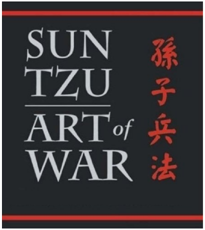

THE UNIVERSITY OF
AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

# Algorithms and Data Structures

## Lecture 8 Matrix multiplication and Strassen's algorithm

Jiamou Liu
The University of Auckland




故用兵之法，十則圍之，五則攻之，倍則分之， 敵則能戰之，少則能守之，不若則能避之。

*``It is the rule in war, if ten times the enemy's strength, surround them; if five times, attack them; if double, be able to divide them; if equal,engage them; if fewer, be able to evade them; if weaker, be able to avoid them.''*

---*``Chapter III Strategic Attack" 500BC*

# Matrix Multiplications

**Matrix Multiplication Problem**

INPUT: Two $n \times n$ integer matrices $A, B$
OUTPUT: Their product matrix $A \times B$.

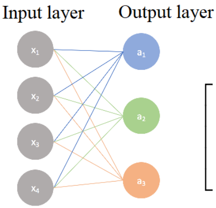
**Exercise.** Compute $A \times B$ by hand, where

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & 4 \\ 5 & 2 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 1 & 0 \\ 3 & 0 & -2 \\ 1 & 4 & 3 \end{bmatrix}$$

Why matrix multiplication?

- Linear programming and optimisation
- Neural networks and machine learning
- Supercomputing
- etc.

**Example.** A multi-layer perceptron (neural network):




$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix} \xrightarrow[\textit{activation}]{} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

# Standard Matrix Multiplications Algorithm

**Standard Matrix Multiplication Algorithm**

The $(i, j)$-entry of $A \times B$ is $\sum_{k=1}^{n} A[i,k]B[k,j]$, i.e.,

$$\begin{pmatrix} a_1 & b_1 & c_1 \\ d_1 & e_1 & f_1 \\ g_1 & h_1 & i_1 \end{pmatrix} \times \begin{pmatrix} a_2 & b_2 & c_2 \\ d_2 & e_2 & f_2 \\ g_2 & h_2 & i_2 \end{pmatrix} =$$

$$\begin{pmatrix} a_1a_2 + b_1d_2 + c_1g_2 & a_1b_2 + b_1e_2 + c_1h_2 & a_1c_2 + b_1f_2 + c_1i_2 \\ d_1a_2 + e_1d_2 + f_1g_2 & d_1b_2 + e_1e_2 + f_1h_2 & d_1c_2 + e_1f_2 + f_1i_2 \\ g_1a_2 + h_1d_2 + i_1g_2 & g_1b_2 + h_1e_2 + i_1h_2 & g_1c_2 + h_1f_2 + i_1i_2 \end{pmatrix}$$

# Standard Matrix Multiplications Algorithm

**Standard Matrix Multiplication Algorithm**

The $(i,j)$-entry of $A \times B$ is $\sum_{k=1}^{n} A[i,k]B[k,j]$, i.e.,

$$\begin{pmatrix} a_1 & b_1 & c_1 \\ d_1 & e_1 & f_1 \\ g_1 & h_1 & i_1 \end{pmatrix} \times \begin{pmatrix} a_2 & b_2 & c_2 \\ d_2 & e_2 & f_2 \\ g_2 & h_2 & i_2 \end{pmatrix} =$$

$$\begin{pmatrix} a_1a_2 + b_1d_2 + c_1g_2 & a_1b_2 + b_1e_2 + c_1h_2 & a_1c_2 + b_1f_2 + c_1i_2 \\ d_1a_2 + e_1d_2 + f_1g_2 & d_1b_2 + e_1e_2 + f_1h_2 & d_1c_2 + e_1f_2 + f_1i_2 \\ g_1a_2 + h_1d_2 + i_1g_2 & g_1b_2 + h_1e_2 + i_1h_2 & g_1c_2 + h_1f_2 + i_1i_2 \end{pmatrix}$$

