Accelerating Neural Network Training using Arbitrary Precision Approximating Matrix Multiplication Algorithms

Author: Grey Ballard, Jack Weissenberger, Luoping Zhang
08/09/21 Wake Forest University

Summary of Contributions

- Curate practical, approximate, fast matrix multiplication algorithms
- Generate efficient multithreaded code for all of them, with a 25% improvement over best classical implementation (BLAS GEMM)
- Demonstrate robustness of Neural Network training with approximate matrix multiplication
- Accelerate multithreaded training performance of Multi-Layer Perceptron and VGG-19 networks up to 15%

Neural Network

Application:

Face Recognition

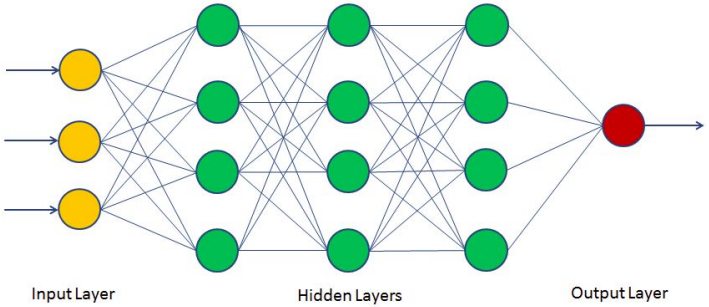
Smart Assistant

Related Company: Amazon, Google, etc

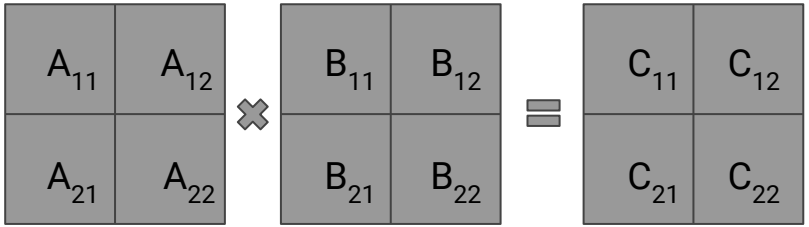
Theoretical Basis:

Weight Optimization

Matrix Operation: Matrix Multiplication



Matrix Multiplication



A B C

$C_{11} = A_{11}*B_{11} + A_{12}*B_{21}$

$C_{21} = A_{21}*B_{11} + A_{22}*B_{21}$

$C_{12} = A_{11}*B_{12} + A_{12}*B_{22}$

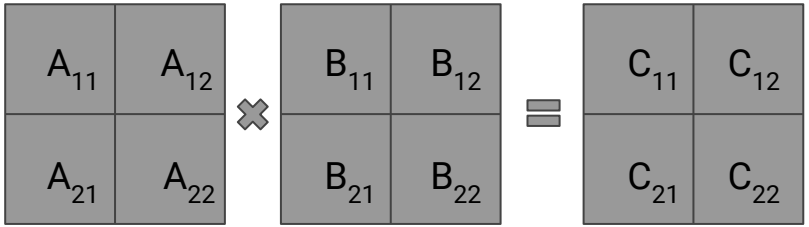
$C_{22} = A_{21}*B_{12} + A_{22}*B_{22}$

Classic Algorithm:

