Analysis of the Time Complexity of Strassen Algorithm

Xiang Wang
School of Information and Electronic Engineering
Tianjin Vocational Institute
Tianjin, China
e-mail: pingfan6699@163.com

***Abstract*—Algorithms of matrix multiplication are widely used in software engineering field. Application of Strassen algorithm makes a significant contribution to optimize the algorithm . Therefore, thorough study based on time complexity of matrix multiplication algorithm is very important. This paper talks about the time complexity of Strassen algorithm and general algorithm for matrix multiplication, and makes a comparison between the two algorithm routines so as to discuss the advantages and disadvantages of Strassen algorithm. The rational utilization plan of matrix multiplication algorithms is also discussed.**

Keywords-algorithms for matrix multiplication; time complexity; Strassen algorithm

I. A GENERAL ALGORITHM FOR MATRIX MULTIPLICATION

Let A=(a_{ij})m × n be an m × n matrix, and let B=(b_{ij})n × p be an n × p matrix. Given an (m×p) matrix A with r row partitions and s column partitions and a (p×n) matrix with s row partitions and t column partitions that are compatible with the partitions of A, the matrix product C=A×B. generally we can explain the matrix multiplication through the following expression, i.e.,

$$A=\begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1s} \\ A_{21} & A_{22} & \cdots & A_{2s} \\ \vdots & \vdots & & \vdots \\ A_{r1} & A_{r2} & \cdots & A_{rs} \end{pmatrix} B=\begin{pmatrix} B_{11} & B_{12} & \cdots & B_{1t} \\ B_{21} & B_{22} & \cdots & B_{2t} \\ \vdots & \vdots & & \vdots \\ B_{s1} & B_{s2} & \cdots & B_{st} \end{pmatrix}.$$

$$AB=\begin{pmatrix} C_{11} & C_{12} & \cdots & C_{1t} \\ C_{21} & C_{22} & \cdots & C_{2t} \\ \vdots & \vdots & & \vdots \\ C_{r1} & C_{r2} & \cdots & C_{rt} \end{pmatrix} \quad (1)$$

Thus AB can be formed blockwise, yielding as an (m×p) matrix with r row partitions and t column partitions. The matrices in your matrix are calculated by multiplying:

$$C_{ij}=\sum_{\beta=1}^{s} A_{i\beta}B_{\beta j} \text{ and i=1,…,r;j=1,…,t.} \quad (2)$$

The function of matrix multiplication is as follows:

```
template<class Type>
Type M (Type a[][N], Type b[][N],int m, int n, int p)
{
 Type c[][N];
 for(int k=0;k<m;k++ )
  for(int u=0;u<n;u++ )
    for(int v=0;v<p;v++)
      c[k][u]=c[k][u]+a[k][v]*b[v][u];
 return *c;
}
```

We can obtain the following conclusion very easily: time complexity of a general algorithm for matrix multiplication depends on the rows and columns of matrix A and matrix B., and generally we can explain the time complexity through the following expression, i.e., O(m×n×p). If matrix A and matrix B are both n × n matrices, A×B is just a special case of matrix multiplication. And if n is a power of 2, both matrix A and matrix B can be partitioned into 4 n/2×n/2 blocks. The partitioned matrix can then be written as

$$A=\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \qquad B=\begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

$$AB=\begin{pmatrix} A_{11}B_{11}+A_{12}B_{21} & A_{11}B_{12}+A_{12}B_{22} \\ A_{21}B_{11}+A_{22}B_{21} & A_{21}B_{12}+A_{22}B_{22} \end{pmatrix} \quad (3)$$

In this case, if we adopt the general algorithm for matrix multiplication, then we need make eight n/2×n/2 Matrix Multiplications and four Matrix additions (or matrix subtractions). Let T(n) be time complexity of matrix multiplication, we can explain the T(n) through the following recursive expression, i.e.,

$$T(n)=\begin{cases} 1 & n=1 \\ 8T\left(\frac{n}{2}\right)+4\left(\frac{n}{2}\right)^2 & n>1 \end{cases} \quad (4)$$

and generally we can explain the time complexity through the following expression, i.e., O(n^3).