**Standard matrix multiplication algorithm:** a three-nested loop.

- Inner-most loop: Compute value for an entry
- Middle loop: Compute values in a row
- Outer-most loop: Compute values in all rows

# Standard Matrix Multiplications Algorithm

**Standard Matrix Multiplication Algorithm**

The $(i, j)$-entry of $A \times B$ is $\sum_{k=1}^{n} A[i,k]B[k,j]$, i.e.,

$$\begin{pmatrix} a_1 & b_1 & c_1 \\ d_1 & e_1 & f_1 \\ g_1 & h_1 & i_1 \end{pmatrix} \times \begin{pmatrix} a_2 & b_2 & c_2 \\ d_2 & e_2 & f_2 \\ g_2 & h_2 & i_2 \end{pmatrix} =$$

$$\begin{pmatrix} a_1a_2 + b_1d_2 + c_1g_2 & a_1b_2 + b_1e_2 + c_1h_2 & a_1c_2 + b_1f_2 + c_1i_2 \\ d_1a_2 + e_1d_2 + f_1g_2 & d_1b_2 + e_1e_2 + f_1h_2 & d_1c_2 + e_1f_2 + f_1i_2 \\ g_1a_2 + h_1d_2 + i_1g_2 & g_1b_2 + h_1e_2 + i_1h_2 & g_1c_2 + h_1f_2 + i_1i_2 \end{pmatrix}$$

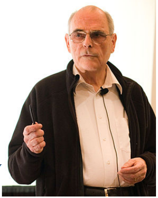
**Standard matrix multiplication algorithm:** a three-nested loop.

- Inner-most loop: Compute value for an entry
- Middle loop: Compute values in a row
- Outer-most loop: Compute values in all rows

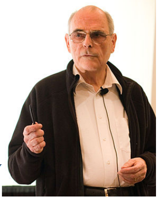
Time complexity $\Theta(n^3)$

Let's try
*Divide-and-Conquer*
on this problem

Volker Strassen (1969)
Professor of Math and Stats
University of Konstanz, Germany
Knuth Prize Winner 2008



Let's try
*Divide-and-Conquer*
on this problem



Volker Strassen (1969)
Professor of Math and Stats
University of Konstanz, Germany
Knuth Prize Winner 2008

**Observations**

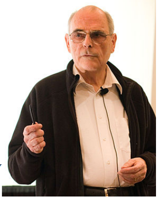
We may divide a $2n \times 2n$ matrix into four $n \times n$ sub-matrices.

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{pmatrix} \times \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} \\ b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} \\ b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} \\ b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} \end{pmatrix} =$$

$$\begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} \\ c_{2,1} & c_{2,2} & c_{2,3} & c_{2,4} \\ c_{3,1} & c_{3,2} & c_{3,3} & c_{3,4} \\ c_{4,1} & c_{4,2} & c_{4,3} & c_{4,4} \end{pmatrix}$$

Let's try
*Divide-and-Conquer*
on this problem

Volker Strassen (1969)
Professor of Math and Stats
University of Konstanz, Germany
Knuth Prize Winner 2008



**Observations**

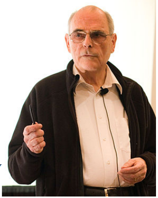
We may divide a $2n \times 2n$ matrix into four $n \times n$ sub-matrices.

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{pmatrix} \times \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} \\ b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} \\ b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} \\ b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} \end{pmatrix} =$$

$$\begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} \\ c_{2,1} & c_{2,2} & c_{2,3} & c_{2,4} \\ c_{3,1} & c_{3,2} & c_{3,3} & c_{3,4} \\ c_{4,1} & c_{4,2} & c_{4,3} & c_{4,4} \end{pmatrix}$$

## Let's try *Divide-and-Conquer* on this problem

Volker Strassen (1969)
Professor of Math and Stats
University of Konstanz, Germany
Knuth Prize Winner 2008



