Towards an alternate approach in implementation of Strassen's Matrix Multiplication

Dr.N Guruprasad, C N Ravi, G Pradeep
Associate Professor, Information Science and Engineering Department, New Horizon College of Engineering, Bangalore, India, guruprasadn@newhorizonindia.edu
Kavitha K K, Beena B M
Assistant Professor Information Science and Engineering Department, New Horizon College of Engineering, Bangalore, India, kkkavitha@newhorizonindia.edu

Abstract:
Ever since the dawn of the computer age, researchers have been trying to find an optimal way of multiplying matrices,afundamental operation that is a bottleneck for many important algorithms Faster matrix multiplication would give more efficient algorithms for many standard linear algebra problems, such as inverting matrices, solving of MM because Strassen's reduces the total number of operations. The Strassen's method of matrix multiplication is a typical divide and conquer algorithm. Strassen achieved this operation reduction by replacing computationally expensive systems of linear equations, and finding determinants. Even some basic graph algorithms run only as fast as matrix multiplication. Strassen's matrix multiplication (MM) has benefits with respect to any (highly tuned) implementations MMs with matrix additions (MAs). Strassen's method is not the asymptotically faster known matrix multiplication algorithm, but it is most widely used for large matrices. In this paper we describe a new variant of Strassen's where in additive complexity is reduced to 15 operations as compared to 18.
Keywords: additive complexity, efficiency, matrix addition, matrix multiplication, Strassen's algorithm.

I. Introduction:

VOLKER STRASSEN is a German mathematician born in 1936. He is well known for his works on probability, but in the computer science and algorithms he's mostly recognized because of his algorithm for matrix multiplication that's still one of the main methods that outperforms the general matrix multiplication algorithm. Strassen firstly published this algorithm in 1969 and proved that the n^3 algorithm isn't the optimal one [1]. Actually the given solution by Strassen is slightly better, but his contribution is enormous because this resulted in many more researches about matrix multiplication that led to some faster approaches.

The standard method for multiplying n*n matrices requires $O(n^3)$ multiplication. The road to current best upper bound was built on clever ideas and increasingly complex combinatorial techniques. Many researchers long believed that the standard $O(n^3)$ algorithm was the best possible and it was even proved that in certain modes of computation no algorithm could do better. It was therefore a source of great excitement when towards end of 1960's Strassen caused a considerable stir by improving this algorithm. He stunned the entire research world with his $O(n^{2.81})$ algorithm for multiplying matrices [2]. From an algorithmic point of view, this break through is a landmark in the history of divide and conquer – which is considered as a top design technique for designing algorithms.

II. Strassen's approach:

Still sometimes in practice, Strassen's algorithm is reminiscent of a shortcut, first observed by Gauss for multiplying complex numbers. Though finding the product (a+bi)(c+di) = ac – bd + i(bc+ad) seems to require four multiplication, Gauss observed that it can actually be done with three multiplication – ac, bd and (a+b)(c+d) – because bc + ad = (a+b)(c+d)-ac-bd. Strassen's algorithm reduces the number of multiplications by increasing the efficiency from eight to seven by expressing the entries of AB (A and B are 2 * 2 matrices) as linear combinations. Given that large matrices can be added mush faster than they can be multiplied, saving one multiplication more than compensates for the supplementary additions [5]. This trick applied to 2 * 2 matrix reduces the problem to seven multiplications, making the recursive application giving an algorithm that runs in $O(n^{\log_2 7}) = O(n^{2.81})$

A. Example based on Strassen:

A conventional way of multiplying two matrixes requires eight multiplications. Given two matrices A and B, the eight multiplications performed are depicted below:

C1 = A1*B1 + A2*B3
C2 = A1*B2 + A2*B4
C3 = A3*B1 + A4*B3
C4 = A3*B2 + A4*B4

However Strassen used a strategy to show how to use only seven recursive calls by carefully arranging the computation.

$$\begin{vmatrix} C1 & C2 \\ C3 & C4 \end{vmatrix} = \begin{vmatrix} E+I+J-G & D+G \\ E+F & D+H+J-F \end{vmatrix}$$

where:

D = A1 (B2 - B4)
E = A4 (B3 - B1)
F = (A3 + A4) B1
G = (A1 + A2)B4
H = (A3 - A1) (B1 + B2)

I = (A2 - A4) (B3 + B4)
J = (A1 + A2) (B1 + B4)

Considering the following matrix :

$$\begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} * \begin{vmatrix} 5 & 6 \\ 7 & 8 \end{vmatrix}$$

By putting in the seven smaller products we get D=-2, E=8, F=35, G=24, H=22, I=-30 and J=65

Final result is:

$$\begin{vmatrix} 19 & 22 \\ 43 & 50 \end{vmatrix}$$

This requires 18 additions – ten in computing the seven small products and eight in the final computation.

III. Described sequence

A new variant is proposed requiring minimum number of operations.

