

The Single-vector and Multi-vector Mixed Compressed Storage of Tangent Matrix

Wenjiao Da^{1, a}, Xiuli Wang^{1, b}, Jing Wen^{2, c} and Han Zhang^{1, d}

¹School of Electronics and Information Engineering, Lanzhou City University, Lanzhou 730070, China;

²Institute of Information Technology Application, Lanzhou City University, Lanzhou 730070, China.

^ada_wenjiao@qq.com, ^b674230540@qq.com, ^cjingwen@qq.com, ^d3351735917@qq.com

Keywords: The compressed storage, single-vector, multi-vector, the tangent matrix.

Abstract. Matrix is a mathematical object, commonly used in scientific computing and engineering calculation. We are not interested in data itself in the data structure, but interested in how to store the elements in the matrix, and make the various operations can run effectively. The main purpose of the compressed storage is to make more of the same nonzero elements share the same storage unit according to the distribution of matrix element, while the zero elements don't allocate storage space. In this paper, we studied the single-vector and multi-vector compressed storage problems of tangent matrix, and obtained the row and column priority compressed storage address mapping function. These two kinds of compressed storage have a high compression ratio. These conclusions hope to provide the basis theory of data compression storage for the scientific research workers.

1. Introduction

Matrix is a mathematical object, commonly used in scientific computing and engineering calculation. We are not interested in data type or value in the data structure, but how to store the elements in the matrix. When programming in a high-level language, often use a two-dimensional array to store the elements in the matrix. If adopt this method of storage, we can random access each data element, thus can easily realize operations of the matrix. But, when there are a large number of zero elements in the matrix and have regular distribution, if we still use a two-dimensional array to store the matrix, a particular element will consume large amounts of storage unit. For high order matrix, the storage method is not only waste storage unit, but also takes a lot of time for invalid computation, it is obviously not desirable. In order to save the storage space, we need to compress storage for such matrix.

The main purpose of the compressed storage is to make more of the same nonzero elements share the same storage unit according to the distribution of matrix element, while the zero elements don't allocate storage space. In this paper, we studied the compressed storage problems of tangent matrix, and obtained the corresponding storage address mapping function.

2. Definition of Tangent Matrix

If a tangent matrix is a cut matrix, in the cut $((a,b) (c,d))$, a, b, c, d should also meet the following relationship.

$$1 < a \leq m, b = 1, c = m, d < n$$

For a given m rows n columns ($m < n$) of the matrix A , the original matrix can be divided into two parts, the first element of the dividing line is the i'' ($2 \leq i'' \leq m-1$), and it is in the first column. There are some different data elements in the upper part above the diagonal, the data elements in the lower part of the diagonal are all zero or the same integer. As shown in figure 1.

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ 0 & a_{32} & a_{33} & a_{34} \end{bmatrix} \quad A_2 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ 0 & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\ 0 & 0 & a_{43} & a_{44} & a_{45} & a_{46} \end{bmatrix} \quad A_3 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{17} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & a_{27} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} & a_{37} \\ 0 & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & a_{47} \\ 0 & 0 & a_{53} & a_{54} & a_{55} & a_{56} & a_{57} \end{bmatrix}$$

Fig.1 the results of compression storage at n and m

For a tangent matrix which has m row n column, data elements can be divided into three parts, they can be compressed storage respectively, one part can be compressed storage into a one-dimensional array of B . The other two parts can be respectively compressed into the two-dimensional arrays C_1 and C_2 , they have different sizes.

When the first element of the dividing line is the i'' , and it is in the first column, that is $a_{i''1}$. It can be compressed to the one-dimensional array B , the other two parts can be compressed storage into the two-dimensional arrays C_1 , C_2 . the size of C_1 is $((i''-1) \times n)$. C_2 has $m-i''+1$ rows and $n-m-i''+1$ columns, the size of C_2 is:

$$k = \frac{(m - i'' + 1 + 1) \times (m - i'' + 1)}{2}$$

3. The Single-vector and Multi-Vector Mixed Compressed Storage of Tangent Matrix

For a given m rows n columns of the matrix A , it can also be compressed by single-vector and multi-vector mixed compressed storage. First of all, we can observe the following simple examples:

For $m=3, n=4$ on the tangent matrix, as shown in figure 2:

