Chapter 13

Divide and Conquer Algorithms

Divide and Conquer Algorithms

- แนวคิดแบ่งแยก (Divide) และเอาชนะ (Conquer)
- หลักการสำคัญ : แก้ปัญหาโดยการแบ่งปัญหาออกเป็นปัญหาย่อยๆ แล้วทำการแก้ปัญหาย่อยก่อน แล้วนำผลลัพธ์ของปัญหาย่อยมารวมกัน
  - Divide : แบ่งปัญหาเป็นส่วนย่อยๆ
  - Conquer : แก้ปัญหาย่อย (ทำเป็น Recursion**)
  - Combine : รวมผลลัพธ์ที่ได้จากปัญหาย่อย

Factorial : Recursion Defined

- การเขียนอัลกอริทึมที่มีการ**<u>เรียกตัวเองซ้ำ</u>**
- Case Study: Factorial

$$\text{Factorial}(n) = \begin{cases} 1 & ; n = 0 \\ n*\underline{(n-1)*(n-2)*\ldots*3*2*1} & ; n>0 \end{cases}$$

$$n*\underline{\text{Factorial}(n-1)}$$

- Base case : ส่วนหยุดการเรียกซ้ำ -> Factorial(0)
- Recursive case : ส่วนทำการเรียกซ้ำ -> n*Factorial(n-1)

Divide and Conquer Algorithms

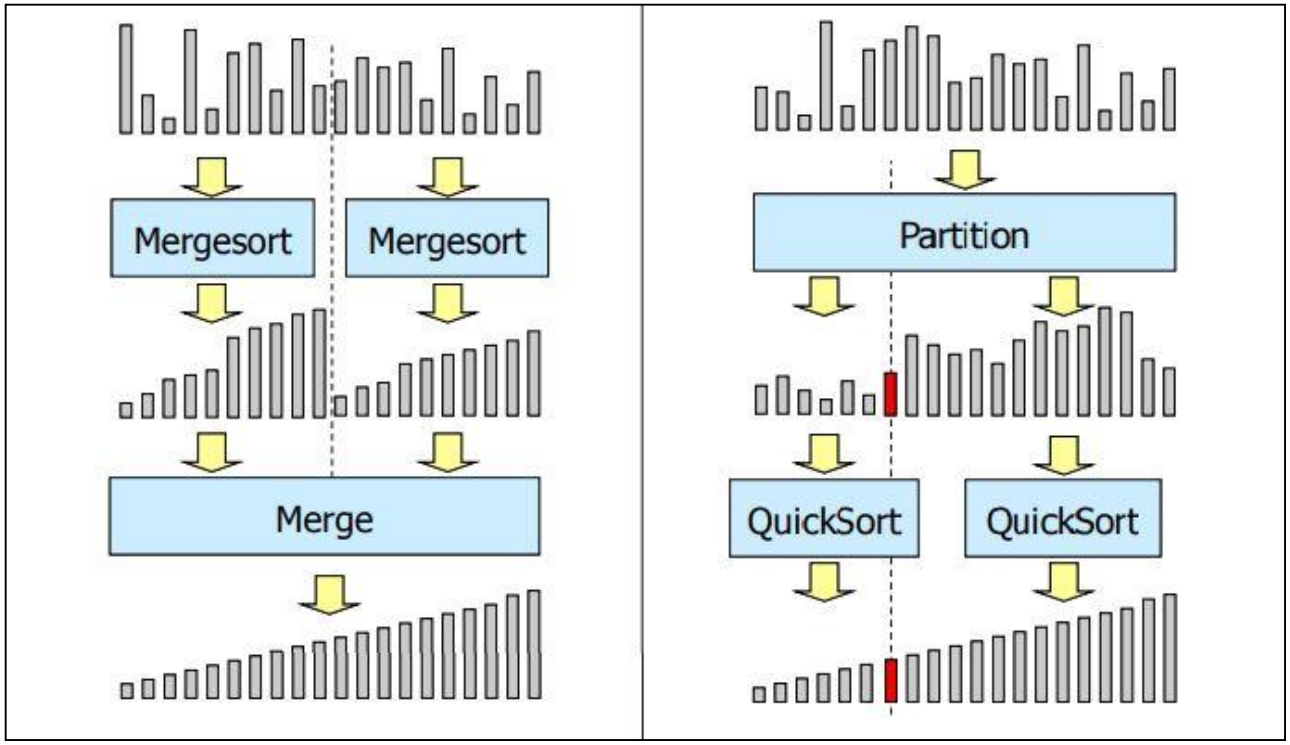
```
DQ( P ) {
  if ( P is trivial ) return Solve( P )
  Divide P into P1, P2, …, Pk
  for ( i = 1 to k )
    Si = DQ( Pi )
  S = Combine( S1, S2, …, Sk )
  return S
}
```

