SPARSE MATRIX ADDITION

BY CHAITALI PATIL

Assistant Professor,Dept. of Computer Engg.
K.K.Wagh Institute of Engineering Education and Research, Nashik

Objectives

How to perform sparse matrix addition

Sparse Matrix Representation: Triplet Representation (Array Representation)

$$
\begin{array}{c}
\begin{array}{cc}
 & \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{array} &
\begin{bmatrix}
0 & 0 & 0 & 0 & 9 & 0 \\
0 & 8 & 0 & 0 & 0 & 0 \\
4 & 0 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 5 \\
0 & 0 & 2 & 0 & 0 & 0
\end{bmatrix}
\end{array} \\
5x6
\end{array}
$$

Rows	Columns	Values	
5	6	6	0
0	4	9	1
1	1	8	2
2	0	4	3
2	3	2	4
3	5	5	5
4	2	2	6

Question

$$\begin{pmatrix} 10 & 20 & 0 & 0 & 0 & 0 \\ 0 & 30 & 0 & 40 & 0 & 0 \\ 0 & 0 & 50 & 60 & 70 & 0 \\ 0 & 0 & 0 & 0 & 0 & 80 \end{pmatrix}$$

For above given matrix, give its 0^{th} row values, if it is represented in sparse represented matrix using Triplet.
Give Triplet representation of given sparse matrix

Answer

$$\begin{pmatrix} 10 & 20 & 0 & 0 & 0 & 0 \\ 0 & 30 & 0 & 40 & 0 & 0 \\ 0 & 0 & 50 & 60 & 70 & 0 \\ 0 & 0 & 0 & 0 & 0 & 80 \end{pmatrix}$$

Sparse Matrix

4	6	8
0	0	10
0	1	20
1	1	30
1	3	40
2	2	50
2	3	60
2	4	70
3	5	80

Triplet Representation

Sparse Matrix Addition

- Matrix 1: (5x5) +Matrix 2: (5x5)

	0	1	2
0	5	5	5
1	1	2	10
2	1	4	12
3	3	3	5
4	4	1	15
5	4	2	12

	0	1	2
0	5	5	5
1	1	3	8
2	2	4	23
3	3	3	9
4	4	1	20
5	4	2	25

0	5	5	7
1	1	2	10
2	1	3	8
3	1	4	12
4	2	4	23
5	3	3	14
	4	1	35
	4	2	37

Sparse Matrix Addition

- Matrix 1: (5x5) +Matrix 2: (5x5) Addition Matrix

	1	2	3
0	5	5	5
1*	1	2	10
2	1	4	12
3	3	3	5
4	4	1	15
5	4	2	12

	1	2	3
0	5	5	5
1*	1	3	8
2	2	4	23
3	3	3	9
4	4	1	20
5	4	2	25

0	5	5	?
1	1	2	10
2			
3			
4			
5			

Sparse Matrix Addition

- Matrix 1: (5x5) +Matrix 2: (5x5) AdditionMatrix

Matrix 1:

	1	2	3
0	5	5	5
1	1	2	10
2*	1	4	12
3	3	3	5
4	4	1	15
5	4	2	12

Matrix 2:

	1	2	3
0	5	5	5
1*	1	3	8
2	2	4	23
3	3	3	9
4	4	1	20
5	4	2	25

AdditionMatrix:

0	5	5	7
1	1	2	10
2	1	3	8
3	1	4	12
4	2	4	23
5	3	3	14
6	4	1	35
7	4	2	37

Sparse Matrix Addition

Perform Sparse Matrix addition of Sparse represented matrix A and B and store it in matrix C

$$\begin{array}{ccc} A & & B \\ \begin{pmatrix} 6 & 7 & 8 \\ 0 & 1 & 1 \\ 2 & 2 & 9 \\ 2 & 3 & 8 \\ 3 & 0 & 3 \\ 4 & 3 & 5 \\ 4 & 4 & 4 \\ 5 & 2 & 2 \\ 5 & 3 & 3 \end{pmatrix} & + & \begin{pmatrix} 6 & 7 & 6 \\ 0 & 1 & 2 \\ 1 & 2 & 7 \\ 2 & 1 & 2 \\ 3 & 0 & 5 \\ 4 & 2 & 4 \\ 5 & 3 & 9 \\ 6 & 4 & 8 \end{pmatrix} \end{array}$$

