

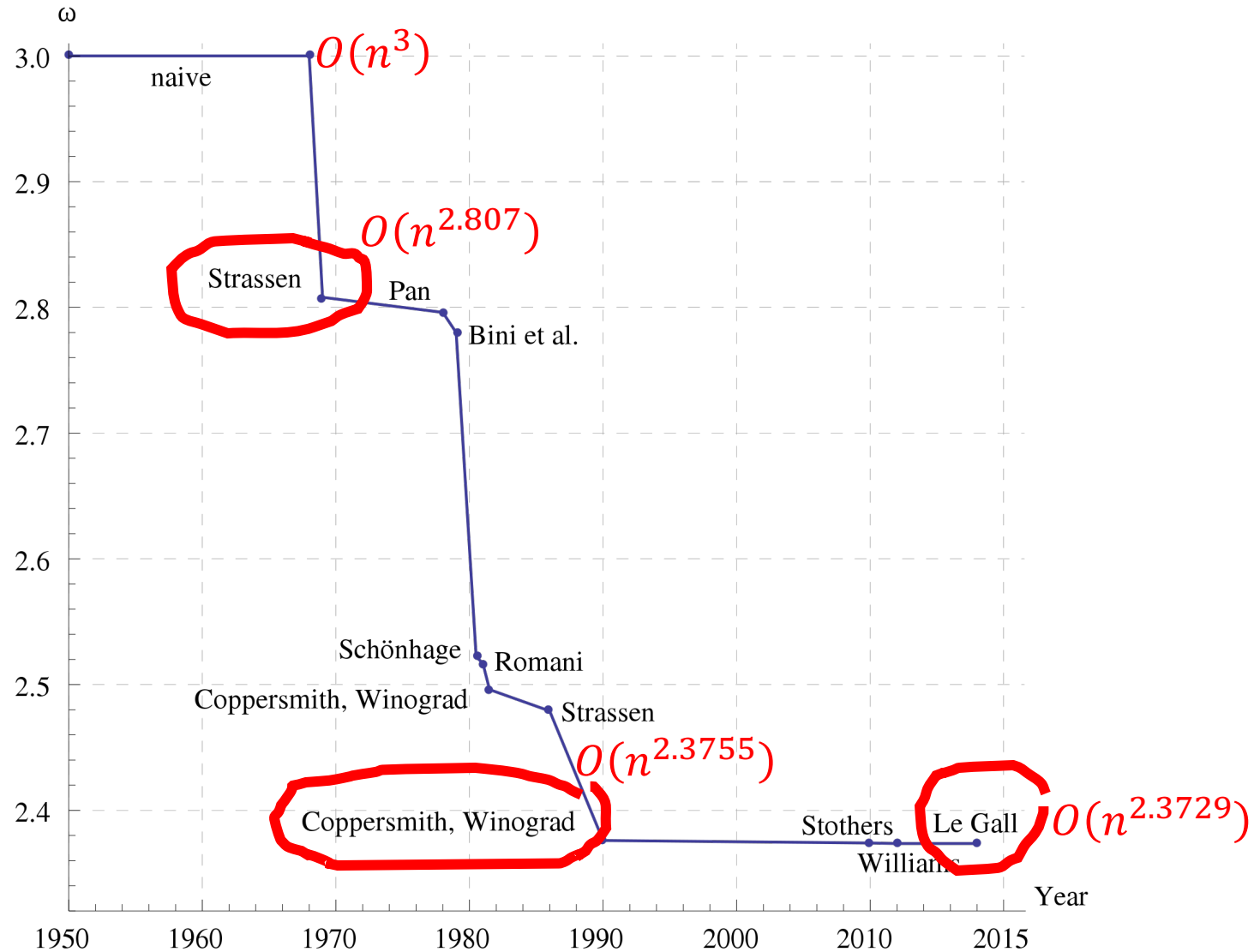
Fast Matrix Multiplication

Shusen Wang

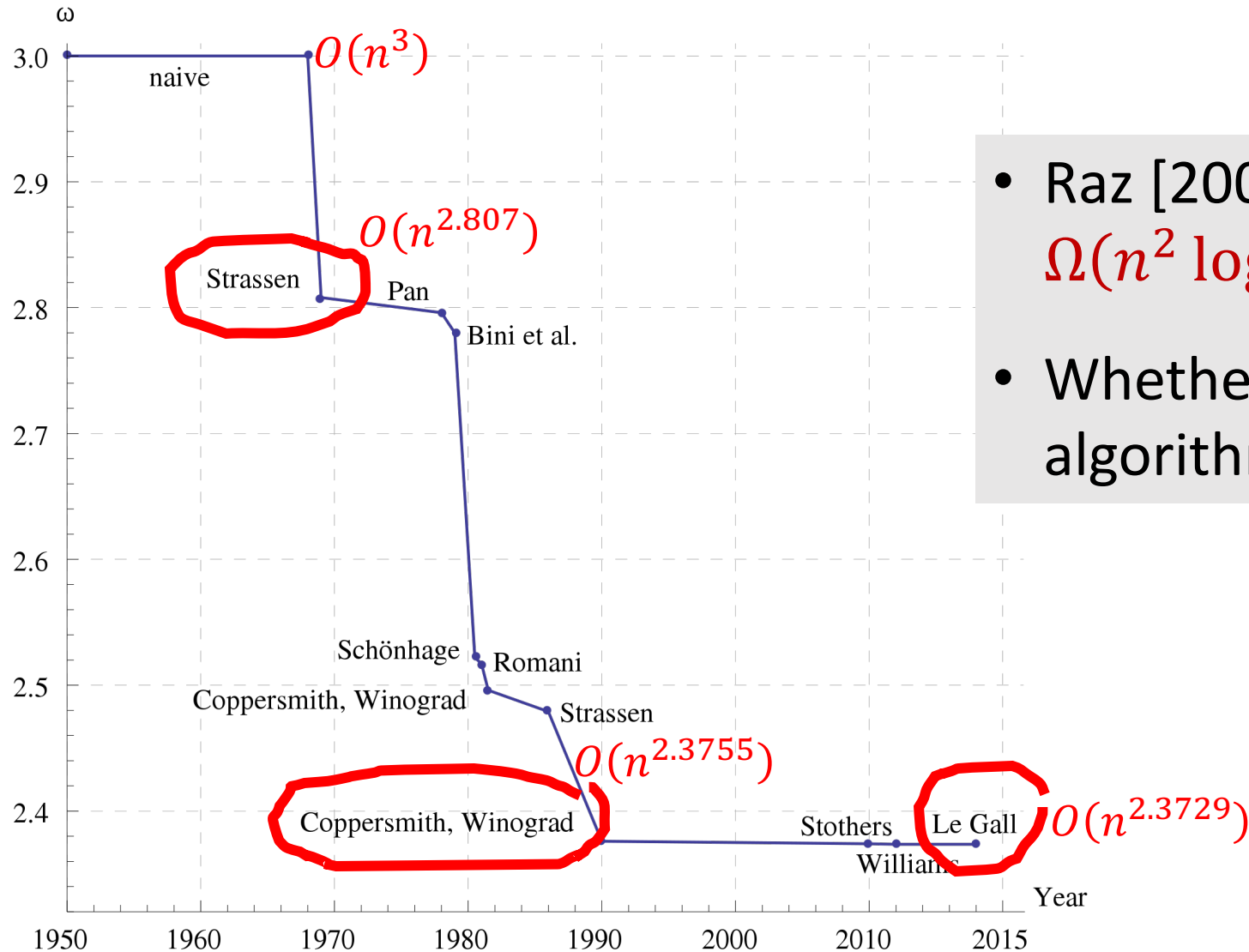
Fast Matrix Multiplication

- Let **A** and **B** be $n \times n$ matrices.
- The multiplication $\mathbf{C} = \mathbf{AB}$ cost $O(n^3)$ time (nested for-loop).
- Can the time complexity be lower?
- Strassen algorithm [1969] has $O(n^{2.807})$ time complexity.
- Coppersmith–Winograd algorithm [1990] has $O(n^{2.3755})$ time complexity.
- Le Gall [2014] improves the time complexity to $O(n^{2.3729})$.

Fast Matrix Multiplication



Fast Matrix Multiplication



- Raz [2002] proved a lower bound $\Omega(n^2 \log n)$.
- Whether there exists an $O(n^2 \log n)$ algorithm is yet unknown.

Divide-and-Conquer Matrix Multiplication

Block Matrix Multiplication

$\mathbf{C}_{1,1}$	$\mathbf{C}_{1,2}$
$\mathbf{C}_{2,1}$	$\mathbf{C}_{2,2}$

C

=

$\mathbf{A}_{1,1}$	$\mathbf{A}_{1,2}$
$\mathbf{A}_{2,1}$	$\mathbf{A}_{2,2}$

A

.

$\mathbf{B}_{1,1}$	$\mathbf{B}_{1,2}$
$\mathbf{B}_{2,1}$	$\mathbf{B}_{2,2}$

B

Block Matrix Multiplication

The diagram illustrates the multiplication of two 2x2 block matrices, A and B, to produce a 2x2 block matrix C. Matrix A is represented by a 2x2 grid of blocks: $A_{1,1}$ (pink), $A_{1,2}$ (orange), $A_{2,1}$ (gray), and $A_{2,2}$ (green). Matrix B is represented by a 2x2 grid of blocks: $B_{1,1}$ (light blue), $B_{1,2}$ (gray), $B_{2,1}$ (light purple), and $B_{2,2}$ (yellow). The resulting matrix C is represented by a 2x2 grid of blocks: $C_{1,1}$ (red), $C_{1,2}$ (yellow), $C_{2,1}$ (light blue), and $C_{2,2}$ (gray). The equation is shown as $C = A \cdot B$.

- $C_{1,1} = A_{1,1}B_{1,1} + A_{1,2}B_{2,1}.$

Block Matrix Multiplication

The diagram illustrates the multiplication of block matrices. On the left is the result matrix C , which is a 2x2 block matrix with blocks $C_{1,1}$ (dark grey), $C_{1,2}$ (red), $C_{2,1}$ (light blue), and $C_{2,2}$ (grey). This is followed by an equals sign, then matrix A , a dot, and matrix B . Matrix A is a 2x2 block matrix with blocks $A_{1,1}$ (green), $A_{1,2}$ (blue), $A_{2,1}$ (grey), and $A_{2,2}$ (light green). Matrix B is a 2x2 block matrix with blocks $B_{1,1}$ (light blue), $B_{1,2}$ (green), $B_{2,1}$ (purple), and $B_{2,2}$ (yellow).

- $C_{1,1} = A_{1,1}B_{1,1} + A_{1,2}B_{2,1}.$
- $C_{1,2} = A_{1,1}B_{1,2} + A_{1,2}B_{2,2}.$

Block Matrix Multiplication

$$\begin{bmatrix} \mathbf{C}_{1,1} & \mathbf{C}_{1,2} \\ \mathbf{C}_{2,1} & \mathbf{C}_{2,2} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \end{bmatrix}$$

- $\mathbf{C}_{1,1} = \mathbf{A}_{1,1}\mathbf{B}_{1,1} + \mathbf{A}_{1,2}\mathbf{B}_{2,1}.$
- $\mathbf{C}_{1,2} = \mathbf{A}_{1,1}\mathbf{B}_{1,2} + \mathbf{A}_{1,2}\mathbf{B}_{2,2}.$
- $\mathbf{C}_{2,1} = \mathbf{A}_{2,1}\mathbf{B}_{1,1} + \mathbf{A}_{2,2}\mathbf{B}_{2,1}.$

Block Matrix Multiplication

$$\begin{bmatrix} \mathbf{C}_{1,1} & \mathbf{C}_{1,2} \\ \mathbf{C}_{2,1} & \mathbf{C}_{2,2} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \end{bmatrix}$$

- $\mathbf{C}_{1,1} = \mathbf{A}_{1,1}\mathbf{B}_{1,1} + \mathbf{A}_{1,2}\mathbf{B}_{2,1}.$
- $\mathbf{C}_{1,2} = \mathbf{A}_{1,1}\mathbf{B}_{1,2} + \mathbf{A}_{1,2}\mathbf{B}_{2,2}.$
- $\mathbf{C}_{2,1} = \mathbf{A}_{2,1}\mathbf{B}_{1,1} + \mathbf{A}_{2,2}\mathbf{B}_{2,1}.$
- $\mathbf{C}_{2,2} = \mathbf{A}_{2,1}\mathbf{B}_{1,2} + \mathbf{A}_{2,2}\mathbf{B}_{2,2}.$

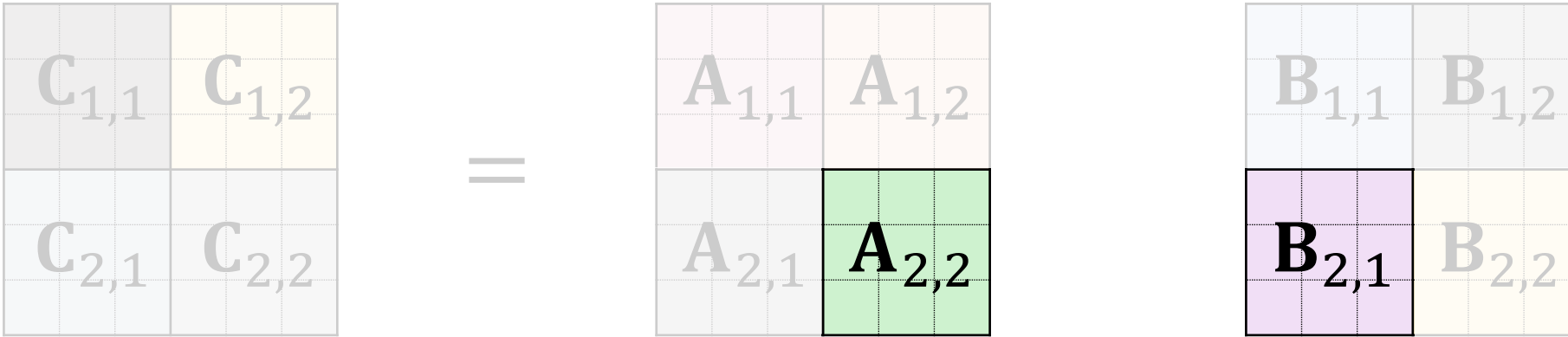
Block Matrix Multiplication

The diagram illustrates the block matrix multiplication $C = A \cdot B$. Matrix C is a 2x2 block matrix with blocks $C_{1,1}$, $C_{1,2}$, $C_{2,1}$, and $C_{2,2}$. Matrix A is a 2x2 block matrix with blocks $A_{1,1}$, $A_{1,2}$, $A_{2,1}$, and $A_{2,2}$. Matrix B is a 2x2 block matrix with blocks $B_{1,1}$, $B_{1,2}$, $B_{2,1}$, and $B_{2,2}$. The blocks are represented as 3x3 submatrices.

- $C_{1,1} = A_{1,1}B_{1,1} + A_{1,2}B_{2,1}.$
- $C_{1,2} = A_{1,1}B_{1,2} + A_{1,2}B_{2,2}.$
- $C_{2,1} = A_{2,1}B_{1,1} + A_{2,2}B_{2,1}.$
- $C_{2,2} = A_{2,1}B_{1,2} + A_{2,2}B_{2,2}.$

- 8 matrix multiplications
- 4 matrix additions

Divide-and-Conquer



- Further partition A_{ij} and B_{kl} into sub-matrices.
- Recursively apply the block multiplication to compute $A_{ij}B_{kl}$.

Does divide-and-conquer help?

- Partition the $n \times n$ matrices **A** and **B** into $\frac{n}{2} \times \frac{n}{2}$ sub-matrices.
- To compute the $n \times n$ matrix multiplication **C** = **AB**, perform
 - $\frac{n}{2} \times \frac{n}{2}$ matrix multiplications for 8 times,
 - $\frac{n}{2} \times \frac{n}{2}$ matrix additions for 4 times.

Does divide-and-conquer help?

- Partition the $n \times n$ matrices **A** and **B** into $\frac{n}{2} \times \frac{n}{2}$ sub-matrices.
- To compute the $n \times n$ matrix multiplication **C** = **AB**, perform
 - $\frac{n}{2} \times \frac{n}{2}$ matrix multiplications for 8 times,
 - $\frac{n}{2} \times \frac{n}{2}$ matrix additions for 4 times.
- Recursion relation: $T(n) = 8 \cdot T\left(\frac{n}{2}\right) + 4c \left(\frac{n}{2}\right)^2$.
 - $T(n)$: time complexity for multiplying $n \times n$ matrices.
 - cn^2 : time complexity for adding $n \times n$ matrices. (c is constant.)

The Master Theorem

Recurrence relation: $T(n) = a \cdot T(n/b) + c \cdot n^d$.

- The master theorem:

$$T(n) = \begin{cases} O(n^d), & \text{if } a < b^d; \\ O(n^d \log n), & \text{if } a = b^d; \\ O(n^{\log_b a}), & \text{if } a > b^d. \end{cases}$$

- For block matrix multiplication, $a = 8$, $b = 2$, and $d = 2$.

The Master Theorem

Recurrence relation: $T(n) = a \cdot T(n/b) + c \cdot n^d$.

- The master theorem:

$$T(n) = \begin{cases} O(n^d), & \text{if } a < b^d; \\ O(n^d \log n), & \text{if } a = b^d; \\ O(n^{\log_b a}), & \text{if } a > b^d. \end{cases}$$

- For block matrix multiplication, $a = 8$, $b = 2$, and $d = 2$.
- Thus, $T(n) = O(n^{\log_2 8}) \approx O(n^3)$.
- No speedup at all.

Strassen Algorithm

Strassen Algorithm

$C_{1,1}$	$C_{1,2}$
$C_{2,1}$	$C_{2,2}$

C

=

$A_{1,1}$	$A_{1,2}$
$A_{2,1}$	$A_{2,2}$

A

.

$B_{1,1}$	$B_{1,2}$
$B_{2,1}$	$B_{2,2}$

B

Strassen Algorithm

- $\mathbf{M}_1 = (\mathbf{A}_{1,1} + \mathbf{A}_{2,2})(\mathbf{B}_{1,1} + \mathbf{B}_{2,2}).$
- $\mathbf{M}_2 = (\mathbf{A}_{2,1} + \mathbf{A}_{2,2}) \mathbf{B}_{1,1}.$
- $\mathbf{M}_3 = \mathbf{A}_{1,1} (\mathbf{B}_{1,2} - \mathbf{B}_{2,2}).$
- $\mathbf{M}_4 = \mathbf{A}_{2,2} (\mathbf{B}_{2,1} - \mathbf{B}_{1,1}).$
- $\mathbf{M}_5 = (\mathbf{A}_{1,1} + \mathbf{A}_{1,2}) \mathbf{B}_{2,2}.$
- $\mathbf{M}_6 = (\mathbf{A}_{2,1} - \mathbf{A}_{1,1})(\mathbf{B}_{1,1} + \mathbf{B}_{1,2}).$
- $\mathbf{M}_7 = (\mathbf{A}_{1,2} - \mathbf{A}_{2,2})(\mathbf{B}_{2,1} + \mathbf{B}_{2,2}).$

- $\mathbf{C}_{1,1} = \mathbf{M}_1 + \mathbf{M}_4 - \mathbf{M}_5 + \mathbf{M}_7.$
- $\mathbf{C}_{1,2} = \mathbf{M}_3 + \mathbf{M}_5.$
- $\mathbf{C}_{2,1} = \mathbf{M}_2 + \mathbf{M}_4.$
- $\mathbf{C}_{2,2} = \mathbf{M}_1 - \mathbf{M}_2 + \mathbf{M}_3 + \mathbf{M}_6.$

- 7 matrix multiplications
- 18 matrix additions

Strassen Algorithm

- Divide-and-conquer: recursively compute $\mathbf{A}_{ij}\mathbf{B}_{kl}$ using the partitioning.
- To compute the $n \times n$ matrix multiplication $\mathbf{C} = \mathbf{AB}$, perform
 - $\frac{n}{2} \times \frac{n}{2}$ matrix multiplications for **7** times,
 - $\frac{n}{2} \times \frac{n}{2}$ matrix additions for 18 times.
- Recursion relation: $T(n) = \mathbf{7} \cdot T\left(\frac{n}{2}\right) + 18c \left(\frac{n}{2}\right)^2$.

The Master Theorem

Recurrence relation: $T(n) = a \cdot T(n/b) + c \cdot n^d$.

- The master theorem:

$$T(n) = \begin{cases} O(n^d), & \text{if } a < b^d; \\ O(n^d \log n), & \text{if } a = b^d; \\ O(n^{\log_b a}), & \text{if } a > b^d. \end{cases}$$

- For Strassen Algorithm, $a = 7$, $b = 2$, and $d = 2$.

The Master Theorem

Recurrence relation: $T(n) = a \cdot T(n/b) + c \cdot n^d$.

- The master theorem:

$$T(n) = \begin{cases} O(n^d), & \text{if } a < b^d; \\ O(n^d \log n), & \text{if } a = b^d; \\ O(n^{\log_b a}), & \text{if } a > b^d. \end{cases}$$

- For Strassen Algorithm, $a = 7$, $b = 2$, and $d = 2$.
- Thus, $T(n) = O(n^{\log_2 7}) \approx O(n^{2.807})$.

Summary

Block Matrix Multiplication

- Naively multiplying two $n \times n$ matrices costs $O(n^3)$ time.
- Block matrix multiplication:
 - Partition $n \times n$ matrices into four $\frac{n}{2} \times \frac{n}{2}$ sub-matrices.
 - Perform 8 multiplications and 4 additions.
- Divide-and-conquer does not help!
- The time complexity is still $O(n^{\log_2 8}) = O(n^3)$.

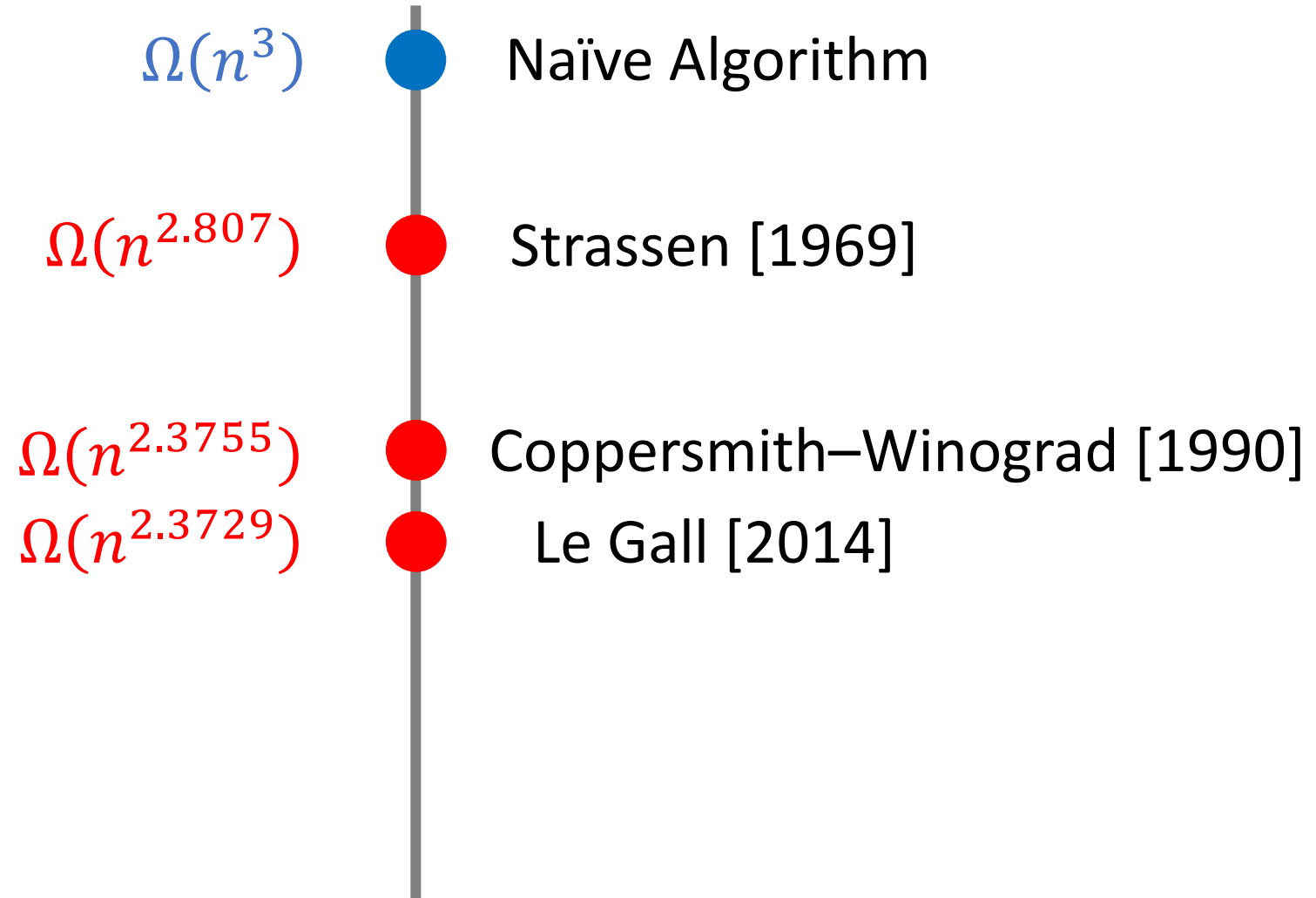
Strassen Algorithm

- Strassen algorithm also performs block matrix multiplication.
- Reduce the number of multiplications from 8 to 7.
- The time complexity is reduced to