ตัวอย่างปัญหาที่ใช้ Divide & Conquer

- Sorting
- Modular exponentiation
- Integer multiplication
- Matrix multiplication

Sorting

- Merge Sort
- Quick Sort

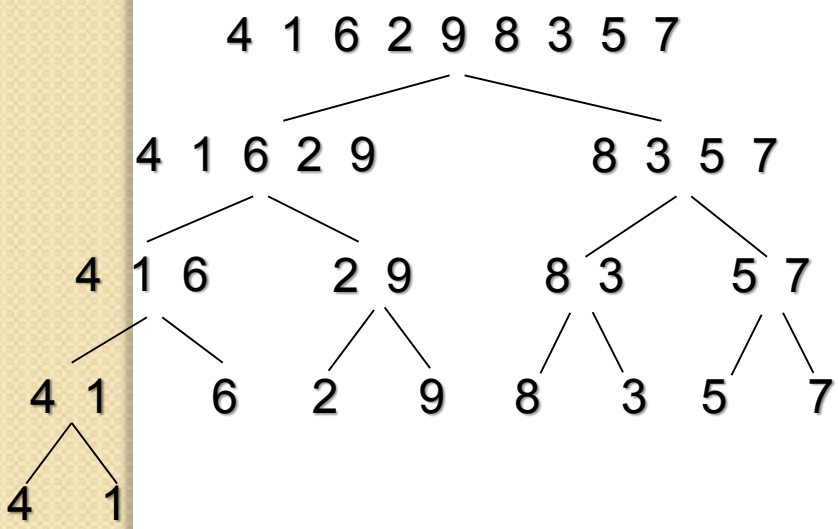


Merge Sort

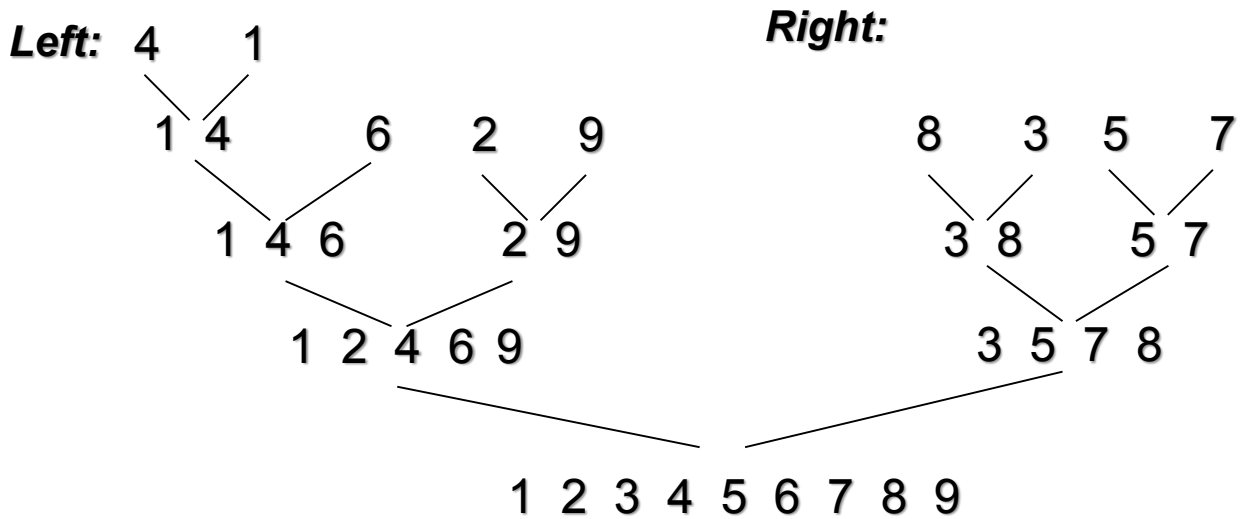
- Divide :
  - ทำการแบ่งลิสต์ข้อมูลออกทีละ 2 ส่วน (Sub-list) เท่าๆ กัน จนถึงระดับที่ไม่สามารถแบ่งได้อีก (Sub-list มีข้อมูล 1 ตัว)
- Conquer :
  - ทำการเรียงลำดับข้อมูลในแต่ละ Sub-list
- Combine :
  - ทำการผสาน (Merge) Sub-list ที่อยู่ติดกัน ให้เป็นลิสต์เดียว ไล่ขึ้นไปเรื่อยๆ จนเหลือลิสต์ข้อมูลเพียงลิสต์เดียว