II. Analysis Of Time Complexity Of The Strassen Algorithm

Strassen algorithm used for matrix multiplication, which is invented by the German mathematician Volker Strassen in 1969, is considered as one of the fastest and best matrix multiplication algorithms. We define 7 new matrices:

$F1=A_{11}(B_{12}-B_{22})$,
$F2=(A_{11}+A_{12})B_{22}$,
$F3=(A_{21}+A_{22})B_{11}$,
$F4=A_{22}(B_{21}-B_{11})$,
$F5=(A_{11}+A_{22})(B_{11}+B_{22})$,
$F6=(A_{12}-A_{22})(B_{21}+B_{22})$,
$F7=(A_{11}-A_{21})(B_{11}+B_{12})$
Thus

$$AB=\begin{pmatrix} F_5+F_4-F_2+F_6 & F_1+F_2 \\ F_3+F_4 & F_1+F_5-F_3-F_7 \end{pmatrix} \tag{5}$$

The numbers of n/2 ×n/2 matrix multiplication may set back to 7 through the Strassen algorithm, when compared with the general algorithm for matrix multiplication. This meantime, there are 18 matrix addition or matrix subtraction. As n is a power of 2, if let k be nonnegative integer, then n is k-th power of 2. If k=0, then the time complexity of matrix multiplication is 1. If k>0, generally we can explain time complexity of matrix multiplication through the following recursive expression,

i.e., $$7\mathrm{T}\left(\frac{\mathrm{n}}{2}\right)+18\left(\frac{\mathrm{n}}{2}\right)^2 \tag{6}$$

so we can explain the T(n) through the following recursive expression,

i.e., $$T(n)=\begin{cases} 1 & \mathrm{n}=1 \\ 7\mathrm{T}\left(\frac{\mathrm{n}}{2}\right)+18\left(\frac{\mathrm{n}}{2}\right)^2 & \mathrm{n}>1 \end{cases} \tag{7}$$

If n>1, then

$$\mathrm{T(n)}=7\mathrm{T}\left(\frac{\mathrm{n}}{2}\right)+18\left(\frac{\mathrm{n}}{2}\right)^2 \tag{8}$$

We come next to the analysis of equation(8). We replace variable n with 2^k.

$$\begin{aligned}
\mathrm{T(n)} &= 7\mathrm{T}\left(\frac{2^{\mathrm{k}}}{2}\right)+18\left(\frac{2^{\mathrm{k}}}{2}\right)^2 \\
&= 7\mathrm{T}\left(2^{\mathrm{k-1}}\right)+18\left(2^{\mathrm{k-1}}\right)^2 \\
&= 7\left(7\mathrm{T}\left(2^{\mathrm{k-2}}\right)+18\left(2^{\mathrm{k-2}}\right)^2\right)+18\left(2^{\mathrm{k-1}}\right)^2 \\
&= 7^2\mathrm{T}\left(2^{\mathrm{k-2}}\right)+7\times18\left(2^{\mathrm{k-2}}\right)^2+18\left(2^{\mathrm{k-1}}\right)^2 \\
&= 7^2\left(7\mathrm{T}\left(2^{\mathrm{k-3}}\right)+18\left(2^{\mathrm{k-3}}\right)^2\right) \\
&\quad +7\times18\left(2^{\mathrm{k-2}}\right)^2+18\left(2^{\mathrm{k-1}}\right)^2 \\
&= 7^3\mathrm{T}\left(2^{\mathrm{k-3}}\right)+7^2\times18\left(2^{\mathrm{k-3}}\right)^2 \\
&\quad +7\times18\left(2^{\mathrm{k-2}}\right)^2+18\left(2^{\mathrm{k-1}}\right)^2 \\
&= \cdots=\sum_{i=0}^{k-1}7^i\times18\left(2^{\mathrm{k-i-1}}\right)^2 \\
&= 18\left(2^{\mathrm{k}}\right)^2\sum_{i=0}^{k-1}\frac{7^i}{\left(2^{\mathrm{i+1}}\right)^2} \\
&= \frac{9}{2}\left(2^{\mathrm{k}}\right)^2\sum_{i=0}^{k-1}\left(\frac{7}{4}\right)^i \\
&= \frac{9}{2}\times\left(2^{\mathrm{k}}\right)^2\times\frac{1-\left(\frac{7}{4}\right)^k}{1-\frac{7}{4}} \\
&= 6\times7^k-6\times4^k
\end{aligned}$$

