Kartikeya Kumar
Panel - E
Roll no. - 38

Batch : E2
PRN: 1032232023

Page No. 01
Date 06 08 2024

DATA STRUCTURES

EXPERIMENT NO. - 2 SPARSE MATRIX OPERATIONS

PROBLEM STATEMENT

Write a C program for sparse matrix realization and operations on it — Simple Transpose and Fast Transpose.

OBJECTIVE

1. To study the concept of sparse matrix, how it is stored and displayed.
2. To understand the implementation of sparse matrix operations — simple transpose and fast transpose.

IMPLEMENTATION

☆ Platform

→ 64-bit Open source Linux or its derivatives.
→ Open source C Programming tool like gcc/Eclipse Editor.

☆ Pseudo Code

• Sparse Matrix Realization (Compact Form)

```
algorithm compact(A, m, n, B)
{                                   // A is input matrix
    B(0,0) = m                      // m = no. of rows
    B(0,1) = n                      // n = no. of columns
    k = 1                           // B = compact form
```

```
for i=0 to m
    for j=0 to n
        if A(i,j) != 0
        {
            B(k,0) = i
            B(k,1) = j
            B(k,2) = A(i,j)
            k++
        }
    B(0,2) = k-1
}
```

- Simple Transpose

```
algorithm simpleTranspose(B, C)
{
    (m, n, t) = (B(0,0), B(0,1), B(0,2))
    (C(0,0), C(0,1), C(0,2)) = (n, m, t)
    if t<=0 then return
    q = 1
    for col=0 to n
        for p=1 to t
            if B(p,1) = col
            {
                (C(q,0), C(q,1), C(q,2)) = (B(p,1), B(p,0), B(p,2))
                q++
            }
}
```

// B is Compact form
// C is Simple Transpose
// m is no. of rows, n is no. of columns, t is no. of non-zero elements

- Fast Transpose

```
algorithm fastTranspose (A, B)
{                                                         // A is compact form
    (m, n, t) = (A(0,0), A(0,1), A(0,2))                  // B is to be transpose
    (B(0,0), B(0,1), B(0,2)) = (n, m, t)                  // m is no. of rows
    if t<=0 then return                                   // n is no. of columns
    for i=0 to n                                          // t is no. of non-zero elements
        S(i) = 0
    for i=1 to t
        S(A(i,1)) = S(A(i,1)) + 1
    T(0) = 1
    for i=1 to n
        T(i) = T(i-1) + S(i-1)
    for i=1 to t
    {
        j = A(i,1)
        (B(T(j), 0), B(T(j), 1), B(T(j), 2)) = (A(i,1), A(i,0), A(i,2))
        T(j) = T(j) + 1
    }
}
```

☆ Time Complexity

- Compact Form : $f(m,n) = O(m \cdot n)$
- Simple Transpose : $f(n,t) = O(n \cdot t)$
- Fast Transpose : $f(n,t) = O(n+t)$

☆ Conclusion

Thus, implemented sparse matrix operations assignment. This system is able to perform different operations on sparse matrices such as simple and fast transpose and their time complexities.

☆ FAQs

1. What is a sparse matrix? List its applications.

Ans. Any $m \times n$ matrix which contains a large number of zeros is called as sparse matrix.

Some applications of sparse matrix are:

- Scientic computing – finite element analysis
  partial differential equations
  computational fluid dynamics
- Image and signal processing – image compression
  computer vision
- Machine learning – natural-languge processing (term-document matrix)
  recommender systems
- Graph theory – adjacency matrices of sparse graphs
- Network analysis – social networks
  transportation networks

2. Represent sparse matrices with suitable data structures. Explain with an example simple and fast transpose.

Ans. Sparse matrices can be efficiently represented using two primary methods:

① Array Representation
→ Each element contains row index, column index and value
→ Example:

$$\begin{bmatrix} 0 & 0 & 3 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Representation:

no. of rows	no. of columns	no. of non-zero elements
3	4	3
0	2	3
1	~~8~~1	6
2	3	1

② Linked List Representation
→ Each non-zero element is a node in a linked list.
→ The list can be organized by row, column or a combination.
→ Example:

$$\begin{bmatrix} 0 & 0 & 3 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Representation: Row 0: (2, 3) → NULL
Row 1: (1, 6) → NULL
Row 2: (3, 1) → NULL

There are two methods to obtain the transpose of a ~~matrix~~ sparse matrix in array representation compact form — Simple transpose and Fast transpose.

- ~~Sparse Matrix~~ Simple Transpose
  → It is computed by traversing the second column of sparse matrix and searching for column indices in sequence from 0 to ~~no~~ (number of columns − 1) then copying the rows in sequence to transpose
  → Example:

Given:

3	3	5
0	0	2
0	1	-3
0	2	4
1	1	6
2	0	7

→

3	3	5
0	0	2
0	2	7
1	0	-3
1	1	6
2	0	4

- Fast Transpose
  → It is computed by calculating the frequency of each column and using the frequency of each column ~~to find~~ value to find location of each row in transpose matrix, then copying elements into their correct positions into transpose.
  → Example:

Given:

3	3	5
0	0	2
0	1	-3
0	2	4
1	1	6
2	0	7

→

S = (2, 2, 1)

T = (0, 3, 5)

	3	3	5
0 →	0	0	2
	0	2	7
1 →	1	0	-3
	1	1	6
2 →	2	0	4

3. Find out the addition of two sparse matrices in triplet form and also find simple and ~~tran~~ fast transpose.

M1 =

4	5	6
0	3	5
1	3	8
1	4	45
2	3	4
3	2	45
4	1	2

M2 =

4	5	6
0	3	7
0	4	6
1	4	4
2	1	8
3	2	45
4	4	21

Ans. M1 + M2 =

4	5	9
0	3	12
0	4	6
1	3	8
1	4	49
2	1	8
2	3	4
3	2	90
4	1	2
4	4	21

Simple Transpose:

5	4	9
1	2	8
1	4	2
2	3	90
3	0	12
3	1	8
3	2	4
4	0	6
4	1	49
4	4	21

Fast ~~Simple~~ Transpose:

$S = (0, 2, 1, 3, 3)$

$T = (1, 1, 3, 4, 7)$

	5	4	9
0, 1 →	1	2	8
	1	4	2
2 →	2	3	90
3 →	3	0	12
	3	1	8
	3	2	4
4 →	4	0	6
	4	1	49
	4	4	21