A Variation of Strassen's Matrix Multiplication Algorithms

Sarah M. Loos

Undergraduate, Computer Science, Indiana University, smloos@indiana.edu

ABSTRACT

A very simple recasting of this classic 7-multiplication recursion improves its time performance for rectangular matrices of order n when n is *not* a power of two. No time is lost in the rare cases when $n = 2^p$. A similar relabeling applies, as well, to Winograd's 15-addition variant.

SUB-CUBIC MULTIPLICATION

Strassen's $O(n^{2.807})$ matrix-multiplication algorithm was published forty years ago, stunning algorithmists with a simple recurrence that improved the best asymptotic time for a very important and heretofore "obviously cubic" problem. Since then there have been debates about its practicality, based on numeric stability, extra space, and locality. Among those issues has been how to deal with the n between powers of two and with rectangular matrices.

We cleave a matrix of order n by partitioning the n indices into $2^{\lceil \lg n \rceil - 1}$ and $n - 2^{\lceil \lg n \rceil - 1}$. In the simplest cases — square matrices of order n — it results in a northwest quadrant that is square with order a power of two, and southwest, northeast, and southeast matrices fitted around it. This is close to static padding with zeroes, except that bounds checking identifies entire blocks of zeroes.

Recursions on such a northwest quadrant, therefore, will surely generate a lot of work, whereas recursions on the southeast might generate very little. In the standard algorithm below, it is clear that the big northwest quadrants are used frequently relative to the others. With this cleaving, any sum with one of them as an addend, or any product of two northwest factors, is likely to require a full dose of scalar operations, and so they are identified below in red.

$$
\begin{aligned}
\mathbf{M_1} &\coloneqq (\mathbf{A_{nw}} + A_{se})(\mathbf{B_{nw}} + B_{se}),\\
M_2 &\coloneqq (A_{sw} + A_{se})\mathbf{B_{nw}},\\
M_3 &\coloneqq \mathbf{A_{nw}}(B_{ne} - B_{se}),\\
M_4 &\coloneqq A_{se}(B_{sw} - \mathbf{B_{nw}}),\\
M_5 &\coloneqq (\mathbf{A_{nw}} + A_{ne})B_{se},\\
\mathbf{M_6} &\coloneqq (A_{sw} - \mathbf{A_{nw}})(\mathbf{B_{nw}} + B_{ne}),\\
M_7 &\coloneqq (A_{ne} - A_{se})(B_{sw} + B_{se});
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{C_{nw}} &\coloneqq \mathbf{M_1} + M_4 - M_5 + M_7,\\
C_{ne} &\coloneqq M_3 + M_5,\\
C_{sw} &\coloneqq M_2 + M_4,\\
C_{se} &\coloneqq \mathbf{M_1} - M_2 + M_3 + \mathbf{M_6}.
\end{aligned}
$$

THE SIMPLE VARIANT

A simple relabeling of the quadrants generates an equivalent algorithm without a northwestern tilt.

Let Z be the zero matrix; it is the additive identity and the multiplicative annihilator: $A + Z = A$; $AZ = Z = ZA$, assuming compatible factors. These algorithms treat any quadrant that is out-of-bounds as Z, and so this algebra prunes away many scalar operations.

Let I be the identity matrix, and consider the simple permutation $P = \begin{pmatrix} Z & I \\ I & Z \end{pmatrix}$,

that flips columns or rows, depending on whether it is multiplied on the right or left of a matrix. Then

$$AB = \begin{pmatrix} A_{nw} & A_{ne} \\ A_{sw} & A_{se} \end{pmatrix}\begin{pmatrix} B_{nw} & B_{ne} \\ B_{sw} & B_{se} \end{pmatrix} = APP^TB = \begin{pmatrix} A_{ne} & A_{nw} \\ A_{se} & A_{sw} \end{pmatrix}\begin{pmatrix} B_{sw} & B_{se} \\ B_{nw} & B_{ne} \end{pmatrix}.$$

When the quadrants in Strassen's Algorithm are renamed according to the right most product, almost the same algorithm results. Although they take different intermediate values, the sums for the final quadrants of C remain the same as above.

$$
\begin{aligned}
M_1 &\coloneqq (A_{ne} + A_{sw})(B_{sw} + B_{ne}),\\
M_2 &\coloneqq (A_{se} + A_{sw})B_{sw},\\
M_3 &\coloneqq A_{ne}(B_{se} - B_{ne}),\\
M_4 &\coloneqq A_{sw}(\mathbf{B_{nw}} - B_{sw}),\\
M_5 &\coloneqq (A_{ne} + \mathbf{A_{nw}})B_{ne},\\
M_6 &\coloneqq (A_{se} - A_{ne})(B_{sw} + B_{se}),\\
\mathbf{M_7} &\coloneqq (\mathbf{A_{nw}} - A_{sw})(\mathbf{B_{nw}} + B_{ne});
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{C_{nw}} &\coloneqq M_1 + M_4 - M_5 + \mathbf{M_7},\\
C_{ne} &\coloneqq M_3 + M_5,\\
C_{sw} &\coloneqq M_2 + M_4,\\
C_{se} &\coloneqq M_1 - M_2 + M_3 + M_6.
\end{aligned}
$$

CONCLUSION

Because we expect only the northwest quadrants to be full, we re-label the quadrants of Strassen's algorithm, as well as Winograd's variant, to favor work on the others. The other three quadrants, to the south and east, can be far smaller – even empty – and so we use them more. As soon as they go out-of-bounds in recurrence, they are treated as zeroes and many operations are saved.

In spite of our simple and effective improvements to the $O(n^{\lg 7})$ algorithm, the race is still won by the classic $O(n^3)$ algorithm, which makes better use of memory bandwidth.

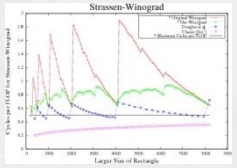
Our modified algorithms perform no worse than the classic ones in all cases and their performance is much improved on rectangular matrices as size grows beyond any power of two.

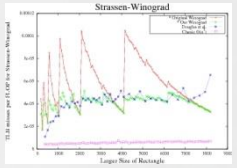
EXPERIMENTAL RESULTS

In order to demonstrate our alternative version, we implemented the 15-addition Strassen-Winograd version using C++ and MTL4, which provides recursators as a generic-programming alternative to iterators. Tests were run on a uniprocessor AMD Opteron, clocked at 2.0 Ghz, using Gnu g++ -O3, and Release 6595 of MTL4.

Similarly we implemented the original Strassen-Winograd algorithm and the classic $O(n^3)$ algorithm (using MTL4 recursators). All three algorithms run to $32 \times 32 \times 32$ base cases that are implemented in tuned assembly-language. As a result, all tests are on sizes that are even multiples of 32. We also compared to the implementation of Strassen-Winograd by Douglas *et al.* available from NetLib.

The following plots present the results for rectangular matrix multiplication of size $n \times 2n \times n$ where n is plotted on the horizontal axis. All plots are normalized by dividing by $5.2n^{\lg 7}$. All sizes between 96×192 up to 4096×8192 in steps of 32 or 64 were tested. Notably only our algorithm and the classic $O(n^3)$ algorithm exhibit smooth performance. The latter plot rises because its n^3 time is normalized only by $n^{\lg 7}$, but it still remains fastest.





PRINCIPAL REFERENCES

[1] C. C. Douglas, M. Heroux, G. Slishman, and R. M. Smith. GEMMW: a portable level-3 BLAS Winograd variant of Strassen's matrix-matrix multiply algorithm. J. Comput. Phys., 110(1):1-10, 1994. http://dx.doi.org/10.1006/jcph.1994.1001

[2] P. Gottschling, D. S. Wise, and M. D. Adams. Representation transparent matrix algorithms with scalable performance. In Proc. 21st ACM Int. Conf. on Supercomputing, pages 116-125. ACM Press, New York, June 2007. http://doi.acm.org/10.1145/1274971.1274989

[3] D.S. Johnson. A theoretician's guide to the experimental analysis of algorithms. In M. H. Goldwasser, D. S. Johnson, and C. C. McGeoch, editors, Data Structures, Near Neighbor Searches, and Methodology: 5th & 6th DIMACS Implementation Challenges, Volume 59 of DIMACS Ser. Discrete Math. Theoret. Comput. Sci., pages 215-250. Amer. Math. Soc., Providence, 2002. http://www.research.att.com/~dsj/papers.html

[4] V. Strassen. Gaussian elimination is not optimal. Numer. Math., 13(4):354-356, Aug. 1969. http://dx.doi.org/10.1007/BF02165411

Supported, in part, by NSF under an REU and grants numbered CCF-0736560, CCF-0541364 and EIA-0202048.
October 1st, 2008