Workshop #3
CSCI 432

Matrix Multiplication Verification

5	8	1
3	6	7
4	1	5

A

X

2	4	7
5	1	2
3	6	5

B

=

53	34	56
57	60	68
28	47	55

C

Problem: Given three $n \times n$ matrices A, B, and C, verify that $AB = C$.

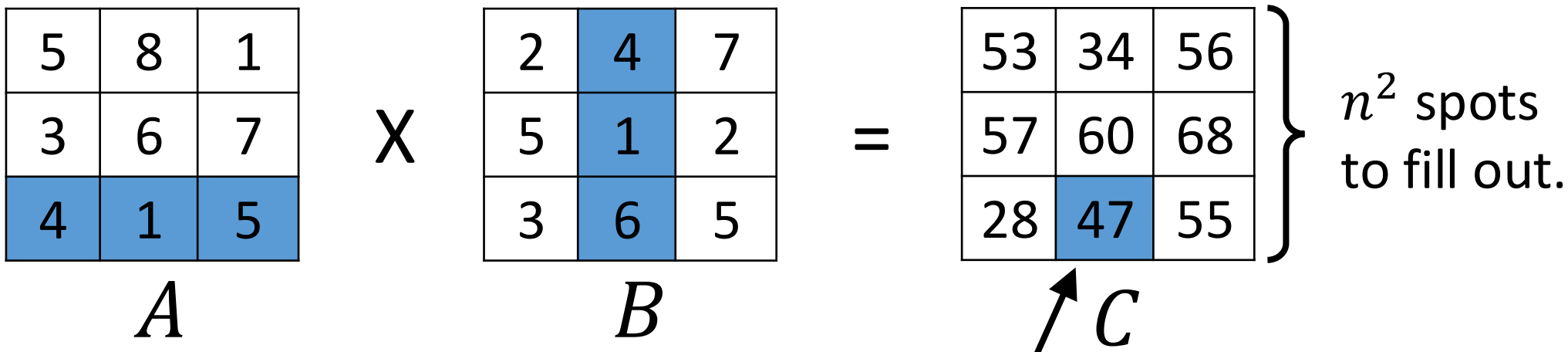
Obvious Solution?

Do the multiplication of A and B directly and compare to C.

Running Time?

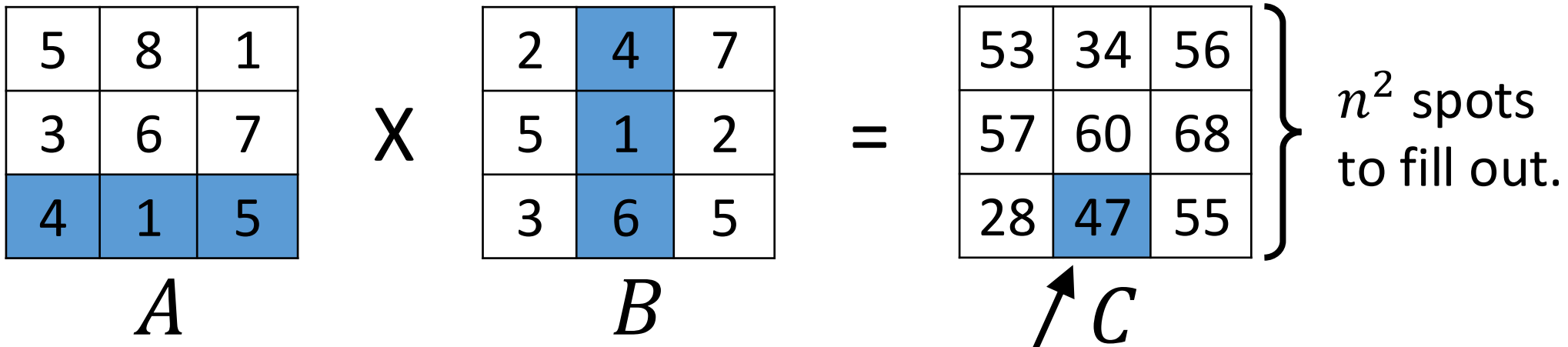
$O(n^3)$

Matrix Multiplication



Thus, $O(n^3)$ total running time.

Matrix Multiplication

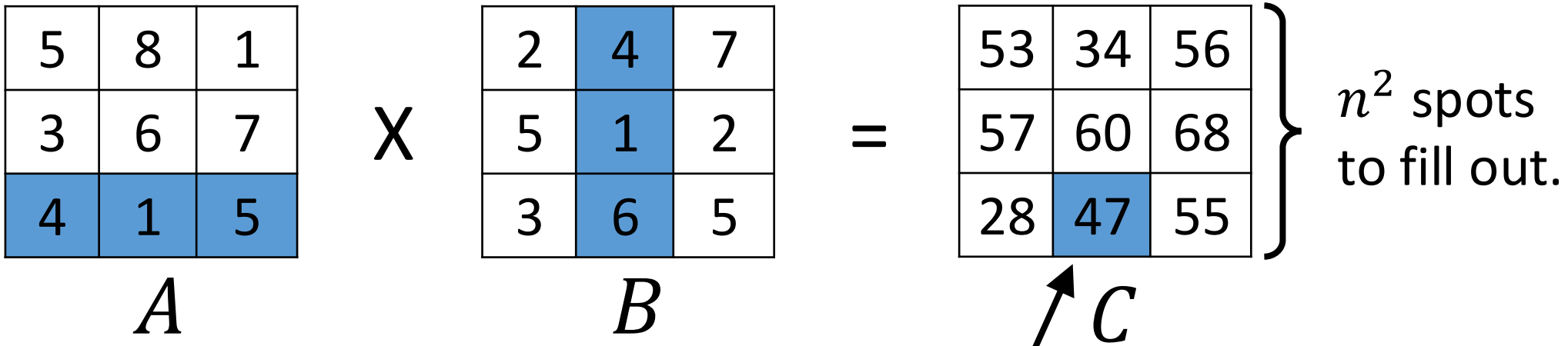


Thus, $O(n^3)$ total running time.