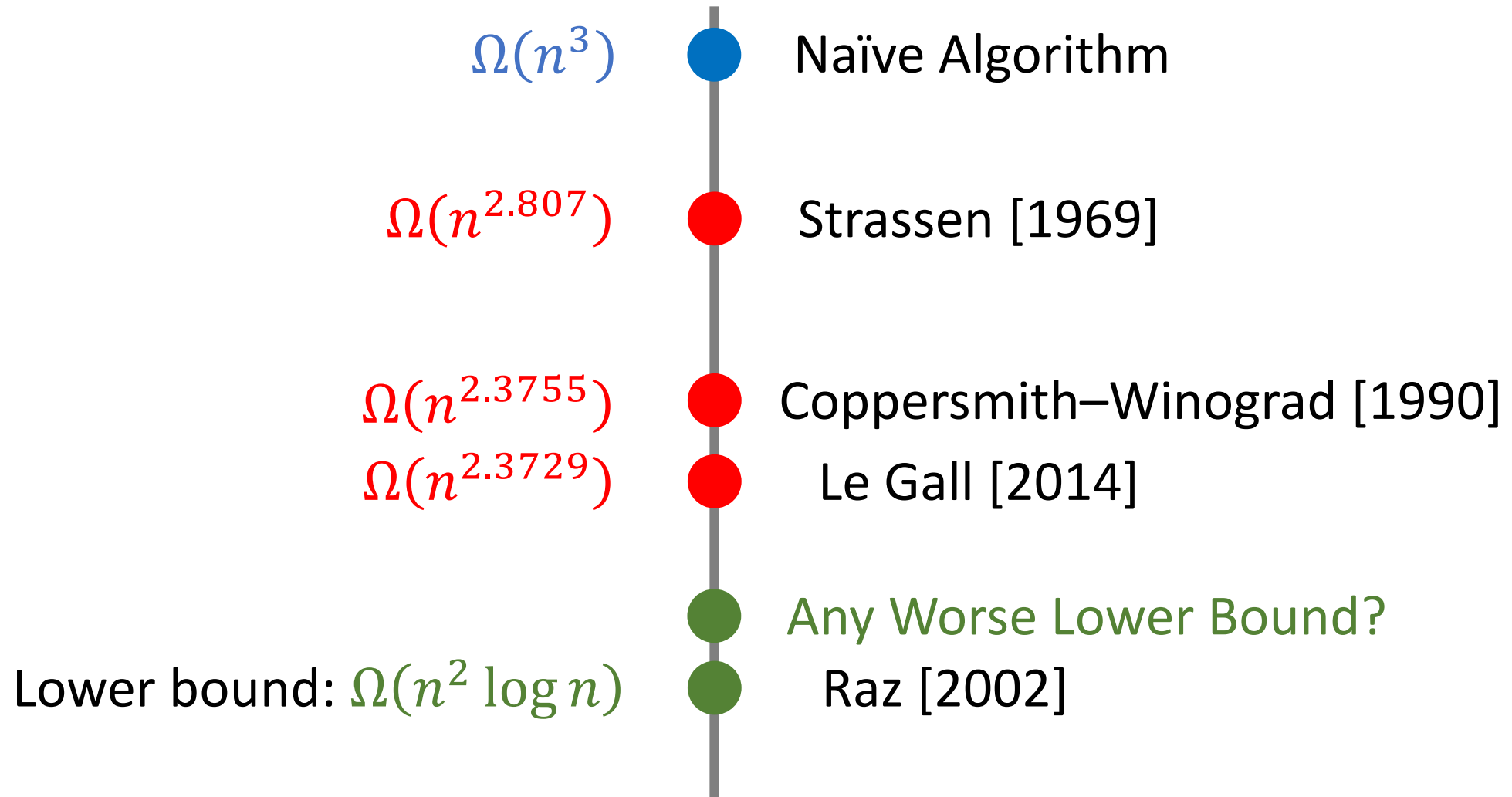
$$O(n^{\log_2 7}) \approx O(n^{2.807}).$$

- There are better algorithms than Strassen algorithm.