Merge Sort Example

1. Devide unsorted list into two sublists



2. Merging and sorting

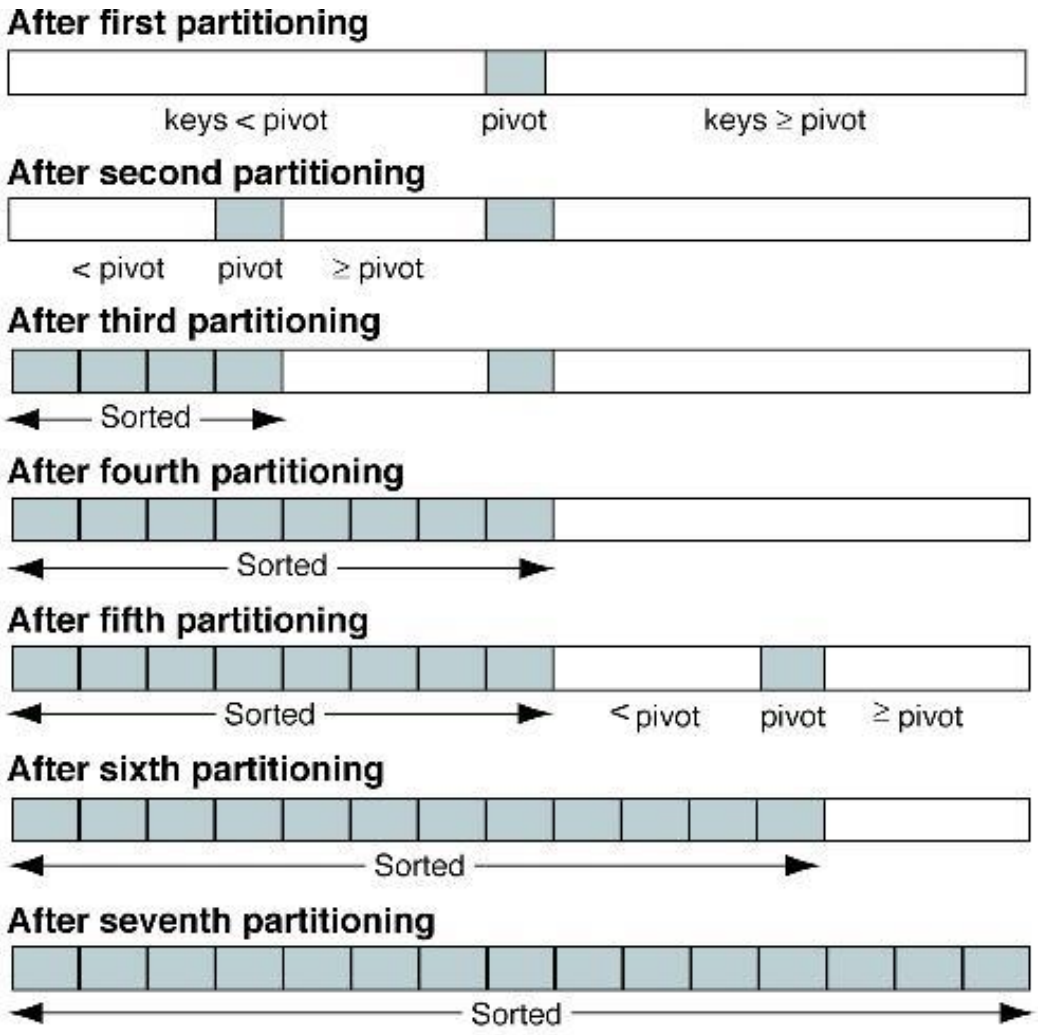


Sorting result

Quick Sort

- Divide
  - แบ่งลิสต์ข้อมูลออกเป็น 2 ชุด ด้วยค่า Pivot
    - ลิสต์ข้อมูลฝั่งซ้ายจะต้องมีค่าน้อยกว่า Pivot
    - ลิสต์ข้อมูลฝั่งขวาจะต้องมีค่ามากกว่าหรือเท่ากับ Pivot
- Conquer
  - ทำการจัดเรียงข้อมูลในลิสต์ตามกระบวนการ Partition (วิ่งพอยน์เตอร์หาตำแหน่งที่ถูกต้องของ Pivot)
- Combine
  - การจัดเรียงตำแหน่งของ Pivot ของแต่ละลิสต์ย่อย

Quick Sort


After first partitioning
keys < pivot
pivot
keys ≥ pivot
After second partitioning
< pivot
pivot
≥ pivot
After third partitioning
Sorted
After fourth partitioning
Sorted
After fifth partitioning
Sorted
< pivot
pivot
≥ pivot
After sixth partitioning
Sorted
After seventh partitioning
Sorted

Modular exponentiation

- Modulo Operation : คืนค่าเศษที่ได้จากการหาร
  - 5 mod 3 = 2
  - 17 mod 7 = 3
- (a*b) mod m = [(a mod m) * (b mod m)] mod m
  - (3*5) mod 4
    - [(3 mod 4) * (5 mod 4)] mod 4
    - (3*1) mod 4 = 3
- 2^{20} mod 10 = ?

$[(2^{10} \bmod 10) \times (2^{10} \bmod 10)] \bmod 10$

2^{20} mod 10 = ?

- Brute force
  - 2 * 2 * 2 * … * 2 (2 คูณกัน 20 ครั้ง)
  - นำผลลัพธ์ที่ได้ไป mod 10

$\}\ O(N)$

- มีทางที่ดีกว่านี้ไหม ?
  - ใช้ Divide and conquer
  - ย่อยปัญหา
    - 2^{20} mod 10 = [(2^{10} mod 10) * (2^{10} mod 10)] mod 10
    - = $(2^{10}\ mod\ 10)^2$ mod 10
    - $2^{10}\ mod\ 10$ = ?

$2^{20} = 2^{10} \times 2^{10}$

2^{20} mod 10 = ?

- $2^{20} \bmod 10 = (2^{10} \bmod 10)^2 \bmod 10$ $= 6$
- $2^{10} \bmod 10 = (2^{5} \bmod 10)^2 \bmod 10$ $= 4$
- $2^{5} \bmod 10 = 2 * (2^{2} \bmod 10)^2 \bmod 10$ $= 2$
- $2^{2} \bmod 10 = (2^{1} \bmod 10)^2 \bmod 10$ $= 4$
- $2^{1} \bmod 10 = 2 * (2^{0} \bmod 10)^2 \bmod 10$ $= 2$
- $2^{0} \bmod 10 = 1$

$O(\log n) = 550W$

2^{20} mod 10 = ?

- 2^{20} mod 10 = $(2^{10} \text{ mod } 10)^2$ mod 10 = 6 Ans
- 2^{10} mod 10 = $(2^{5} \text{ mod } 10)^2$ mod 10 = 4
- 2^{5} mod 10 = 2 * $(2^{2} \text{ mod } 10)^2$ mod 10 = 2
- 2^{2} mod 10 = $(2^{1} \text{ mod } 10)^2$ mod 10 = 4
- 2^{1} mod 10 = 2 * $(2^{0} \text{ mod } 10)^2$ mod 10 = 2
- 2^{0} mod 10 = 1

- O(log n) ≈ รอบ

Integer multiplication: Brute Force

- การคูณเลขหลายหลัก (สมัยมัธยมฯ) ทำยังไง
- 2018 * 1234

n=4

```
      2018
      1234
  --------
      8072  → 4
     6054   → 4
    4036    → 4
   2018     → 4
  -------
   2490212
```

16 ครั้ง (n^2)

- $O(n^2)$: n คือจำนวนหลักของตัวเลข -> มีวิธีที่เร็วกว่านี้ไหม ?

Integer multiplication

	$n/2$ digits	$n/2$ digits
$A =$	A_L 20	A_R 18
$B =$	B_L 12	B_R 34

$$A \times B = (A_L 10^{n/2} + A_R) \times (B_L 10^{n/2} + B_R)$$
$$= A_L \times B_L 10^n + (A_L \times B_R + A_R \times B_L) 10^{n/2} + A_R \times B_R$$

- 2018*1234 =
  - $((20 * 10^2) + 18) * ((12 * 10^2) + 34)$
  - $(20*12* 10^4) + ((20*34+18*12) * 10^2) + (18*34)$
- $O(n^2)$

Integer multiplication

- 2018 * 1234
  - $2018 = (20 * 10^2) + 18 \Rightarrow (a*10^{n/2}+b)$
  - $1234 = (12 * 10^2) + 34 \Rightarrow (c*10^{n/2}+d)$
  - $(a*10^{n/2}+b) * (c*10^{n/2}+d) = ac*10^n + (ad + bc)*10^{n/2} + bd$
  - $2018*1234 = (20*12* 10^4) + ((20*34+18*12) * 10^2) + (18*34)$
  - $= 2400000 + 89600 + 612$
  - $= 2490212$

Integer multiplication : Karatsuba

- (a+b)*(c+d) = ac + **<u>ad + bc</u>** + bd
- **<u>ad+bc</u>** = (a+b)*(c+d) – ac - bd