Goal Today: Multiply square matrices in less than $O(n^3)$ time.

Matrix Multiplication

But first, what is the running time for adding square matrices?



Result of multiplying a row by a column - $O(n)$ work

Thus, $O(n^3)$ total running time.

Goal Today: Multiply square matrices in less than $O(n^3)$ time.

Matrix Addition

5	8	1
3	6	7
4	1	5

X

+

2	4	7
5	1	2
3	6	5

Y

=

7	12	8
8	7	9
7	7	10

Z

$Z_{ij} = X_{ij} + Y_{ij}$

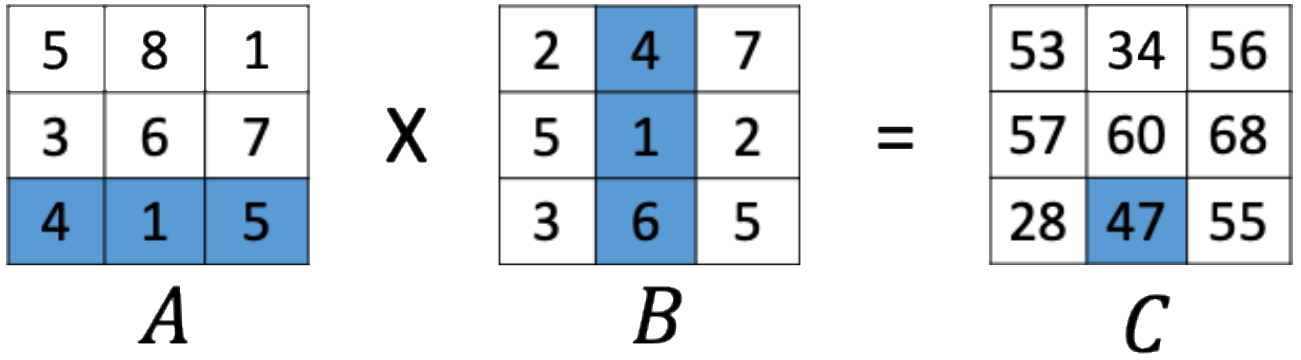
X, Y, Z are $n \times n$ matrices. Running time?

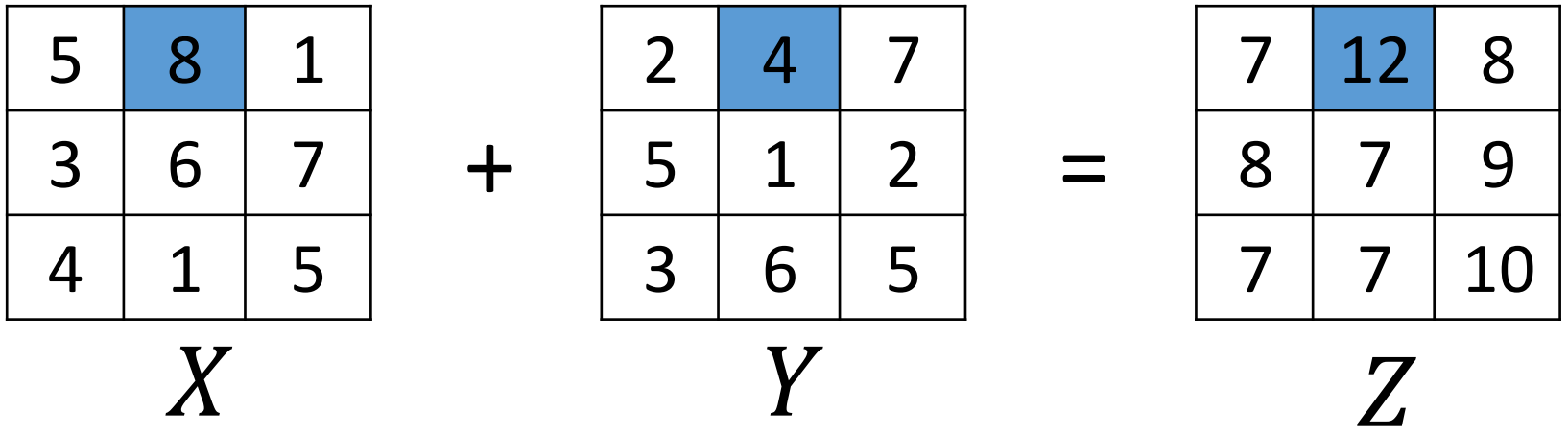
How big is the solution - n^2

How long to calculate a single entry - $O(1)$

Total time - $O(n^2)$

Matrix Addition





$Z_{ij} = X_{ij} + Y_{ij}$

X, Y, Z are $n \times n$ matrices. Running time?

How big is the solution - n^2

How long to calculate a single entry - $O(1)$

Total time - $O(n^2)$

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Alternate approach:

1. Recursively break X and Y into $\frac{n}{2} \times \frac{n}{2}$ blocks.
2. Multiply blocks, add to get Z.

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

C E

$$\begin{bmatrix} I & J \\ K & L \end{bmatrix} \begin{bmatrix} M & N \\ O & P \end{bmatrix} = \begin{bmatrix} IM + JO & IN + JP \\ KM + LO & KN + LP \end{bmatrix}$$

Alternate approach:

1. Recursively break X and Y into $\frac{n}{2} \times \frac{n}{2}$ blocks.
2. Multiply blocks, add to get Z.

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

$$\begin{bmatrix} I & J \\ K & L \end{bmatrix} \begin{bmatrix} M & N \\ O & P \end{bmatrix} = \begin{bmatrix} IM + JO & IN + JP \\ KM + LO & KN + LP \end{bmatrix}$$

C E

K M

$$\begin{bmatrix} Q & R \\ S & T \end{bmatrix} \begin{bmatrix} U & V \\ W & X \end{bmatrix} = \begin{bmatrix} QU + RW & QV + RX \\ SU + TW & SV + TX \end{bmatrix}$$

Alternate approach:

1. Recursively break X and Y into $\frac{n}{2} \times \frac{n}{2}$ blocks.
2. Multiply blocks, add to get Z.

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

Recurrence Relations Running Time

Recursive divide and conquer running time can be characterized as:

$$T(n) = aT(n/b) + D(n) + C(n)$$

Where, a – Number of subproblems at each recursive call

n/b – Size of subproblem relative to previous

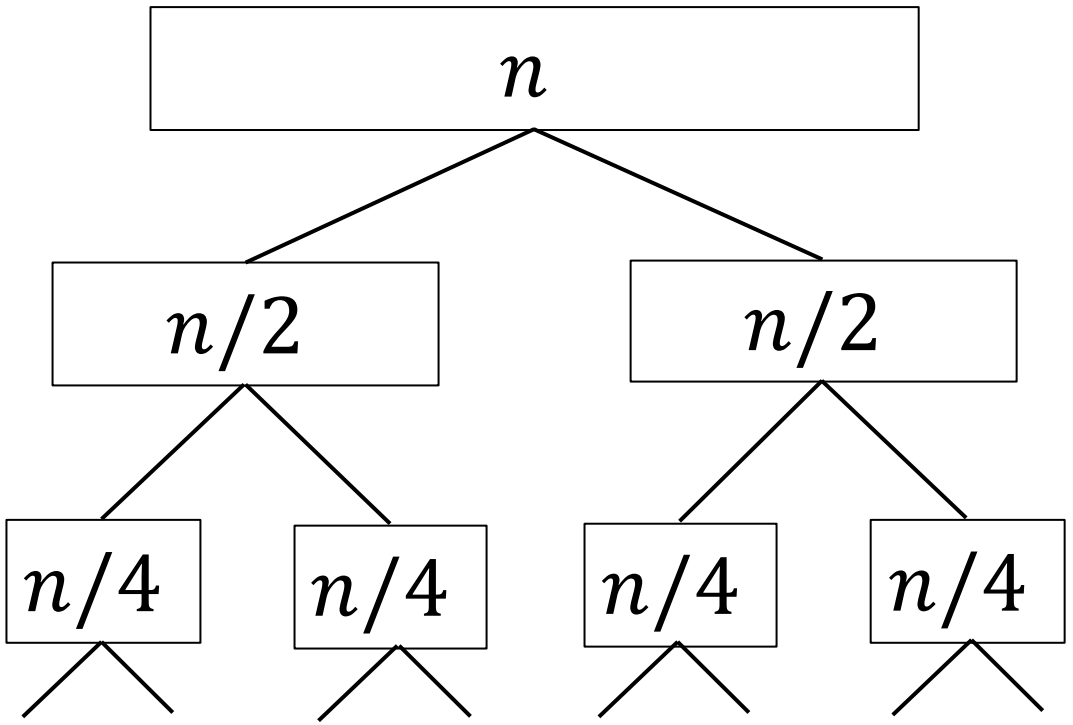
$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Master theorem: If $T(n) = aT(n/b) + O(n^d)$ for constants $a \geq 1, b > 1, d \geq 0$, then:

$$T(n) \in \begin{cases} O(n^d), & d > \log_b a \\ O(n^d \log n), & d = \log_b a \\ O(n^{\log_b a}), & d < \log_b a \end{cases}$$

Merge Sort – Running Time



Running Time $=$ Amount of work per level $\times$ Number of levels

$O(n)$

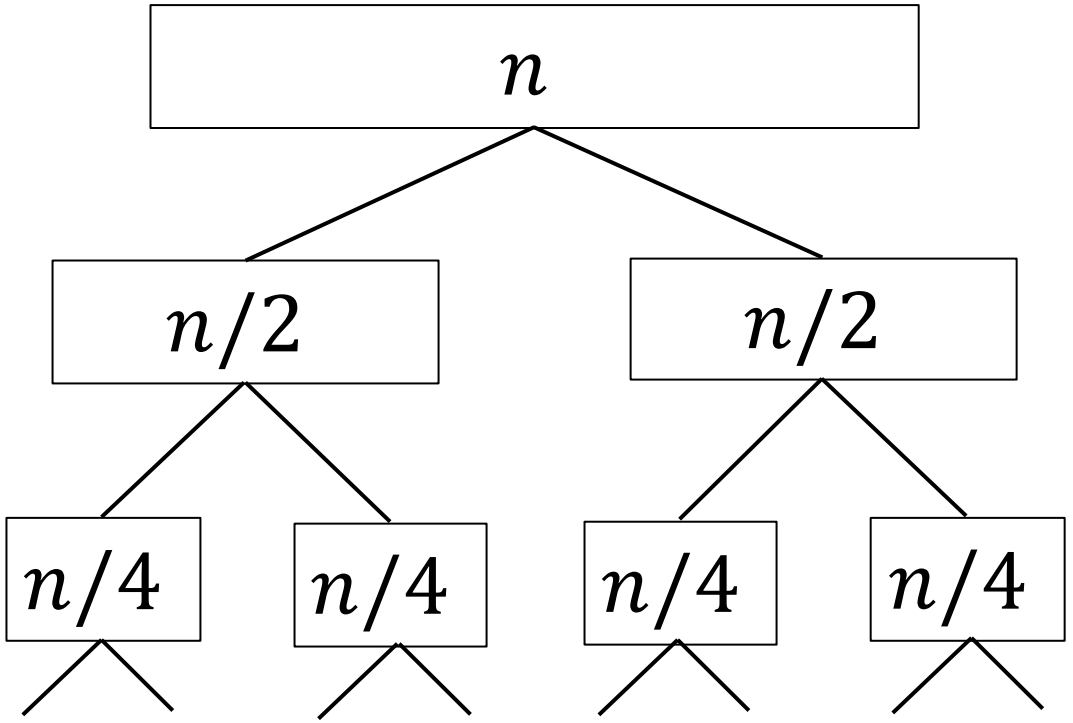
In level k, there are 2^k arrays. At the bottom, there are n arrays.

$\Rightarrow \frac{n}{2^k} = 1$

$\Rightarrow k = \log(n).$

$O(n \log(n))$ running time

Merge Sort – Running Time



Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$$T(n) = aT(n/b) + D(n) + C(n)$$

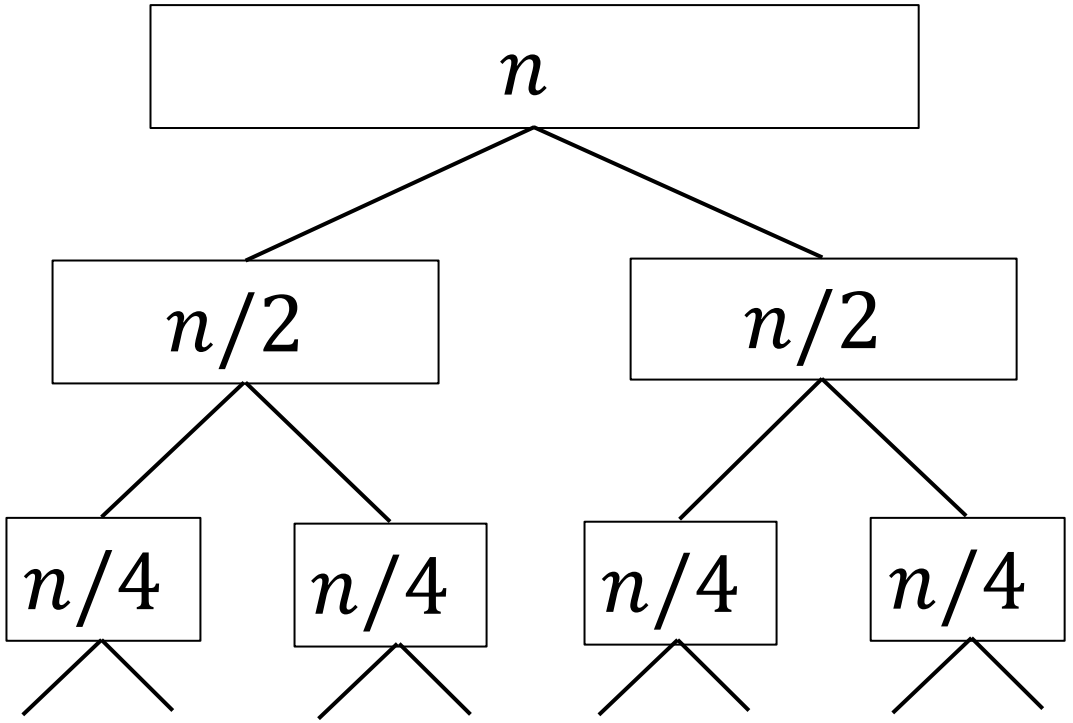
a – Number of subproblems

n/b – Size of subproblem

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Merge Sort – Running Time



Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$$T(n) = aT(n/b) + D(n) + C(n)$$