We compute the following :
S1= a22+a12; S2=a22-a21; S3=s2+a12; S4=s3-a11
T1= b22+b12; T2=b22-b21; T3=t2+b12; T4=t3-b11

Computing the seven small products and simultaneously applying to the above said problem, we get:
D = s1t1 = 84; E = s2t2 = 1; F = x3t3 = 21; G = a11b11 = 5 ; H = a12b21 = 14 ; I = s4b12 = 12; J= a21t4 = 6

Now to reduce the additive complexity, we calculate something called as post calculations.They are :
PC1 = F + H = 35; PC2 = D – PC1 = 49; PC3 = PC1 – E = 34

Now these computations are carefully arranged to get the final result.

$$\begin{vmatrix} C1 & C2 \\ C3 & C4 \end{vmatrix} = \begin{vmatrix} G + H & PC3 - I \\ PC2 - J & E + PC2 \end{vmatrix}$$

Final result is :

$$\begin{vmatrix} 19 & 22 \\ 43 & 50 \end{vmatrix}$$

This requires 15 additions – eight in initial computation of S and T, three in post calculation while other four in the final computation.

IV. Analysis of Strassens algorithm:

If T(N) or M(N) is number of multiplications made by the algorithm in multiplying two n*n matrices (where n is power of 2) we have following recurrence relation :

$$\begin{aligned} T(N) &= 1 && \text{if } n = 1 \\ &= 7T(N/2) && \text{otherwise} \end{aligned}$$

Considering $T(n) = 7T(n/2)$

Replacing n by n/2 we get :

$T(n/2) = 7T(n/2^2)$

$T(n) = 7 . 7T(n/2^2) = 7^2T(n/2^2)$

In general = $7^iT(n/2^i)$

Now to get terminal condition $T(1) = 1$

Let $n=2^i \rightarrow A$ and $T(n) = 7^iT(n/n) = 7^i \rightarrow B$

Taking log of the equation we get $i = \log_2 n$

Substituting we get : $T(n) = 7^{\log_2 n} = n^{\log_2 7} = n^{2.807}$

Hence complexity is approximately $O(n^{2.81})$

V. Post Strassens's work:

Following Strassen's discovery, a number of researchers attempted to improve this constant [4]. The obvious thing to try first was to multiply two 2*2 matrices with six scalar multiplications. But in 1972 HOPCROFT and KERRproved this is impossible when commutativity of multiplication cannot be used. The next thing to try was to find a way to multiply 3*3 matrices with at most 21 scalar multiplication. This would result in a time O(N $\log_3 21$) – faster than Strassen's algorithm since $\log_3 21 < \log_2 7$ Unfortunately this too is impossible. Almost a decade passed before PAN discovered a way to multiply two 70 * 70 matrices with 143640 scalar multiplications – compare this with 343000 required by classic algorithm – and indeed $\log_7 143640$ – is a tiny bit, smaller than log 7 [5]. Numerous algorithms were discovered subsequently. For instance it was known at the end of 1979, that matrices could be multiplied in a time of $O(N^{2.521813})$; imagine the excitement in Jan 1980 when this was improved to $O(N^{2.521801})$ [6]. Way back in 1986 – COPPERSMITH and WINOGRAD discovered that it is possible at least in theory, to multiply two N * N matrices in a time $O(N^{2.376}$) Because of hidden constants involved, however none of the algorithm found after Strassen's is of much practical use [7].

VI. Conclusion:

As usual there are details to consider, such as case when N is not a power of 2, but these are basically minor nuisances. Strassen's algorithm is worse than the straight forward algorithm until N is fairly large. It does not generalize for the case where matrices are sparse (contain many zero entries). When run with floating point entries it is less stable numerically than the classic algorithm. Thus it has only limited applicability. Nevertheless it represents an important theoretical milestone and certainly show that in computer science as in many other fields even though a problem seems to have an intrinsic complexity, nothing is certain until proven.

References:

[1]. E. Anderson , Z. Bai , C. Bischof , J. Demmel , J. Dongarra , J. Du Croz , A. Greenbaum , S. Hammarling , A. McKenney , S. Ostrouchov , D. Sorensen, LAPACK's user's guide, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992

[2]. David H. Bailey , King Lee , Horst D. Simon, Using Strassen's algorithm to accelerate the solution of linear systems, The Journal of Supercomputing, v.4 n.4, p.357-371, 1990 [doi>10.1007/BF00129836]

[3]. Richard P. Brent, Algorithms for matrix multiplication, Stanford University, Stanford, CA, 1970

[4]. .Cohen and M. Roth. On the implementation of Strassen's fast multiplication algorithm. ActaInformatica, 6:341-355, 1976.

[5]. Thomas T. Cormen , Charles E. Leiserson , Ronald L. Rivest, Introduction to algorithms, MIT Press, Cambridge, MA, 1990

[6]. S. Huss-Lederman, E. M. Jacobson, J. R. Johnson, A. Tsao, and T. Turnbull.Strassen's algorithm for matrix multiplication: Modeling, analysis, and implementation. Technical report, Center for Computing Sciences, 1996.Technical Report CCS-TR-96-147.

[7]. V. Strassen. Gaussian elimination is not optimal. Numer. Math., 13:354-356, 1969.