Since

$$\begin{aligned}
7^{\mathrm{k}} &= 7^{\log_2 n} \\
&= n^{\log_n 7^{\log_2 n}} \\
&= n^{\log_2 n\log_n 7} \\
&= n^{\log_2 n\frac{\log_2 7}{\log_2 n}} \\
&= n^{\log_2 7} \\
&\approx n^{2.807355} \\
&\approx n^{2.81}
\end{aligned}$$

and

$$4^k = 2^{2k} = 2^{2\log_2 n}$$
$$= 2^{\log_2 n^2} = n^2 \quad ,$$

Then, we have

$$T(n) \approx \begin{cases} 1 & \text{n} = 1 \\ 6n^{2.81} - 6n^2 & \text{n} > 1 \end{cases} \tag{9}$$

This meantime, we can explain the time complexity through the following expression, i.e., $O(n^{2.81})$. It is evident that the Strassen algorithm was more efficient than the general algorithm for matrix multiplication. In fact, if n is not a power of 2, the matrix can be embedded into another matrix (let the matrix be matrix E) whose number of dimension is a power of 2. In matrix multiplication, if the dimension of matrix E take minimum value within the design requirement above is achieved, the dimension will increase twofold at most. Thus, the time complexity of Strassen algorithm can still be described as $O(n^{2.81})$.

III. Variation Trends Of Time Complexity Of The Strassen Algorithm

For the general algorithm for matrix multiplication, let n be a power of 2. If let k be nonnegative integer, then n is k-th power of 2, i.e., $n=2^k$. When n>1, we can deeply analyze time complexity of the general algorithm for matrix multiplication through the following recursive expression,

$$\text{T(n)} = 8\text{T}\left(\frac{\text{n}}{2}\right) + 4\left(\frac{n}{2}\right)^2 \tag{10}$$

We come next to the analysis of equation(10). We replace variable n with 2^k.

$$\text{T(n)} = 8T\left(\frac{2^k}{2}\right) + 4\left(\frac{2^k}{2}\right)^2$$
$$= 8T\left(2^{k-1}\right) + 4\left(2^{k-1}\right)^2$$
$$= \cdots = \sum_{i=0}^{k-1} 8^i \times 4\left(2^{\text{k-i-1}}\right)^2$$
$$= 4\left(2^{\text{k}}\right)^2 \sum_{i=0}^{k-1} \frac{8^i}{\left(2^{\text{i+1}}\right)^2}$$
$$= \left(2^{\text{k}}\right)^2 \sum_{i=0}^{k-1} 2^i$$
$$= \left(2^{\text{k}}\right)^2 \times \frac{1-2^k}{1-2}$$
$$= 2^{3k} - 2^{2k}$$
$$= 2^{3\log_2 n} - 2^{2\log_2 n}$$
$$= n^3 - n^2$$

Thus, we have

$$T(n) = \begin{cases} 1 & \text{n} = 1 \\ n^3 - n^2 & \text{n} > 1 \end{cases} \tag{11}$$

Let us begin to consider the behavior of a new function:

$$K(n) = (n^3 - n^2) - (n^{2.81} - 6n^2)$$
$$\text{i.e., } K(n) = n^3 - n^{2.81} + 5n^2 \tag{12}$$

We can easily get the first order derivative and second order derivative of the function, such that

$$K'(n) = 3n^2 - 2.81n^{1.81} + 10n \tag{13}$$
$$K''(n) = 6n - 5.0861n^{0.81} + 10 \tag{14}$$

We can obtain that if n>1, then $K'(n) > 0$ and $K''(n) > 0$. This implies that Strassen algorithm is always better than the general algorithm for matrix multiplication in terms of time complexity. And the advantage of the Strassen algorithm will became more obvious with the increase of dimension of matrix. Strassen algorithm adopts the method of recursive algorithms, which is helpful to the implementation and analysis of the algorithm. However, the method make the algorithm require a great number of dynamic two dimensional arrays so as to assign memory space. which enhance the time complexity and space complexity when the dimension of matrix is smaller. However, the general algorithm for matrix multiplication is no problem on this aspect. Tests show when the dimension of matrix is less than 500, this situation will get even worse. In order to solve this problem, we can specify a numeric value (such as 500) for algorithm routine, and we will adopt Strassen algorithm or the general algorithm for matrix multiplication based on the comparisons between the dimension and the numeric value.

The function of Strassen algorithm is as follows:

```
template<class Type>
void S (int n,Type A[][N],Type B[][N],Type AB[][N])
{
 ...// definition of related array
int i,j;
 if (n==2)
  ...;//the general algorithm for matrix multiplication
 else
 {
   for(i=0;i<n/2;i++)
    for(j=0;j<n/2;j++)
     {
      A11[i][j]=A[i][j];
      A12[i][j]=A[i][j+n/2];
      A21[i][j]=A[i+n/2][j];
      A22[i][j]=A[i+n/2][j+n/2];
      B11[i][j]=B[i][j];
      B12[i][j]=B[i][j+n/2];
      B21[i][j]=B[i+n/2][j];
     B22[i][j]=B[i+n/2][j+n/2];
    }
          //the matrix is partitioned into 4 blocks.
    S (n/2,A11,BB,M1);
   ...;// BB=B12-B22
    S (n/2,AA,B22,M2);
   ...;//AA=A11+A12
```

```
  S (n/2,AA,B11,M3);
   ...;// AA=A21+A22
  S (n/2,A22,BB,M4);
   ...;// BB=B21-B11
  S (n/2,AA,BB,M5);
  ...;//AA=A11+A22
  ...;//BB=B11+B22
  S (n/2,AA,BB,M6);
  ...;//AA=A12-A22
  ...;//BB=B21-B22
  S (n/2,AA,BB,M7);
  ...;//AA=A11-A21
  ...;//BB=B11-B12
  ...;//AB11=M5+M4-M2+M6
  ...;//AB12=M1+M2
  ...;//AB21=M3+M4
  ...;//AB22=M5+M1-M3-M7
 for(i=0;i<n/2;i++)
  for(j=0;j<n/2;j++)
  {
   AB[i][j]=AB11[i][j];
   AB[i][j+n/2]=AB12[i][j];
   AB[i+n/2][j]=AB21[i][j];
   AB[i+n/2][j+n/2]=AB22[i][j];
  }
 //store the result in matrix AB
   }
}
```

subtraction) to decrease the number of block matrix multiplication, and adopts divide and conquer algorithm to complete matrix multiplication through recursive function. Such methods can effectively reduce the time complexity of matrix multiplication. And Within the algorithm program, the computing process is accomplished by the block matrixes(A11,A12,B11,B12)that act as middle variables, and recursive function is accomplished by $S(n/2,A11,BB,M_i)$ $(i=1,2,3,4,5,6,7)$.While we pay particular attention to the obvious advantages of Strassen algorithm, we should notice that Strassen algorithm need to create a large number of dynamic two dimensional arrays, which takes more computing time to assign memory space. And, the higher the order of the matrices, the more memory space Strassen algorithm program will require. Experiments indicate the algorithm of Strassen is not much faster than the general matrix multiplication algorithm when the order of the matrices is relatively small (usually $n<45$). Therefore, it is necessary to improve Strassen algorithm through setting a boundary for the order of the matrices. When the order of the matrices is bigger than the boundary we adopt Strassen algorithm, otherwise we adopt the general matrix multiplication algorithm.

REFERENCES

[1] Aho, A.V., Hopcroft, J.E.,and Ullman, J.D.(2007). The Design and Analysis of Computer Algorithms(Huang, L.P., Wang, D.J., and Zhang, S., Trans.). Beijing, China: China Machine Press.(1974).

[2] Deng, X.Y.,and Wan, T.T.(2006). Design and Analysis of Algorithms. Beijing, China: Metallurgical Industry Press.

[3] Wang, X.D.(2001). The Design and Analysis of Computer Algorithms. Beijing, China: Publishing House of Electronics Industry.

[4] Liu, J.(2003). An Introduction to Computer Algorithm—Techniques of Design and Analysis. Beijing, China:Science Press.

[5] Su, D.F.,and Zhong, C.(2001).The Design and Analysis of Computer Algorithms. Beijing, China: Publishing House of Electronics Industry.

[6] E.Horowitz.,and S.Sahni.(1978). Fundamentals of Computer Algorithms. U.S.A.: Compter Science Press.

[7] E.Horowitz., S.Sahni.,and S.Rajasekaran.(1996). Computer Algorithms/C++. U.S.A.: Compter Science Press.

[8] T.H.Cormen., C.Leiserson.,and C.Stein.(2001). Introduction to Algorithms. U.S.A:The MIT Press.

[9] Alfred V.Aho.,John E. Hopcroft., and Jeffrey D.Ullman.(1983). Data Structures and and Algorithms. U.S.A: Addison-Wesley.

[10] Sara Baase., and A.Gelder.(2000). Computer Algorithms:Introduction to Design and Analysis. U.S.A: Addison-Wesley.