The Single-vector Compressed Storage of Tangent Matrix

Yongyan Hou^{1, a}, Zhiguo Ren^{2, b} and Lanqing Wu^{1, c}

¹Zhixing college of Northwest normal University, Lanzhou 730070, China;

²School of Electronics and Information Engineering, Lanzhou City University, Lanzhou 730070, China.

^a498264224@qq.com, ^bren_zhiguo@qq.com, ^c2874767408@qq.com

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

Abstract. Matrix is a mathematical object, commonly used in scientific computing and engineering calculation. In the data structure we are not interested in data itself, but 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 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.

1. Tangent matrix is a cut matrix.
2. In cut $((a, b) (c, d))$, a, b, c, d should 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, the data elements can be compressed into a one-dimensional array B by row and column priority.

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 size is:

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

The Single-vector Compressed Storage of Tangent Matrix

2. Row Priority Compressed Storage

For $m=3, n=4$ on the tangent matrix, as shown in figure2:

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ 0 & a_{32} & a_{33} & a_{34} \end{bmatrix} \quad B = \begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline a_{11} & a_{12} & a_{13} & a_{14} & a_{21} & a_{22} & a_{23} & a_{24} & a_{32} & a_{33} & a_{34} \\ \hline 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \hline \end{array}$$

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

For the tangent matrix as shown in figure 2, it can be compressed into one-dimensional array B, the size is 11. For a data element a_{ij} in tangent matrix A_1 , it compressed into one-dimensional array B in row priority, if the element of a_{ij} in A_1 response element is b_k . It can be seen that k is a function of (i,j,i'',n) , k is a function that can be computed:

$$k = \begin{cases} (i-1) \times 4 + j, & i < 2 \\ (2-1) \times 4 + \frac{(4+4+2-i+1) \times (i-2)}{2} + j - i + 2, & i \geq 2 \end{cases} \quad (1)$$

For $m=4, n=5$ on the tangent matrix, as shown in figure3:

$$A_2 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ 0 & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix} \quad B = \begin{array}{|c|c|c|c|c|c|c|c|c|c|} \hline a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ \hline 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{42} & a_{43} & a_{44} & a_{45} \\ \hline 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 \\ \hline \end{array}$$

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

For the tangent matrix as shown in figure 3, it can be compressed into one-dimensional array B, the space size is 19. For a data element a_{ij} in tangent matrix A_2 , it compressed into one-dimensional array B, if the element of a_{ij} in A_2 response element is b_k . It can be seen that k is a function of (i,j,i'',n) , k is a function that can be computed:

$$k = \begin{cases} (i-1) \times 5 + j, & i < 3 \\ (3-1) \times 5 + \frac{(5+5+3-i+1) \times (i-3)}{2} + j - i + 3, & i \geq 3 \end{cases} \quad (2)$$

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

For a tangent matrix A which has m rows n columns, it can be compressed into the one-dimensional array B, the space size of k is:

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

For a data element a_{ij} in tangent matrix A, it compressed 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'', n) , k is a function that can be computed:

$$k = \begin{cases} (i-1) \times n + j, & i < i'' \\ (i''-1) \times n + \frac{(2 \times n + i'' - i + 1) \times (i - i'')}{2} + j - i + i'', & i \geq i'' \end{cases} \quad (4)$$

3. Column Priority Compressed Storage

For $m=3, n=4$ on the tangent matrix, as shown in Figure4:

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ 0 & a_{32} & a_{33} & a_{34} \end{bmatrix} \quad B = \begin{bmatrix} a_{11} & a_{21} & a_{12} & a_{22} & a_{32} & a_{13} & a_{23} & a_{33} & a_{14} & a_{24} & a_{34} \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 1 \end{bmatrix}$$

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

For the tangent matrix as shown in Figure 4, it can be compressed into one-dimensional array B, the space size is 11. For a data element a_{ij} in tangent matrix A_1 , it compressed into one-dimensional array B, if the element of a_{ij} in A_1 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:

$$k = \begin{cases} \frac{(2 + j + 2 - 1 - 1) \times (j - 1)}{2} + i, & i \leq 3 - 2 + 1 \\ \frac{(2 + 3) \times (4 - 3 - 2 + 2)}{2} + [j - (4 - 3 - 2 + 2)] \times 3 + i, & i \geq 3 - 2 + 1 \end{cases} \quad (5)$$

For $m=3, n=4$ on the tangent matrix, as shown in Figure4:

$$A_2 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ 0 & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix} \quad B = \begin{bmatrix} a_{11} & a_{21} & a_{31} & a_{12} & a_{22} & a_{32} & a_{42} & a_{13} & a_{23} & a_{33} \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ a_{43} & a_{14} & a_{24} & a_{34} & a_{44} & a_{15} & a_{25} & a_{35} & a_{45} \\ 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 \end{bmatrix}$$

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

For the tangent matrix as shown in figure 5, it can be compressed into one-dimensional array B, the space size is 19. For a data element a_{ij} in tangent matrix A_2 , it compressed into one-dimensional array B, if the element of a_{ij} in A_2 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:

$$k = \begin{cases} \frac{(3 + j + 3 - 1 - 1) \times (j - 1)}{2} + i, & i \leq 4 - 3 + 1 \\ \frac{(3 + 4) \times (5 - 4 - 3 + 2)}{2} + [j - (5 - 4 - 3 + 2)] \times 4 + i, & i \geq 4 - 3 + 1 \end{cases} \quad (6)$$

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

For a tangent matrix A which has m rows n columns, it can be compressed into one-dimensional array B, the space size of k is:

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

For a data element a_{ij} in tangent matrix A, it compressed 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:

$$k = \begin{cases} \frac{(i'' + j + i'' - 1 - 1) \times (j - 1)}{2} + i, & i \leq m - i'' + 1 \\ \frac{(i'' + m) \times (n - m - i'' + 2)}{2} + [j - (n - m - i'' + 2)] \times m + i, & i \geq m - i'' + 1 \end{cases} \quad (7)$$

4. Summary

In this paper, we studied the single-vector compressed storage problems of tangent matrix, and obtained the row priority compressed storage address mapping function of the single-vector compressed about the tangent matrix(formula 4), also obtained the column priority compressed storage address mapping function of the single-vector compressed storage about the tangent matrix(formula 7).

In a matrix, the data elements can be compressed with a single-vector, its compression ratio can be obtained. $CR = (1 - ((2 \cdot n - m + i'') \cdot (m - i'' + 1) / 2 + (i'' - 1) \cdot n) / (m \cdot n)) \cdot 100\%$. For example: In a 100 - order square matrix. Its compression ratio is:

$$CR = (1 - ((2 \cdot 100 - 100 + 10) \cdot (100 - 10 + 1) / 2 + (10 - 1) \cdot 100) / (100 \cdot 100)) \cdot 100\% = 40.95\%$$

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]Ordóñez 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.

- [15]Nedaie A, Najafi A A. Polar support vector machine: Single and multiple outputs[J]. Neurocomputing, 2015, 171.
- [16]Bacheha L, Hariharan S, Bouman C, et al. Distributed Signal Decorrelation and Detection in Multi View Camera Networks Using the Vector Sparse Matrix Transform.[J]. IEEE Transactions on Image Processing, 2015, 24(12):6011-6024.