### Observations

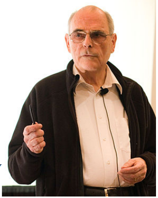
We may divide a $2n \times 2n$ matrix into four $n \times n$ sub-matrices.

$$\begin{pmatrix} \boxed{\begin{matrix} a_{1,1} & A_1 & a_{1,2} \\ a_{2,1} & & a_{2,2} \end{matrix}} & \boxed{\begin{matrix} a_{1,3} & A_2 & a_{1,4} \\ a_{2,3} & & a_{2,4} \end{matrix}} \\ \boxed{\begin{matrix} a_{3,1} & A_3 & a_{3,2} \\ a_{4,1} & & a_{4,2} \end{matrix}} & \boxed{\begin{matrix} a_{3,3} & A_4 & a_{3,4} \\ a_{4,3} & & a_{4,4} \end{matrix}} \end{pmatrix} \times \begin{pmatrix} \boxed{\begin{matrix} b_{1,1} & B_1 & b_{1,2} \\ b_{2,1} & & b_{2,2} \end{matrix}} & \boxed{\begin{matrix} b_{1,3} & B_2 & b_{1,4} \\ b_{2,3} & & b_{2,4} \end{matrix}} \\ \boxed{\begin{matrix} b_{3,1} & B_3 & b_{3,2} \\ b_{4,1} & & b_{4,2} \end{matrix}} & \boxed{\begin{matrix} b_{3,3} & B_4 & b_{3,4} \\ b_{4,3} & & b_{4,4} \end{matrix}} \end{pmatrix} =$$

$$\begin{pmatrix} \boxed{\begin{matrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{matrix}} & \begin{matrix} c_{1,3} & c_{1,4} \\ c_{2,3} & c_{2,4} \end{matrix} \\ \begin{matrix} c_{3,1} & c_{3,2} \\ c_{4,1} & c_{4,2} \end{matrix} & \begin{matrix} c_{3,3} & c_{3,4} \\ c_{4,3} & c_{4,4} \end{matrix} \end{pmatrix}$$

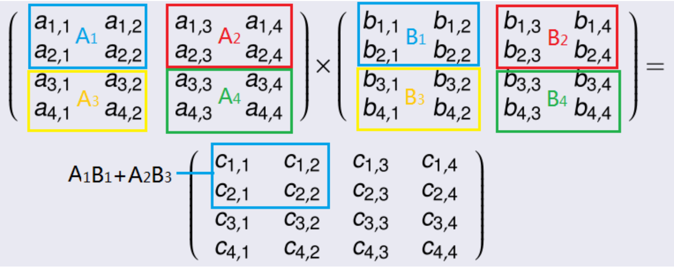
Let's try
*Divide-and-Conquer*
on this problem

Volker Strassen (1969)
Professor of Math and Stats
University of Konstanz, Germany
Knuth Prize Winner 2008

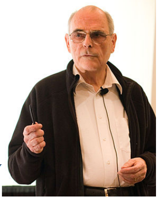


**Observations**

We may divide a $2n \times 2n$ matrix into four $n \times n$ sub-matrices.



Let's try
*Divide-and-Conquer*
on this problem



Volker Strassen (1969)
Professor of Math and Stats
University of Konstanz, Germany
Knuth Prize Winner 2008

**Observations**

We may divide a $2n \times 2n$ matrix into four $n \times n$ sub-matrices.

$$\left(\begin{array}{cc|cc} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ \hline a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{array}\right) \times \left(\begin{array}{cc|cc} b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} \\ b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} \\ \hline b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} \\ b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} \end{array}\right) =$$

$$\begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \times \begin{pmatrix} B_1 & B_2 \\ B_3 & B_4 \end{pmatrix}$$

$$\begin{pmatrix} A_1B_1 + A_2B_3 & A_1B_2 + A_2B_4 \\ A_3B_1 + A_4B_3 & A_3B_2 + A_4B_4 \end{pmatrix}$$