Sparse Matrix Addition

$$
\overset{A}{\begin{pmatrix}
6 & 7 & 8 \\
0 & 1 & 1 \\
2 & 2 & 9 \\
2 & 3 & 8 \\
3 & 0 & 3 \\
4 & 3 & 5 \\
4 & 4 & 4 \\
5 & 2 & 2 \\
5 & 3 & 3
\end{pmatrix}}
+
\overset{B}{\begin{pmatrix}
6 & 7 & 6 \\
0 & 1 & 2 \\
1 & 2 & 7 \\
2 & 1 & 2 \\
3 & 0 & 5 \\
4 & 2 & 4 \\
5 & 3 & 9 \\
6 & 4 & 8
\end{pmatrix}}
=
\overset{C}{\begin{pmatrix}
6 & 7 & 12 \\
0 & 1 & 3 \\
1 & 2 & 7 \\
2 & 1 & 2 \\
2 & 2 & 9 \\
2 & 3 & 8 \\
3 & 0 & 8 \\
4 & 2 & 4 \\
4 & 3 & 5 \\
4 & 4 & 4 \\
5 & 2 & 2 \\
5 & 3 & 12 \\
6 & 4 & 9
\end{pmatrix}}
$$

Algorithm

- Write algorithm for sparse matrix addition

Algorithm for sparse matrix addition


```
1.Let A, B be the matrices to be added and stored in C
2.Let M and N be number of non-zero entries in A and B respectively.
3.Let i, j, and k be the three index variables used for the rows of A,
  B, and C respectively.
4.Let i = j = k = 1, M = A[0][2], N = B[0][2]
5.C[0][0] = A[0][0]
  C[0][1] = A[0][1]
6.while(i ≤ M and j ≤ N) do
  begin
    if(A[i][0] = B[j][0])                //if1
      if(A[i][1] = B[j][1])              //if2
      then
      begin
        C[k][0] = A[i][0]
        C[k][1] = A[i][1]
        C[k][2] = A[i][2] + B[j][2]
        i = i + 1, j = j + 1, k = k + 1
      end
      else if(A[i][1] < B[j][1])         //if3 and else for if2
      then
      begin
        C[k][0] = A[i][0]
        C[k][1] = A[i][1]
        C[k][2] = A[i][2]
        k = k + 1, i = i + 1
      end
      else                               //else for if3
      begin
        C[k][0] = B[j][0]
        C[k][1] = B[j][1]
        C[k][2] = B[j][2]
        j = j + 1, k = k + 1
      end
    else if(A[i][0] < B[j][0])           //if4 and else for if1
    then
    begin
      C[k][0] = A[i][0]
      C[k][1] = A[i][1]
      C[k][2] = A[i][2]
      k = k + 1, i = i + 1
    end
```

Algorithm for sparse matrix addition continued


```
    else                                     //else for if4
    begin
      C[k][0] = B[j][0]
      C[k][1] = B[j][1]
      C[k][2] = B[j][2]
      K = k + 1, j = j + 1
    end
  end while
 7. while(i < = M) do
    begin
      C[k][0] = A[i][0]
      C[k][1] = A[i][1]
      C[k][2] = A[i][2]
      k = k + 1, i = i + 1
    end
 8. while(j <= N) do
    begin
      C[k][0] = B[j][0]
      C[k][1] = B[j][1]
      C[k][2] = B[j][2]
      k = k + 1, j = j + 1
    end
 9. C[0][2] = k
10. stop
```

Example : Transpose of matrices

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix}_{3 \times 4} \qquad A^T = \begin{pmatrix} 1 & 5 & 9 \\ 2 & 6 & 10 \\ 3 & 7 & 11 \\ 4 & 8 & 12 \end{pmatrix}_{4 \times 3}$$

(a)

$$B = \begin{pmatrix} \mathbf{6} & \mathbf{7} & \mathbf{5} \\ 1 & 2 & 7 \\ 2 & 4 & 2 \\ 3 & 6 & 5 \\ 5 & 0 & 4 \\ 5 & 3 & 9 \\ 6 & 1 & 8 \end{pmatrix} \qquad B^T = \begin{pmatrix} \mathbf{7} & \mathbf{6} & \mathbf{5} \\ 2 & 1 & 7 \\ 4 & 2 & 2 \\ 6 & 3 & 5 \\ 0 & 5 & 4 \\ 3 & 5 & 9 \\ 1 & 6 & 8 \end{pmatrix}$$

(b)

Fig. 2.33 Transpose of matrices (a) Conventional matrix and its transpose (b) Sparse matrix and its transpose

In the conventional approach, by Interchanging rows and columns, we get the transpose of the matrix as the elements at position [i][j] and [j][i] are swapped.

Let m and n be the number of rows and columns for matrix A. The transpose of A can be obtained using the following code.

```
for(i = 1; i < m; i++)
        for(j = 1; j < n; j++)
                A[j][i] = A[i][j];
```

Time complexity of this technique is *O(mn)*.