8 multiplications

Complexity: $O(n^3)$

Strassen's Algorithm



A B C

7 multiplications
less subproblems per iteration

$\log_2 7 \approx 2.81$

Complexity: $O(n^{2.81})$

$C_{11} = (A_{11}+A_{22})(B_{11}+B_{22})+A_{22}(B_{21}-B_{11})-(A_{11}+A_{12})B_{22}+(A_{12}-A_{22})(B_{21}+B_{22})$

$C_{12} = A_{11}(B_{12}-B_{22})+(A_{11}+A_{12})B_{22}$

$C_{21} = (A_{21}+A_{22})B_{11}+A_{22}(B_{21}-B_{11})$

$C_{22} = (A_{11}+A_{22})(B_{11}+B_{22})-(A_{21}+A_{22})B_{11}+A_{11}(B_{12}-B_{22})+(A_{21}-A_{11})(B_{11}+B_{12})$

Matrix Dimension

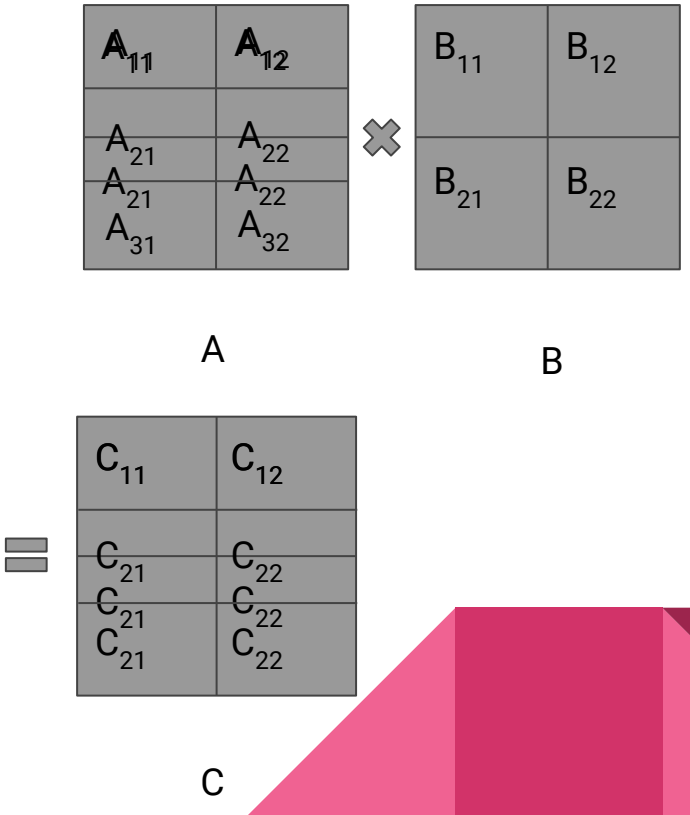
Strassen: $\log_{2x2x2} 7^3 \approx 2.81$

Bini: $\log_{3x2x2} 10^3 = \log_{12} 1000 \approx 2.78$

Algorithm shape/dimension:

Strassen <2,2,2;7>

Bini <3,2,2;10>



APA -- Arbitrary Precision Approximation

Bini's 322 Algorithm <3,2,2>:

$$C_{11} = ((A_{11}+A_{22})\cdot(\varepsilon B_{11}+B_{22})+A_{22}\cdot(-B_{21}-B_{22})-A_{11}\cdot B_{22}+(\varepsilon A_{12}+A_{22})\cdot(-\varepsilon B_{11}+B_{21}))/\varepsilon$$

$$C_{12} = (-A_{11}\cdot B_{22}+(A_{11}+\varepsilon A_{12})\cdot(\varepsilon B_{12}+B_{22}))/\varepsilon$$

$$C_{21} = (\varepsilon A_{12}+A_{22})\cdot(-\varepsilon B_{11}+B_{21})+(A_{21}+A_{32})\cdot(B_{11}+\varepsilon B_{22})-(\varepsilon A_{31}+A_{32})\cdot(B_{12}-\varepsilon B_{22})$$

$$C_{22} = (A_{11}+A_{22})\cdot(\varepsilon B_{11}+B_{22})-(A_{11}+\varepsilon A_{12})\cdot(\varepsilon B_{12}+B_{22})+(A_{21}+\varepsilon A_{31})\cdot(B_{12}-\varepsilon B_{22})$$

$$C_{31} = (-A_{32}\cdot B_{11}+(\varepsilon A_{31}+A_{32})\cdot(B_{12}-\varepsilon B_{22}))/\varepsilon$$

$$C_{32} = ((A_{21}+A_{32})\cdot(B_{11}+\varepsilon B_{22})+A_{21}\cdot(-B_{11}-B_{12})-A_{32}\cdot B_{11}+(A_{21}+\varepsilon A_{31})\cdot(B_{12}-\varepsilon B_{22}))/\varepsilon$$

10 multiplication on (3, 2) x (2, 2) matrices $\longrightarrow O(n^{2.78})$

ε: approximation parameter

Roundoff Error
Approximation Error

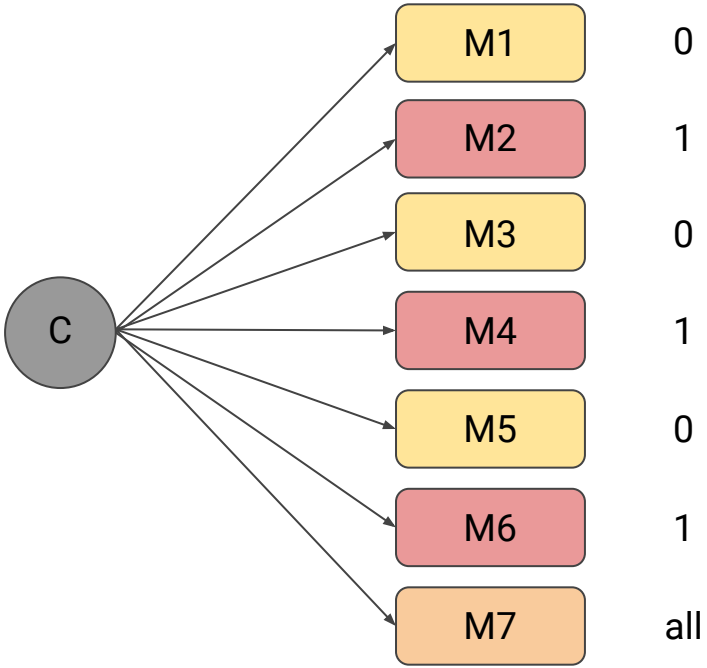
12 APA algorithms

Ref	Dimension	Relative Error	Complexity	Speedup
[1]	<3,2,2>	1e-3	$n^{2.78}$	20%
[2]	<3,3,3;21>	1e-2	$n^{2.771}$	28.6%
[3]	<4,2,2>	1e-2	$n^{2.775}$	23.1%
[3]	<5,2,2>	1e-2	$n^{2.777}$	25%
[3]	<3,3,2>	1e-2	$n^{2.739}$	28.6%
[4]	<4,3,3>	1e-2	$n^{2.759}$	33.3%
[5]	<4,4,2>	1e-2	$n^{2.751}$	33.3%
[6]	<4,4,4>	1e-2	$n^{2.762}$	39.1%
[7]	<5,5,2>	1e-2	$n^{2.769}$	35.1%
[6]	<5,5,5>	1e-2	$n^{2.796}$	38.9%
[8]	<7,2,2>	1e-1	$n^{2.783}$	27.3%
[9]	<3,3,3;20>	1e-1	$n^{2.727}$	35%

Single precision

Neural Networks' Robustness

Parallel Implementation

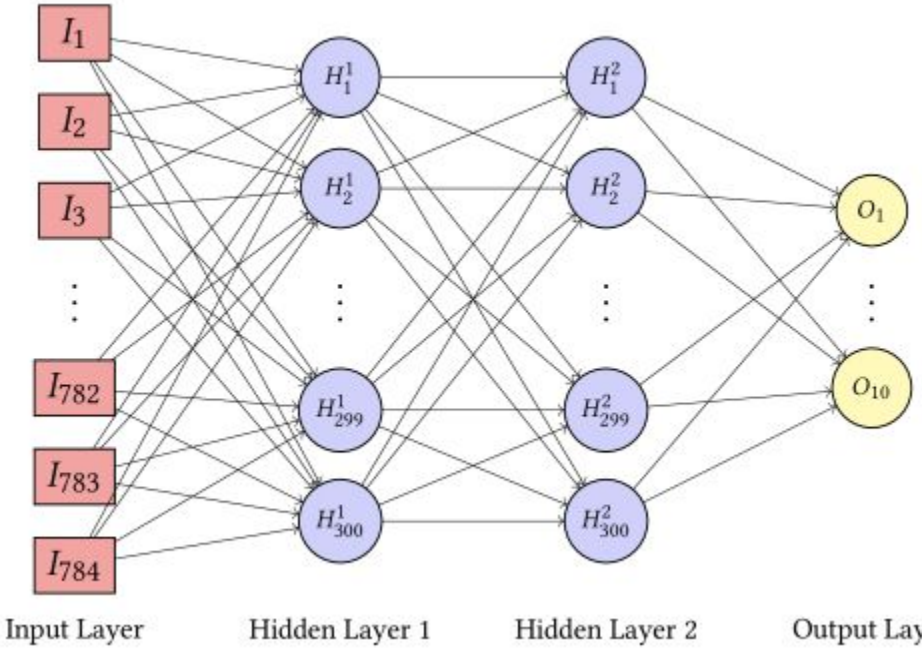


Hybrid approach: BFS + DFS

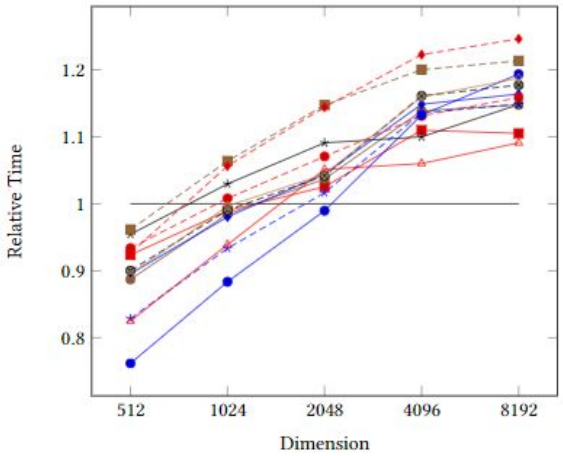
Strassen’s algorithm implementation on 2 threads

Code generation for each algorithm

Multi-Layer Perceptron

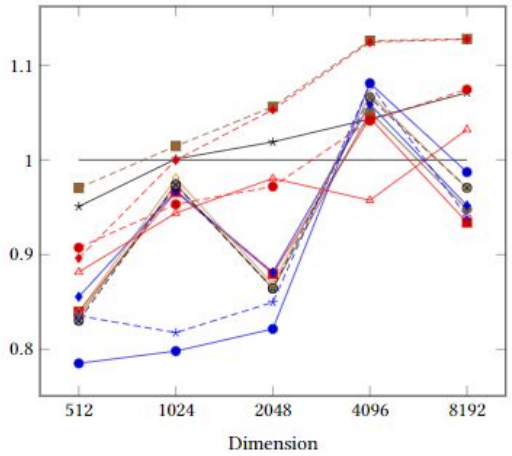


Parallel Performance on Neural Network Training



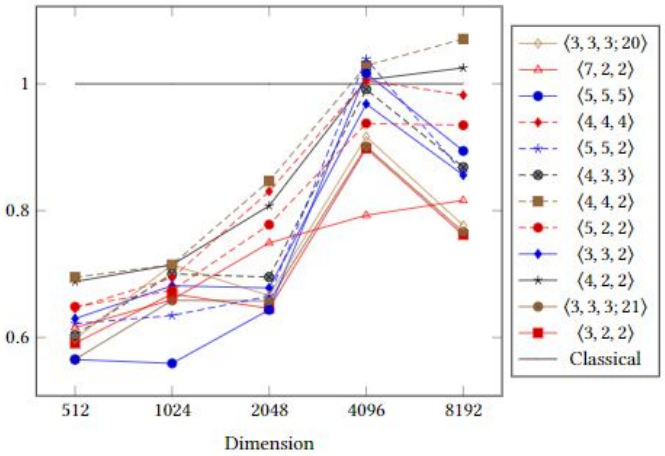
(a) One thread

8%~25%



(b) Six threads

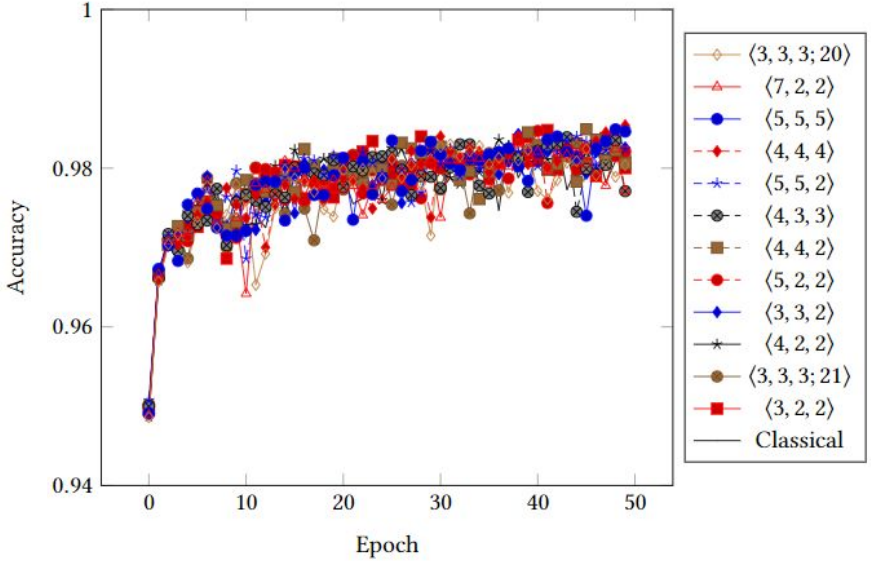
13%: <4,4,2>&<4,4,4>



(c) Twelve threads

7%: <4,4,2>

APA Algorithms Preserve Accuracy



97% ~99% Test accuracy

VGG-19 Model

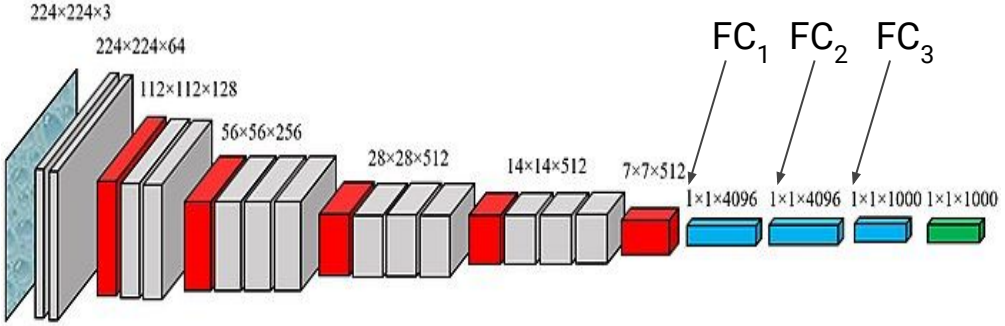
3 Fully-connected Layers:

FC_1 *25088* x *Batch Size* x *4096*
Batch Size x *25088* x *4096*
Batch Size x *4096* x *25088*

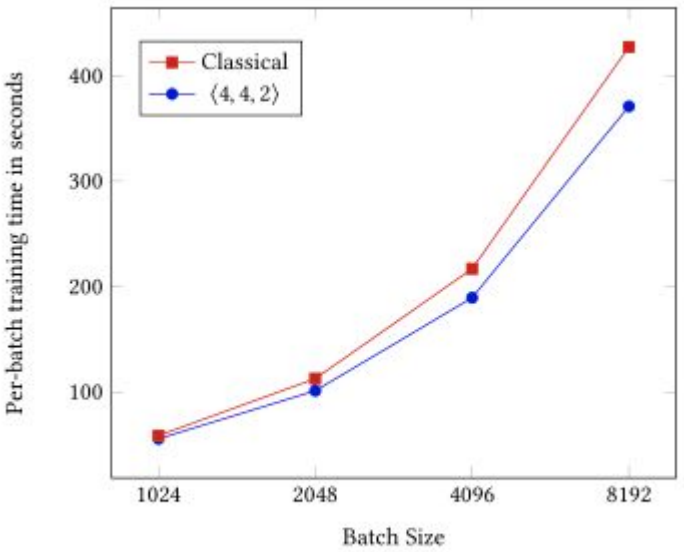
FC_2 *4096* x *Batch Size* x *4096*
Batch Size x *4096* x *4096*
Batch Size x *4096* x *4096*

FC_3 *4096* x *Batch Size* x *1000*
Batch Size x *4096* x *1000*
Batch Size x *1000* x *4096*

Batch Size ={1024, 2048, 3072, 4096}

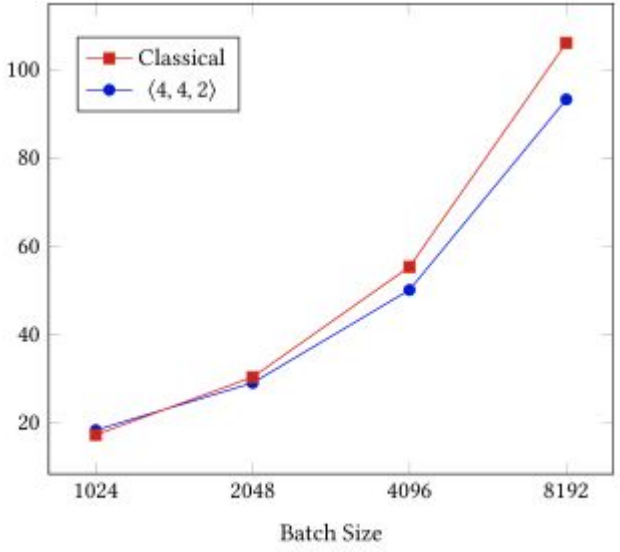


VGG-19 Model



(a) One thread

15% Speedup



(b) Six threads

10% Speedup

Summary of Contributions

- Curate practical, approximate, fast matrix multiplication algorithms
- Generate efficient multithreaded code for all of them, with a 25% improvement over best classical implementation (BLAS GEMM)
- Demonstrate robustness of Neural Network training with approximate matrix multiplication
- Accelerate multithreaded training performance of Multi-Layer Perceptron and VGG-19 networks up to 15%
- Emails: ballard@wfu.edu, jack.weissenberger@gmail.com, robin.zhang@alumni.wfu.edu

Reference

1. D. Bini, M. Capovani, F. Romani, and G. Lotti. 1979.$O(n^{2.7799})$complexity for $n \times n$ approximate matrix multiplication.Information Processing Letters
2. V.B. Alekseev and A.V. Smirnov. 2013. On the exact and approximate bilinear complexities of multiplication of 4x2 and 2x2 matrices. Proceedings of the Steklov Institute of Mathematics
3. A.V. Smirnov. 2013. The bilinear complexity and practical algorithms for matrix multiplication. Computational Mathematics and Mathematical Physics
4. Arnold Schönhage. 1981. Partial and total matrix multiplication. SIAM J. Comput.
5. A.V. Smirnov. 2015. A bilinear algorithm of length 22 for approximate multiplication of 2 x 7 and 7 x 2 matrices. Computational Mathematics and Mathematical Physics
6. A.V. Smirnov. 2016. On the Approximate Bilinear Algorithms for Multiplication of N x N and N x 2 Matrices. Technical Report. ResearchGate
7. A.V. Smirnov. 2016. An Approximate Bilinear Algorithm of Length 27 for Multiplication of 3 x 3 and 3 x 4 Matrices. Technical Report. ResearchGate.
8. A.V. Smirnov. 2014. The Approximate Bilinear Algorithm of Length 46 for Multiplication of 4 x 4 Matrices. Technical Report 1412.1687. arXiv.
9. A.V. Smirnov. 2018. An Approximate Bilinear Algorithm for Multiplying 5 x 5 Matrices of Length. Technical Report. ResearchGate

Algorithm Numerical Accuracy

Algorithm	σ	φ	Relative error
Classic	/	/	1e-7
<3,2,2>	1	1	1e-3
<3,3,3;21>	1	2	1e-2
<4,2,2>	1	2	1e-2
<5,2,2>	1	3	1e-2
<3,3,2>	1	3	1e-2
<4,3,3>	1	3	1e-2
<4,4,2>	1	3	1e-2
<4,4,4>	1	3	1e-2
<5,5,2>	1	3	1e-2
<5,5,5>	1	3	1e-2
<7,2,2>	1	5	1e-1
<3,3,3;20>	1	6	1e-1

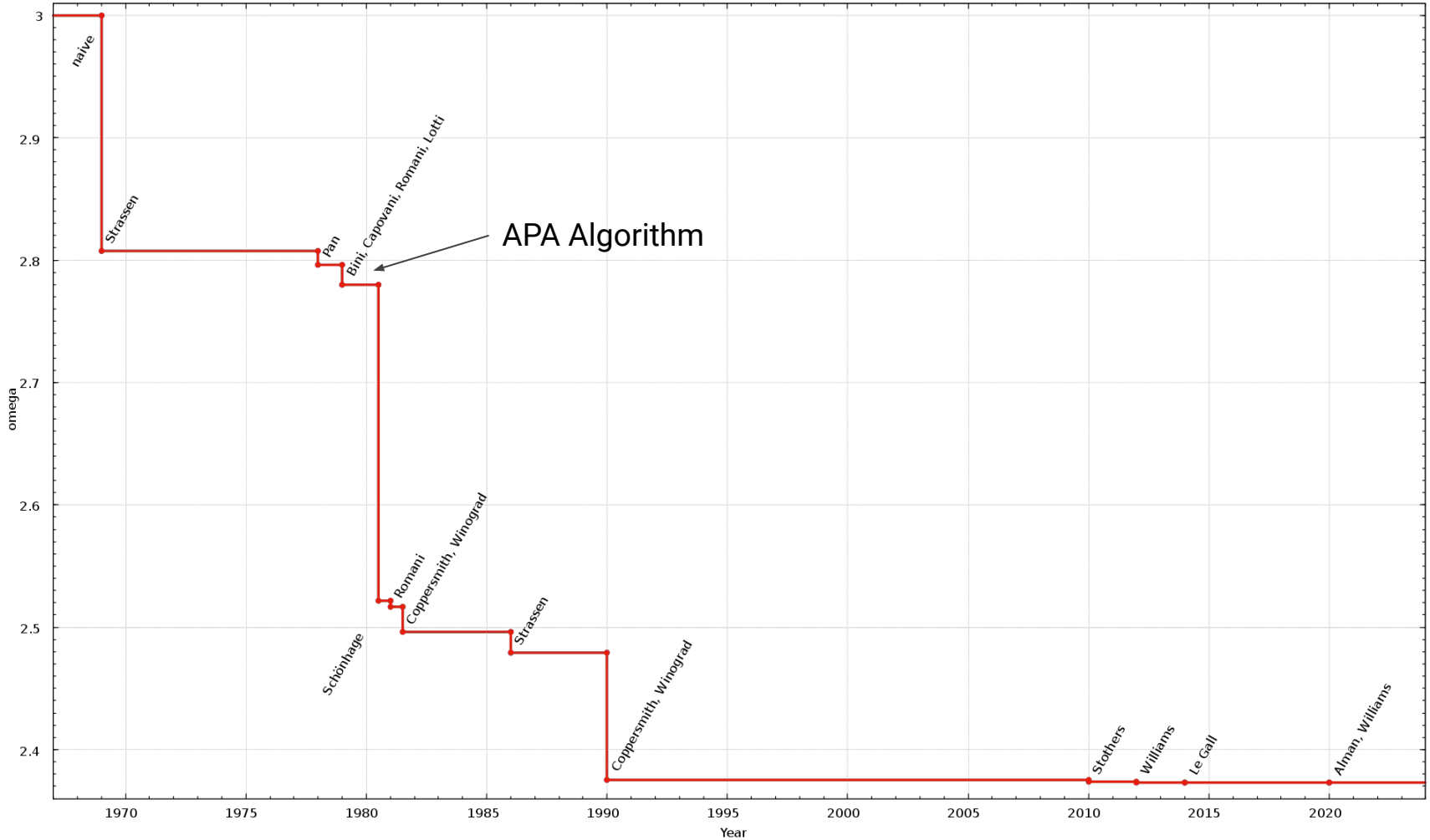
σ: Approximation error

φ: Roundoff error

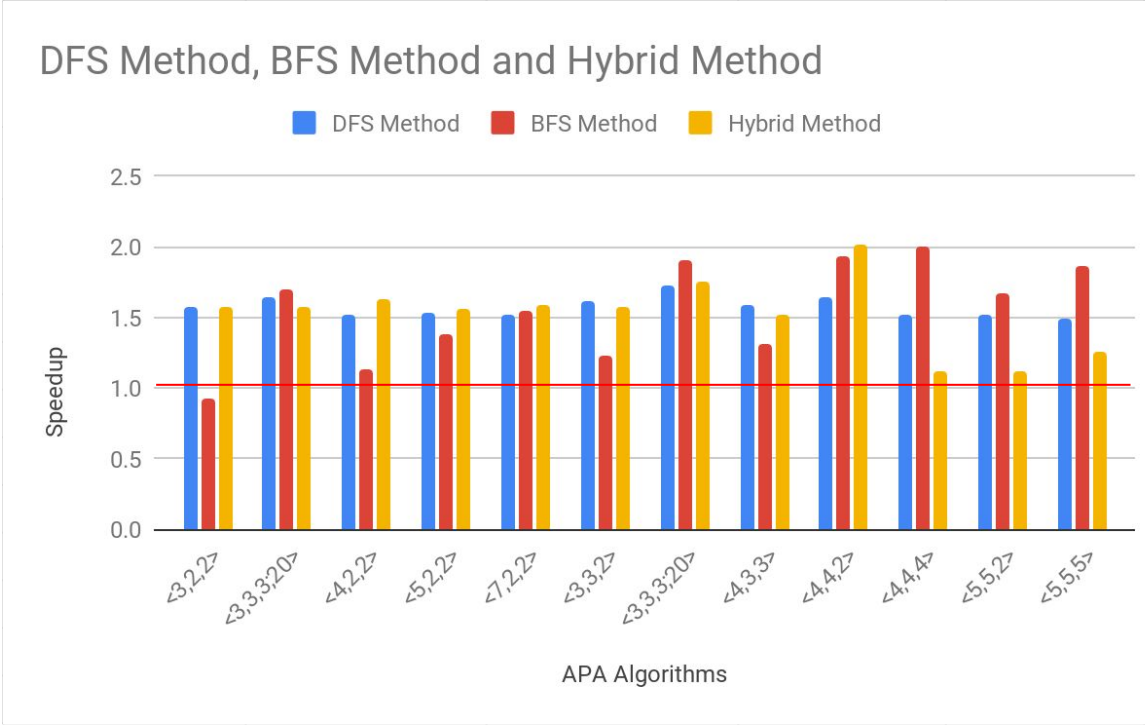
ε: Approximation Parameter

Bini's Theorem: The error produced by an APA-algorithm is $e = O(2^{-d\sigma/(\sigma+\varphi)})$, which is minimized when $\varepsilon = O(2^{-d/(\sigma+\varphi)})$.

2^{-d}: floating point precision


naïve
Strassen
Pan
Bini, Capovani, Romani, Lotti
APA Algorithm
Schönhage
Romani
Coppersmith, Winograd
Strassen
Coppersmith, Winograd
Stothers
Williams
Le Gall
Alman, Williams
omega
Year
3
2.9
2.8
2.7
2.6
2.5
2.4
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020

CPU Parallel Implementation



12-core CPU

Matrix Multiplication of size 6000x6000

12 APA Algorithms