Example:

$$\begin{pmatrix} 1 & 4 & 3 & -1 \\ 0 & 2 & -2 & 4 \\ -1 & 0 & 1 & 0 \\ 5 & 2 & 1 & -2 \end{pmatrix} \times \begin{pmatrix} 3 & 1 & -1 & 1 \\ 1 & 0 & -2 & 3 \\ 2 & 3 & 1 & -3 \\ -1 & -2 & 0 & 1 \end{pmatrix}$$

Result:

$$\begin{pmatrix} 14 & 12 & -6 & 3 \\ -6 & -14 & -6 & 16 \\ -1 & 2 & 2 & -4 \\ 21 & 12 & -8 & 8 \end{pmatrix}$$

**First Attempt**

Recursively solve the 8 sub-matrices multiplications:

$$A_1B_1, A_2B_3, A_1B_2, A_2B_4, A_3B_1, A_4B_3, A_3B_2, A_4B_4$$

Then a fixed number of matrix additions ($\Theta(n^2)$-time). Thus

$$T(n) = 8T(n/2) + cn^2$$

**First Attempt**

Recursively solve the 8 sub-matrices multiplications:

$$A_1B_1, A_2B_3, A_1B_2, A_2B_4, A_3B_1, A_4B_3, A_3B_2, A_4B_4$$

Then a fixed number of matrix additions ($\Theta(n^2)$-time). Thus

$$T(n) = 8T(n/2) + cn^2$$

By Master theorem,

$$T(n) \in \Theta(n^3).$$

No improvement from the standard way.
First attempt fails.

# Strassen's Algorithm

**Next attempt**: "Group" some of the multiplications together so we need $< 8$ sub-matrix multiplication.

- $P_1 = A_1(B_2 - B_4)$
- $P_2 = (A_1 + A_2)B_4$
- $P_3 = (A_3 + A_4)B_1$
- $P_4 = A_4(B_3 - B_1)$
- $P_5 = (A_1 + A_4)(B_1 + B_4)$
- $P_6 = (A_2 - A_4)(B_3 + B_4)$
- $P_7 = (A_1 - A_3)(B_1 + B_2)$

$$\left(\begin{array}{c|c} A_1B_1 + A_2B_3 & A_1B_2 + A_2B_4 \\ \hline A_3B_1 + A_4B_3 & A_3B_2 + A_4B_4 \end{array}\right) =$$
$$\left(\begin{array}{c|c} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ \hline P_3 + P_4 & P_5 + P_1 - P_3 - P_7 \end{array}\right)$$

Thus we only need 7 multiplications of sub-matrices.

## Strassen's Algorithm

Given two input $n \times n$ matrices $A, B$, do the following:

- If $A, B$ have very small dimensions, directly multiply them
- Otherwise divide $A$,$B$ into $A_1, \ldots, A_4$, $B_1, \ldots, B_4$.
- Compute $P_1, \ldots, P_7$, each use one recursive call.
- Then add and subtract $P_i$'s to get the output matrix $A \times B$

## Strassen's Algorithm

Given two input $n \times n$ matrices $A, B$, do the following:

- If $A, B$ have very small dimensions, directly multiply them
- Otherwise divide $A$,$B$ into $A_1, \ldots, A_4$, $B_1, \ldots, B_4$.
- Compute $P_1, \ldots, P_7$, each use one recursive call.
- Then add and subtract $P_i$'s to get the output matrix $A \times B$

## Complexity

- Let $T(n)$ be the time it takes to multiples two $n \times n$ matrices.
- We have $T(n) = 7T(n/2) + cn^2$
- By Master theorem, $T(n)$ is $\Theta(n^{\log 7}) \approx \Theta(n^{2.808})$
- This is asymptotically better than $\Theta(n^3)$!

# Summary

In this lecture, we introduce Strassen's algorithm for matrix multiplication.

- Divide the problem of into seven subproblems of size $n/2$
- Conquer each sub-problem by solving them recursively
- Combine their solutions into a solution for the original problem

Time complexity: $O(n^{2.808})$.


THE UNIVERSITY OF
AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND