q; k++)
                        sum+=a[i][k]*b[k][j];
                c[i][j]=sum;
        }
```

Algorithm 1 for Transposing a Sparse Matrix

- Algorithm 1
  - Pseudo code

```
for each entry {
    take term (i, j, value) and store it as (j, i, value)
}
```

  - For example
    - (0, 0, 15) ➔ (0, 0, 15)
    - (0, 3, 22) ➔ (3, 0, 22)
    - (0, 5, -15) ➔ (5, 0, -15)
    - (1, 1, 11) ➔ (1, 1, 11)
    - …

#NZ = No. of non-zero elements
= No. of entries

  - Complexity: O(#NZ)+O(#NZ*log(#NZ))+O(#NZ*log(#NZ)) ➔ O(#NZ*log(#NZ))

Algorithm 2 for Transposing a Sparse Matrix

- Algorithm 2
  - Idea: Find all terms in column 0 and store them in row 0; find all terms in column 1 and store them in row 1, and so on.
  - Pseudo code

```
for (j=0; j<colNum; j++)
    for all term in column j
        place the entry (i, j, value) in the next position of the output
```

  - Example:

a[0]	6	6	8
a[1]	0	0	15
a[2]	0	3	22
a[3]	0	5	-15
a[4]	1	1	11
a[5]	1	2	3
a[6]	2	3	-6
a[7]	4	0	91
a[8]	5	2	28

===>

b[0]	6	6	8
b[1]	0	0	15
b[2]	0	4	91
b[3]	1	1	11
b[4]	2	1	3
b[5]	2	5	28
b[6]	3	0	22
b[7]	3	2	-6
b[8]	5	0	-15

  - Complexity: O(#col*#NZ)

Algorithm 3 for Transposing a Sparse Matrix

- Algorithm 3
  - A fast algorithm that scans the term list only twice, as follows
    - Find number of terms in a row and then find the starting position of each row.
    - Fill the output matrix.
  - Example

a[0]	6	6	8
a[1]	0	0	15
a[2]	0	3	22
a[3]	0	5	-15
a[4]	1	1	11
a[5]	1	2	3
a[6]	2	3	-6
a[7]	4	0	91
a[8]	5	2	28

===>

i	rowTermCount[i] (no. of terms in row i)	rowStart[i] (Start position for row i)
0	0→1→2	1
1	0→1	3
2	0→1→2	4
3	0→1→2	6
4	0	8
5	0→1	8

===>

b[0]	6	6	8	
b[1]	0	0	15	←rowStart[0]=1
b[2]	0	4	91	
b[3]	1	1	11	←rowStart[1]=3
b[4]	2	1	3	←rowStart[2]=4
b[5]	2	5	28	
b[6]	3	0	22	←rowStart[3]=6
b[7]	3	2	-6	
b[8]	5	0	-15	←rowStart[4]=rowStart[5]=8

  - Complexity: O(#row + #NZ)