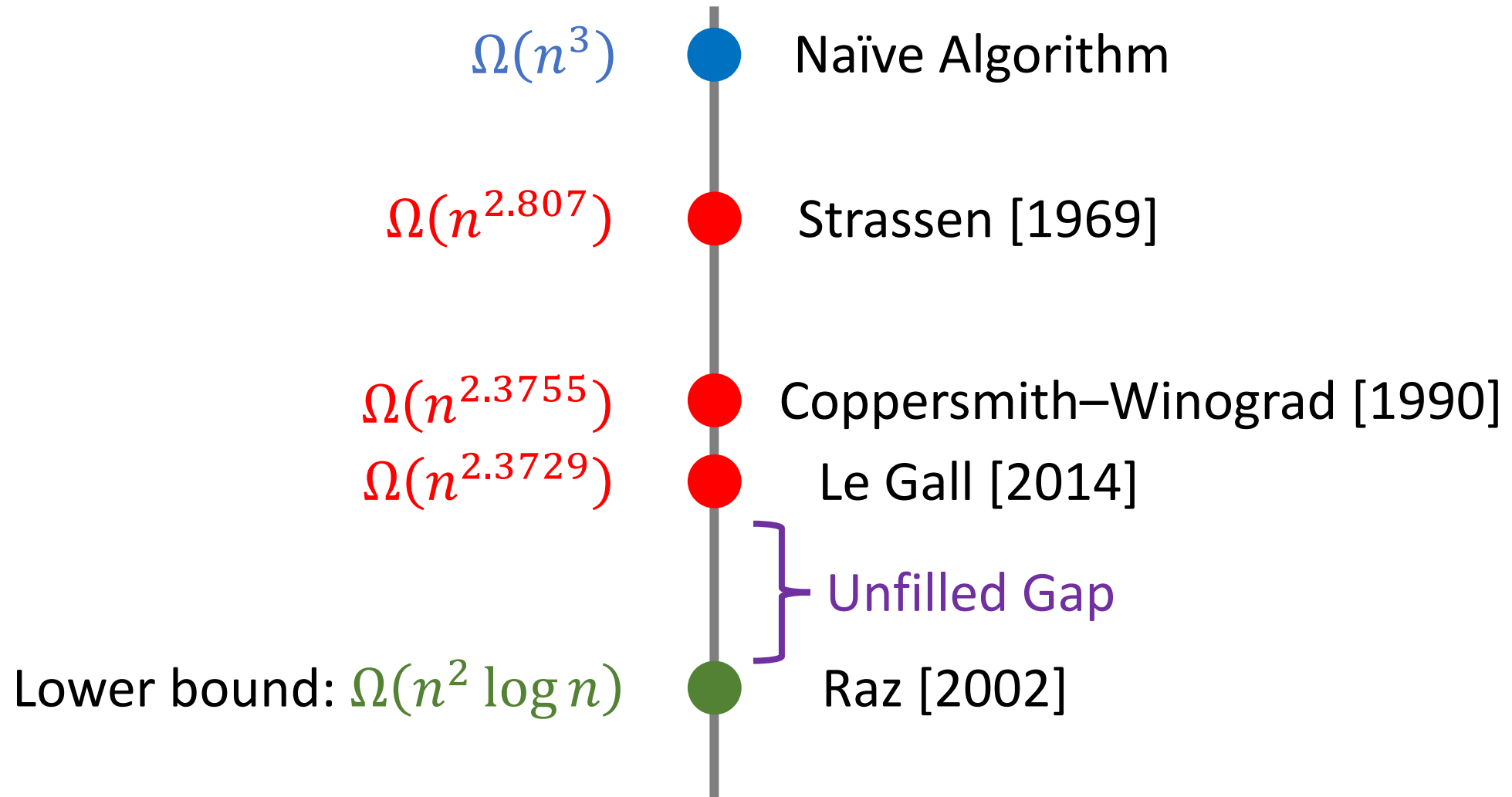
Fast Matrix Multiplication



Fast Matrix Multiplication



Fast Matrix Multiplication



Thank You!