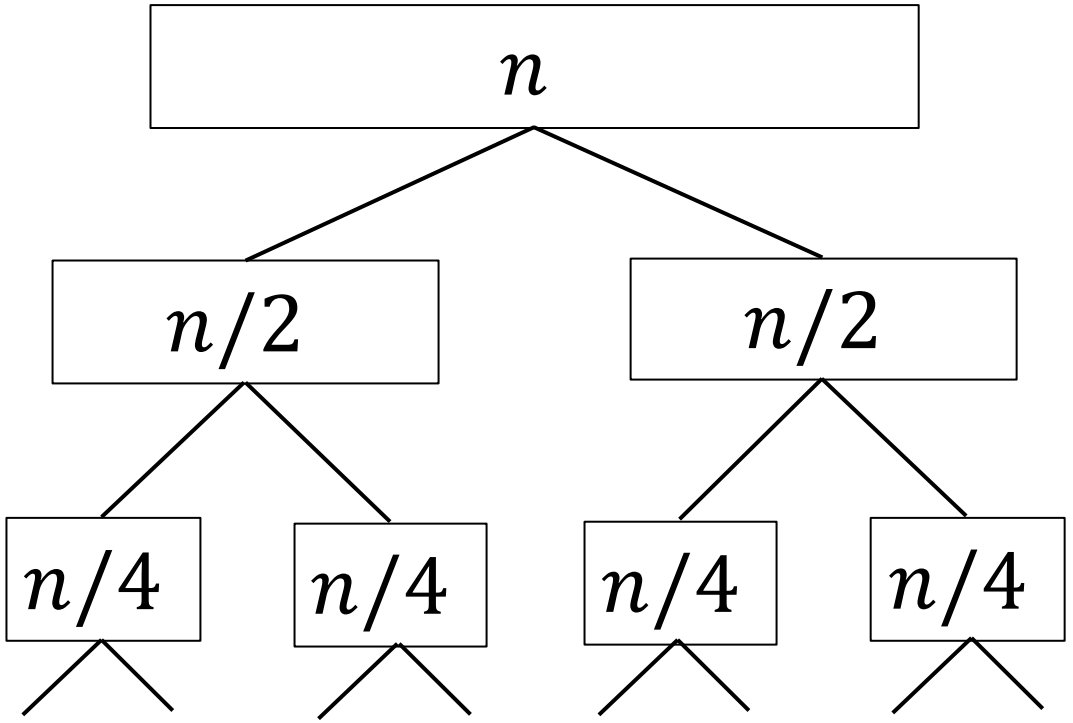
a – Number of subproblems **= 2**

n/b – Size of subproblem

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Merge Sort – Running Time



Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$T(n) = aT(n/b) + D(n) + C(n)$

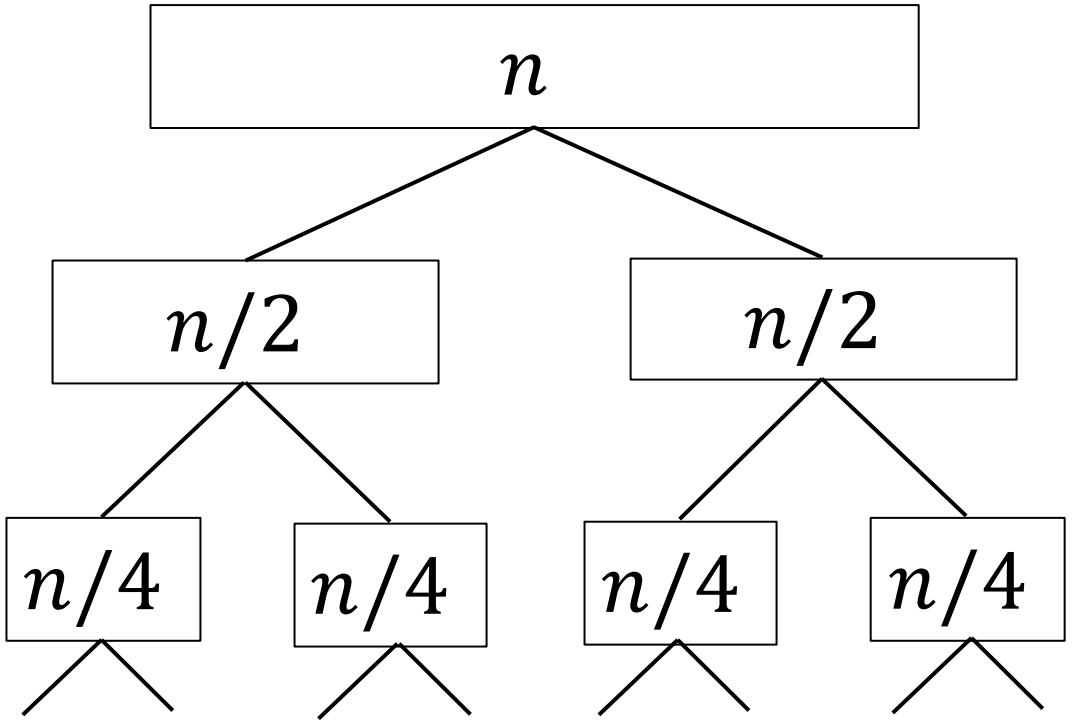
a – Number of subproblems **= 2**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Merge Sort – Running Time



Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$T(n) = aT(n/b) + D(n) + C(n)$