Simple Transpose

- Sparse Matrix 1: (5x5)

	0	1	2
0	5	5	5
1	1	0	10
2	1	4	12
3	3	3	5
4	4	1	15
5	4	2	12

0	5	5	5
1	0	4	15
2	2	1	10
3	2	4	12
4	3	3	5
5	4	1	12

Simple Transpose

- Sparse Matrix 1: (5x5)

	1	2	3
0	5	5	5
1	1	2	10
2	1	4	12
3	3	3	5
4	4	1	15
5	4	2	12

0	5	5	5
1	1	4	15
2	2	1	10
3	2	4	12
4	3	3	5
5	4	1	12

Question

Convert following sparse matrix of size 3 *4 into its transpose.

$$A = \begin{pmatrix} 3 & 4 & 5 \\ 1 & 2 & 12 \\ 2 & 1 & 21 \\ 2 & 3 & 23 \\ 3 & 1 & 31 \\ 3 & 4 & 34 \end{pmatrix}$$

Answer

- Convert following sparse matrix of size 3 *4 into its transpose.

$$A = \begin{pmatrix} 3 & 4 & 5 \\ 1 & 2 & 12 \\ 2 & 1 & 21 \\ 2 & 3 & 23 \\ 3 & 1 & 31 \\ 3 & 4 & 34 \end{pmatrix} \qquad B = A^T = \begin{pmatrix} 4 & 3 & 5 \\ 1 & 2 & 21 \\ 1 & 3 & 31 \\ 2 & 1 & 12 \\ 3 & 2 & 23 \\ 4 & 3 & 34 \end{pmatrix}$$

Fig. 2.34 Simple transpose

Algorithm simple transpose

1. Row = A[0][0], Col = A[0][1] and T = A[0][2]
2. B[0][0] = Col , B[0][1] = Row and B[0][2] = T
3. if T = 0 goto step(5)
4. Let i = 1

```
for j = 0 to Col-1 do
        for k = 1 to T do
                if(A[k][1] = j)
                begin
                    B[i][0] = A[k][1]
                    B[i][1] = A[k][0]
                    B[i][2] = A[k][2]
                    i = i + 1
                end
```

5. Stop

Time Complexity of simple transpose

- Step 4 of Algorithm is repeated T times for each column.
- The time complexity is *O(nT) where n is the number of columns in matrix A and T is number of non-zero elements* in the matrix.
- In a matrix, when all data is relevant, that is, all data members are non-zero, then T = $m * n$.
- Now, the time complexity will be $O(n \cdot T) = O(n \cdot mn) = O(m.n.n)$
- *which is worse than* the conventional transpose with time complexity $O(mn)$.

Fast Transpose

- Let A be a sparse matrix of size *m * n with T non-zero elements*. *Its transpose will be* stored in matrix B.
- Let Freq and RowStartPos be two one-dimensional arrays of size *n*.
- In Freq array, the frequency count of each column in matrix A is stored.
- and RowStart- Pos will be computed and stored at the position where each row entry of matrix A is to be inserted in matrix B.
- Then, the RowStartPos is computed using Freq.

Fast Transpose

```
1. Row = A[0][0]
   Col = A[0][1]
   M = A[0][2]
2. B[0][0] = Col
   B[0][1] = Row
   B[0][2] = M
3. if M = 0 then goto step 9
4. for i = 0 to Col - 1 do
       Freq[i] = 0 {Here Freq array stores the frequency count of each
       column, initially set to 0}
5. for k = 1 to M do
       Tmp = A[k][1]
       Freq[Tmp] = Freq [Tmp] + 1;
6. RowStartPos[0] = 1 {We shall start storing elements in B matrix
   from 2nd row that is B[1][] onwards}
7. for j = 1 to Col - 1 do                          Freq[j-1]
       RowStartPos[j] = RowStartPos[j - 1] + ~~RowStartPos[j - 1]~~;
          {Here RowStartPos n matrix gives the position to place an
           element in resultant matrix}
8. for i = 1 to M + 1 do
   begin
       k = RowStartPos[A[i][1]]
       B[k][0] = A[i][1]
       B[k][1] = A[i][0]
       B[k][2] = A[i][2]
       RowStartPos[k] = RowStartPos[k] + 1
   end
```

B[ ]

0	6	7	8
1	1	2	17
2	2	4	12
3	3	6	50
4	5	0	49
5	5	3	92
6	6	0	38
7	6	1	1.1
8	6	3	15

(a)

Freq [ ]

0	2
1	1
2	1
3	2
4	1
5	0
6	1

(b)

RowStartPos [ ]

0	~~1~~ 2
1	3
2	4
3	~~5~~ 6
4	7
5	8
6	8

(c)

B^T [ ]

0	7	6	8
1	0	5	49
2	0	6	38
3	1	6	11
4	2	1	17
5	3	5	92
6	3	6	15
7	4	2	12
8	6	3	50

(d)

Thank You