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ 0 & a_{32} & a_{33} & a_{34} \end{bmatrix} \quad C_1 = [a_{11} \quad a_{12} \quad a_{13} \quad a_{14}]$$

$$C_2 = \begin{bmatrix} a_{23} & a_{24} \\ a_{33} & a_{34} \end{bmatrix} \quad B = \begin{bmatrix} a_{21} & a_{22} & a_{32} \\ 0 & 1 & 2 \end{bmatrix}$$

Fig.2 The results of compression storage at $n=4$ and $m=3$

For the tangent matrix as shown in figure 2, the data elements $a_{i,j}$ in the A_1 matrix can be divided into three parts of compressed storage:

One part of $i < 2$ can be compressed into a 2-1 rows 4 columns two-dimensional array C_1 , the mapping relationship between $a_{i,j}$ and $c_{i'j'}$ is:

$$\begin{cases} i' = i \\ j' = j \end{cases} \quad (1)$$

One part of $i \geq 2, j > 3-2+1$ can be compressed into a 3-2+1 rows 4-3+2-1 columns two-dimensional array C_2 , the mapping relationship between a_{ij} and $c_{i'j'}$ is:

$$\begin{cases} i' = i - 2 + 1 \\ j' = j - (3 - 2 + 1) \end{cases} \quad (2)$$

One part of $i \geq 2, j \leq 3-2+1$ can be compressed into one-dimensional array B , the space size is k .

$$k = \frac{(3 - 2 + 1 + 1) \times (3 - 2 + 1)}{2}$$

For a data element $a_{i,j}$ in tangent matrix A_1 , it compressed storage into one-dimensional array B , if the element of $a_{i,j}$ in A response element is b_k . It can be seen that k is a function of (i, j, i'', m, n) , k is a function that can be computed:

$$k = \frac{(3-2+1+3-i+1) \times (i-2)}{2} + j - (i-2) \quad (3)$$

For $m=4, n=5$ on the tangent matrix, as shown in Figure3:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ 0 & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix} \quad C_1 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \end{bmatrix}$$

$$C_2 = \begin{bmatrix} a_{33} & a_{34} & a_{35} \\ a_{43} & a_{44} & a_{45} \end{bmatrix} \quad B = \begin{bmatrix} a_{31} & a_{32} & a_{42} \\ 0 & 1 & 2 \end{bmatrix}$$

Fig.3 The results of compression storage at $n=4$ and $m=5$

For the tangent matrix as shown in figure 7, the data elements a_{ij} in the A matrix can be divided into three parts of compressed storage:

One part of $i < 3, j \leq 4-3+1$ can be compressed storage into a 3-1 rows 5 columns two-dimensional array C_1 , the mapping relationship between a_{ij} and $c_{i'j'}$ is:

$$\begin{cases} i' = i \\ j' = j \end{cases} \quad (4)$$

One part of $i \geq 3, j > 4-3+1$ can be compressed storage into a 4-3+1 rows 5-4+3-1 columns two-dimensional array C_2 , the mapping relationship between a_{ij} and $c_{i'j'}$ is:

$$\begin{cases} i' = i - i'' + 1 \\ j' = j - (m - i'' + 1) \end{cases} \quad (5)$$

One part of $i \geq 3, j > 4-3+1$ can be compressed into one-dimensional array B, the space size is k.

$$k = \frac{(4-3+1+1) \times (4-3+1)}{2}$$

For a data element a_{ij} in tangent matrix A_1 , it compressed storage into one-dimensional array B, if the element of a_{ij} in A response element is b_k . It can be seen that k is a function of (i, j, i'', m, n) k is a function that can be computed:

$$k = \frac{(4-3+1+4-i+1) \times (i-3)}{2} + j - (i-3) \quad (6)$$

Through the observation and analysis of several simple examples, as well as some specific reasoning and calculation, we can get the following conclusion:

For a given m rows n columns of the matrix A, it can also be compressed by single-vector and multi-vector mixed compressed storage. The data elements a_{ij} in the A matrix can be divided into three parts of compressed storage:

One part of $i < i''$ can be compressed storage into a $i'-1$ rows n columns two-dimensional array C_1 , the mapping relationship between a_{ij} and $c_{i'j'}$ is:

$$\begin{cases} i' = i \\ j' = j \end{cases} \quad (7)$$

One part of $i \geq 3, j > 4-3+1$ can be compressed storage into a $m-i''+1$ rows $n-m-i''+1$ columns two-dimensional array C_2 , the mapping relationship between a_{ij} and $c_{i'j'}$ is:

$$\begin{cases} i' = i - i'' + 1 \\ j' = j - (m - i'' + 1) \end{cases} \quad (8)$$

One part of $i \geq 3, j > 4-3+1$ can be compressed into a space size of k of one-dimensional array B.

$$k = \frac{(m-i''+1+1) \times (m-i''+1)}{2}$$

For a data element a_{ij} in tangent matrix A, it compressed storage into one-dimensional array B, if the element of a_{ij} in A response element is b_k . It can be seen that k is a function of (i, j, i'', m, n) , k is a function that can be computed:

$$k = \frac{(2 \times m - i'' - i + 2) \times (i - i'')}{2} + j - (i - i'') \quad (9)$$

4. Summary

In this paper, we studied the single-vector and multi-vector compressed storage problems of tangent matrix, and obtained the single-vector and multi-vector compressed storage address mapping function of the compress storage by using single-vector and multi vector mixed compression. Multi-vector compressed storage about the tangent matrix (formula 8 and formula 9), also obtained the single-vector compressed storage address mapping function of the single-vector compressed storage about the tangent matrix (formula 7).

In an n order matrix with r sub matrices (m order), the data elements can be compressed into a multi-vector, its compression ratio can be obtained. $CR = (1 - ((1+m) * m * r / 2) / n^2) * 100\%$.

In a 100-order square matrix, the number of 10 order matrices is 10. Its compression ratio is: $CS = (1 - ((1+10) * 10 * 10 / 2) / 100^2) * 100\% = 94.5\%$.

It can be seen that these two kinds of compressed storage have a high compression ratio. These conclusions hope to provide the theory basis of data compression storage for the data processing and scientific computing algorithm design.

References

- [1] Thomas H.Cormen, Charles E.Leiserson, Ronald L.Rivest, Clifford Stein. Introduction to Algorithm, the third edition. The MIT Press, 2009.
- [2] Alfred V.Aho, John E.Hopcroft, and Jeffrey D.Ullman. Data structures and Algorithms. Addison-Wesley, 1983.
- [3] Bondy J A and Marty U S R, Graph Theory with Applications, The Macmillan Press Ltd, New York, 1976.
- [4] Donald E.Knuth. Fundamental Algorithms, volume 1 of The Art of Computer Programming. Addison- Wesley, 1968. Third edition, 1997.
- [5] Donald E.Knuth. Seminumerical Algorithms, volume 2 of The Art of Computer Programming. Addison- Wesley, 1969. Third edition, 1997.
- [6] Donald E.Knuth. Sorting and Searching, volume 3 of The Art of Computer Programming. Addison- Wesley, 1973. Second edition, 1998.
- [7] Don Coppersmith and Shmuel Winograd. Matrix Multiplication via arithmetic progression. Journal of Symbolic Computation, 9(3):251-280, 1990.
- [8] T.C.Hu and M.T.Shing. Computation of Matrix chian products. Part 1, SIAM Journal on Computing, 11(2):362-373, 1982.
- [9] T.C.Hu and M.T.Shing. Computation of Matrix chian products. Part 2, SIAM Journal on Computing, 13(2):228-251, 1984.
- [10] Mark Allen Weiss. Data Structures and Algorithm analysis in Java. Addison-Wesley, third edition, 2007.
- [11] Zhiguo Ren, Data Structure (C Language Description), Science Press, Peking China, 2016.
- [12] Langr D, Tvrdik P. Evaluation Criteria for Sparse Matrix Storage Formats[J]. IEEE Transactions on Parallel & Distributed Systems, 2016, 27(2):428-440.
- [13] Ordonez C, Zhang Y, Cabrera W. The Gamma Matrix to Summarize Dense and Sparse Data Sets for Big Data Analytics[J]. IEEE Transactions on Knowledge & Data Engineering, 2016:1-1.
- [14] Wilson Rodrí, guezCalderó. Fortran application to solve systems from NLA using compact storage[J]. Dyna, 2015, 82(192):249-256.