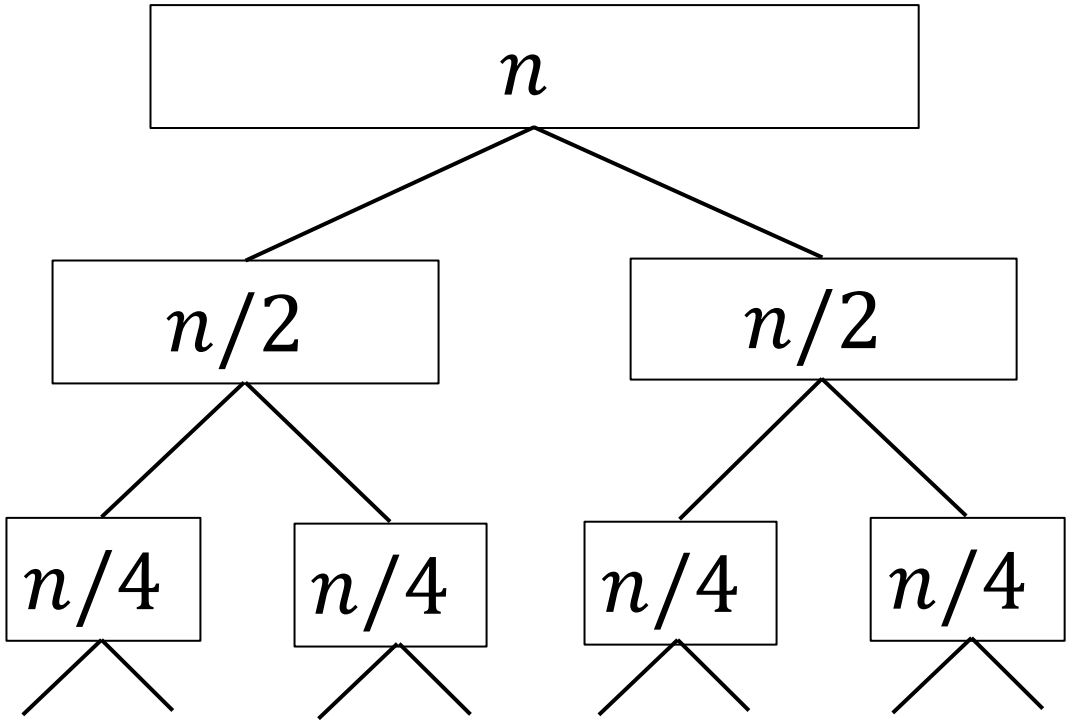
a – Number of subproblems **= 2**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n)$**

$C(n)$ – time to combine problems

Merge Sort – Running Time



Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$$T(n) = aT(n/b) + D(n) + C(n)$$

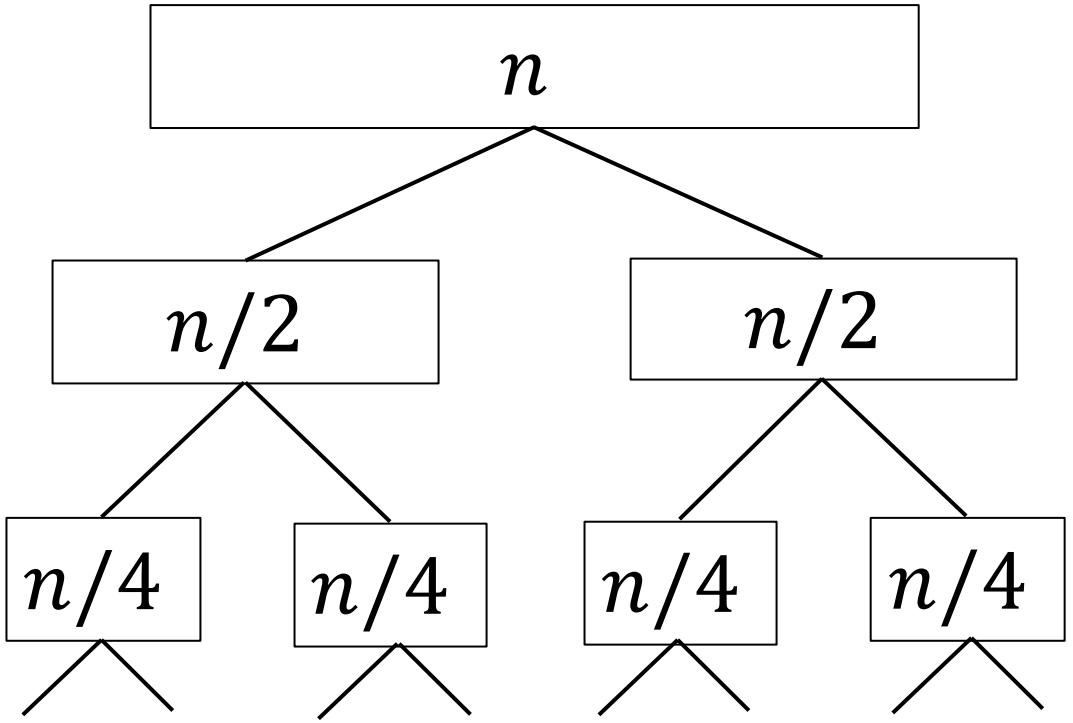
a – Number of subproblems **= 2**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n)$**

$C(n)$ – time to combine problems **= $O(n)$**

Merge Sort – Running Time



$\boldsymbol{log_b a =}$

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$T(n) = aT(n/b) + D(n) + C(n)$

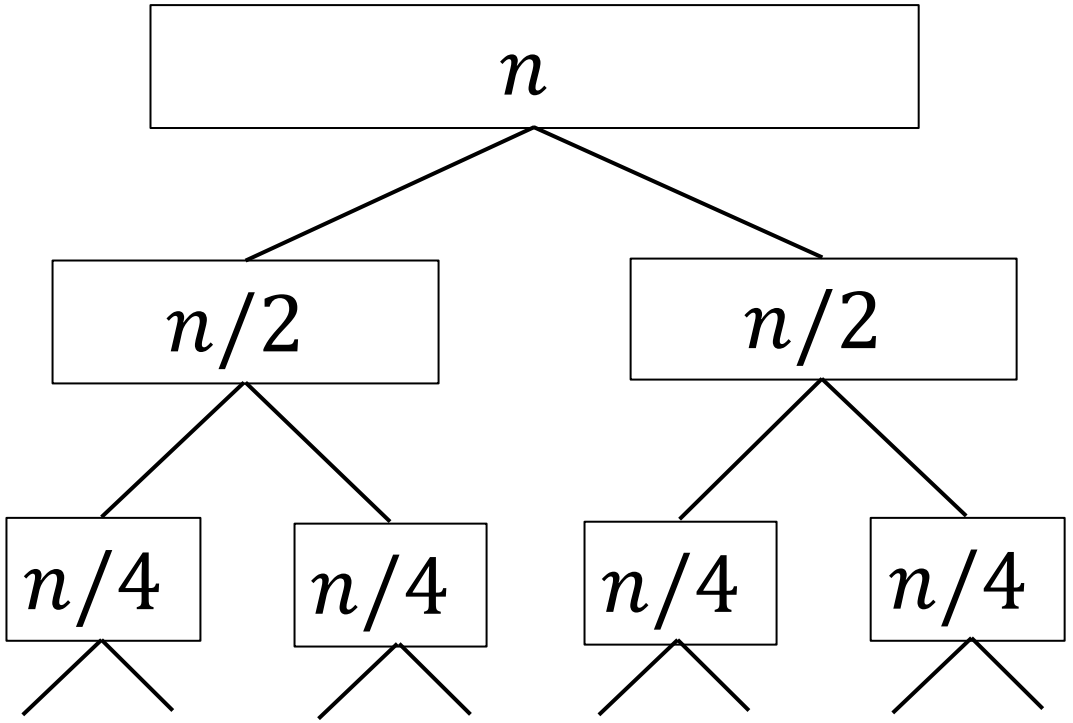
a – Number of subproblems **= 2**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n)$**

$C(n)$ – time to combine problems **= $O(n)$**

Merge Sort – Running Time



$$log_b a = log_2 2 = 1 = d$$

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$$T(n) = aT(n/b) + D(n) + C(n)$$

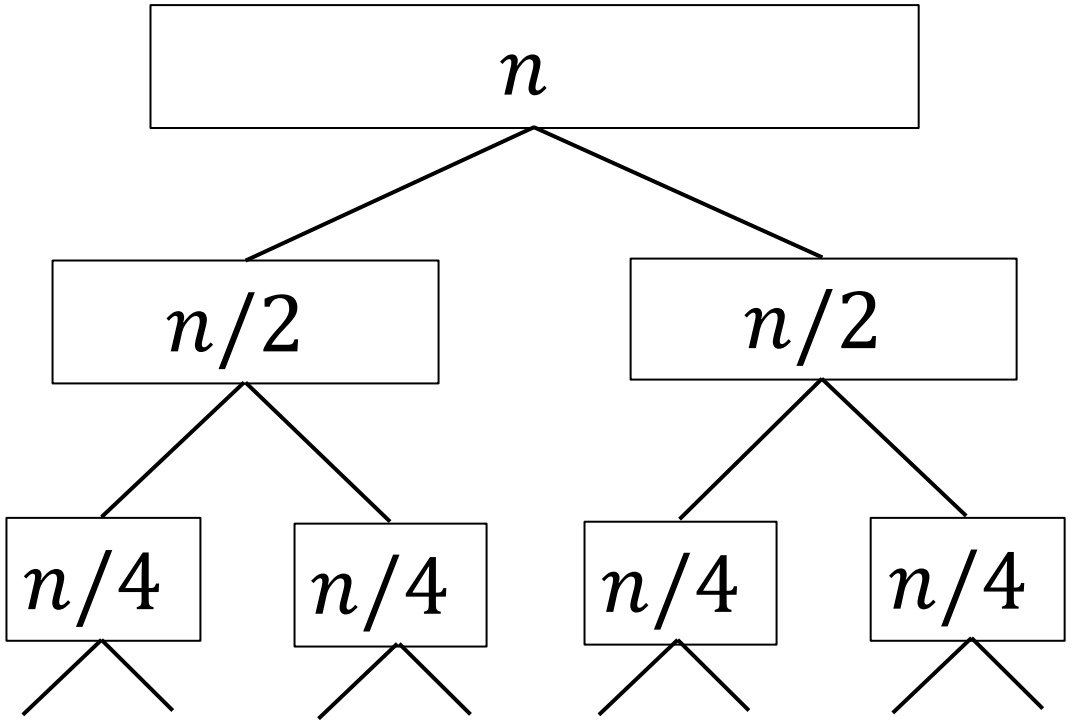
a – Number of subproblems **= 2**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n)$**

$C(n)$ – time to combine problems **= $O(n)$**

Merge Sort – Running Time



$\boldsymbol{log_b a = log_2 2 = 1 = d}$
$\boldsymbol{\Rightarrow T(n) = O(n\ \log n)}$

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems **= 2**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n)$**

$C(n)$ – time to combine problems **= $O(n)$**

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems

n/b – Size of subproblem

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE+BG & AF+BH \\ CE+DG & CF+DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems **= 8**

n/b – Size of subproblem

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems **= 8**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems **= 8**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems **= 8**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems **= $O(n^2)$**

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems **= 8**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems **= $O(n^2)$**

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$log_b a = log_2 8$

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems **= 8**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems **= $O(n^2)$**

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$\log_b a = \log_2 8 = 3 > 2 = d$

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D(n) + C(n)$

a – Number of subproblems **= 8**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems **= $O(n^2)$**

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$\boldsymbol{\log_b a = \log_2 8 = 3 > 2 = d}$

$\boldsymbol{\Rightarrow T(n) = O(n^3)}$

Matrix Multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Running time?

$T(n) = aT(n/b) + D($

a – Number [illegible] = **8**

n/b – Size [illegible]blem = $\boldsymbol{n/2}$

$D(n)$ – time to divide problems = $\boldsymbol{O(n^2)}$

$C(n)$ – time to combine problems = $\boldsymbol{O(n^2)}$

We've Accomplished Nothing.

[illegible]em:

$(n) = aT(n/b) + O(n^d)$, then

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

$\boldsymbol{\log_b a = \log_2 8 = 3 > 2 = d}$

$\boldsymbol{\Rightarrow T(n) = O(n^3)}$

Strassen's Algorithm

$$\begin{matrix}\begin{bmatrix} A & B \\ C & D \end{bmatrix} \\ X\end{matrix} \times \begin{matrix}\begin{bmatrix} E & F \\ G & H \end{bmatrix} \\ Y\end{matrix} = \begin{matrix}\begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix} \\ Z\end{matrix}$$

Strassen's Algorithm

$$P_1 = A(F - H) \quad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \quad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \quad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

X Y Z

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

X Y Z

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$X \qquad Y \qquad Z$

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

$\boldsymbol{P_1 + P_2}$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

$X \qquad Y \qquad Z$

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$X \qquad Y \qquad Z$

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

$\boldsymbol{P_1 + P_2}$
$\boldsymbol{= A(F - H) + (A + B)H}$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

$X \qquad Y \qquad Z$

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$X \qquad Y \qquad Z$

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

$$\boldsymbol{P_1 + P_2}$$
$$\boldsymbol{= A(F - H) + (A + B)H}$$
$$\boldsymbol{= AF - AH + AH + BH}$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

$X \qquad Y \qquad Z$

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$X \qquad Y \qquad Z$

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

$$\boldsymbol{P_1 + P_2}$$
$$\boldsymbol{= A(F - H) + (A + B)H}$$
$$\boldsymbol{= AF - AH + AH + BH}$$
$$\boldsymbol{= AF + BH}$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

$X \qquad Y \qquad Z$

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

X Y Z

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

Running time?

a – Number of subproblems

n/b – Size of subproblem

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

X Y Z

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

Running time?

a – Number of subproblems **= 7**

n/b – Size of subproblem

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

X Y Z

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

Running time?

a – Number of subproblems **= 7**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems

$C(n)$ – time to combine problems

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

X Y Z

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

Running time?

a – Number of subproblems **= 7**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

X Y Z

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

Running time?

a – Number of subproblems **= 7**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems **= $O(n^2)$**

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

$$log_b a = log_2 7 \cong 2.81 > 2 = d$$

Strassen's Algorithm

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ x } \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

X Y Z

for,

$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

Running time?

a – Number of subproblems **= 7**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems **= $O(n^2)$**

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Strassen's Algorithm

$$\log_b a = \log_2 7 \cong 2.81 > 2 = d$$
$$\Rightarrow T(n) = O(n^{2.81})$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$X \qquad Y \qquad Z$

for,
$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

Running time?

a – Number of subproblems **= 7**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems **= $O(n^2)$**

Master theorem:

If $T(n) = aT(n/b) + O(n^d)$, then

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

Strassen's Algorithm

$$\log_b a = \log_2 7 \cong 2.81 > 2 = d$$
$$\Rightarrow T(n) = O(n^{2.81})$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \times \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

X Y Z

for,
$$P_1 = A(F - H) \qquad P_5 = (A + D)(E + H)$$
$$P_2 = (A + B)H \qquad P_6 = (B - D)(G + H)$$
$$P_3 = (C + D)E \qquad P_7 = (A - C)(E + F)$$
$$P_4 = D(G - E)$$

Running time?

Master th

If $T(n) =$

$T(n) =$

a – Number of subproblems **= 7**

n/b – Size of subproblem **= $n/2$**

$D(n)$ – time to divide problems **= $O(n^2)$**

$C(n)$ – time to combine problems **= $O(n^2)$**

$O(n^{2.81})$ vs $O(n^3)$?

1 second vs 2.6 seconds

1 hour vs 4.5 hours

1 day vs 5.5 days

3 days vs 18 days

Strassen's Algorithm

$$log_b a = log_2 7 \cong 2.81 > 2 = d$$
$$\Rightarrow T(n) = O(n^{2.81})$$

History of Matrix Multiplication Algorithms

- $O(n^3)$ – 1812 (Matrix multiplication defined)
- $O(n^{2.81})$ – 1969 ← **Strassen's Algorithm**
- $O(n^{2.3755})$ – 1990
- $O(n^{2.3737})$ – 2010
- $O(n^{2.3729})$ – 2013
- $O(n^{2.3728639})$ – 2014
- $O(n^{2.3728596})$ – 2020
- $O(n^{2.371866})$ – 2022
- $O(n^{2.371339})$ – 2024
- $O(n^{2.371177})$ – 2026

Lower bound: $O(n^2)$. Need to look at all elements of both matrices.

$$P_6 \qquad P_1 + P_2$$
$$P_1 + P_5 - P_3 - P_7 \Big]$$

$$(A + D)(E + H)$$
$$(B - D)(G + H)$$
$$(A - C)(E + F)$$

th

=

=

$O(n^{2.81})$ vs $O(n^3)$?

- **1 second vs 2.6 seconds**
- **1 hour vs 4.5 hours**
- **1 day vs 5.5 days**
- **3 days vs 18 days**