	$n/2$ digits	$n/2$ digits
A =	A_L	A_R
B =	B_L	B_R

$O(n^{1.59})$

$$A \times B = A_L \times B_L 10^n + (A_L \times B_R + A_R \times B_L) 10^{n/2} + A_R \times B_R$$

$$C_1 = A_L \times B_L$$

$$C_2 = A_R \times B_R$$

$$C_3 = (A_L + A_R) \times (B_L + B_R) = (\cancel{A_L B_L} + A_L B_R + A_R B_L + \cancel{A_R B_R})$$

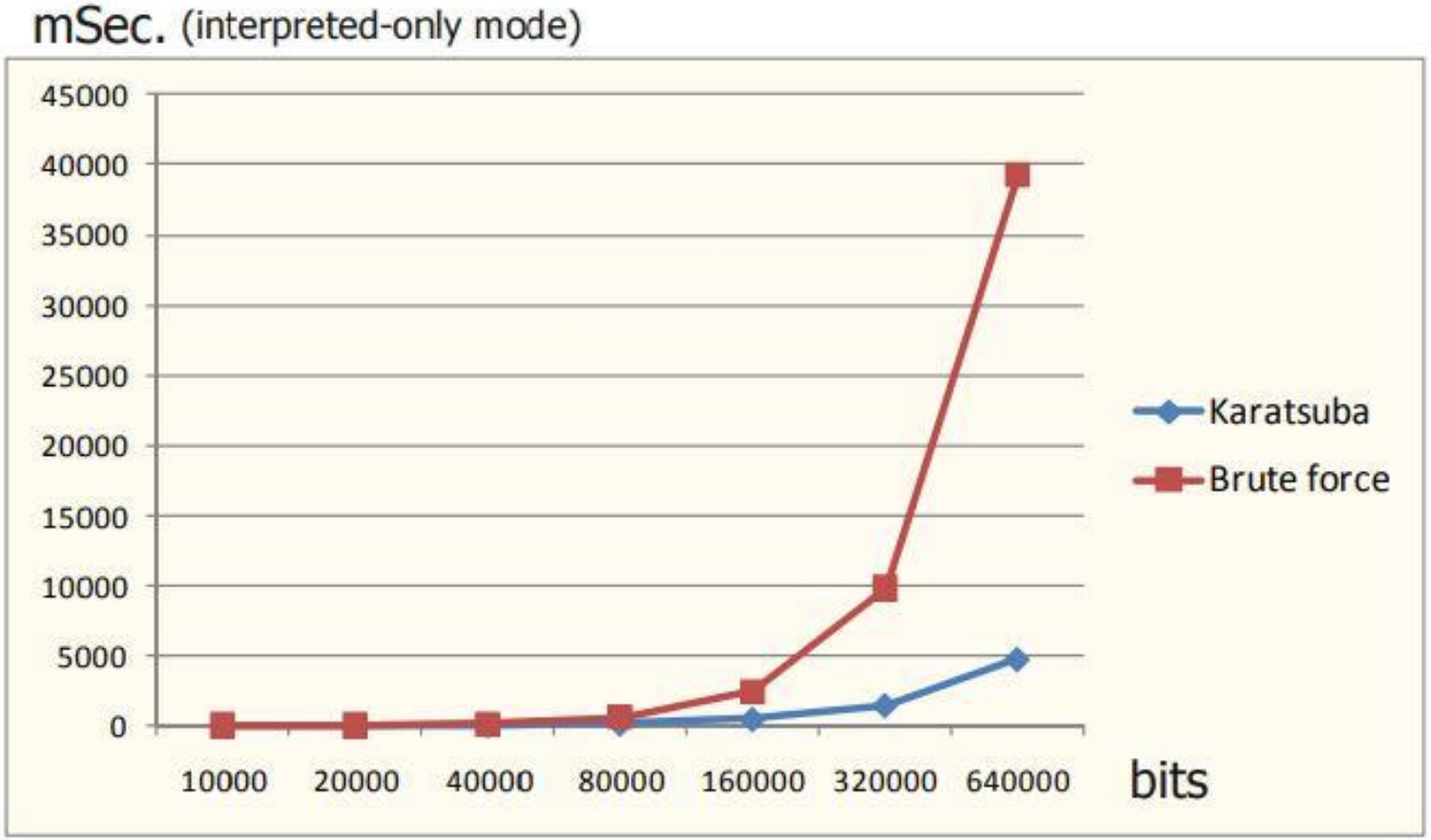
$$A \times B = C_1 10^n + (C_3 - C_1 - C_2) 10^{n/2} + C_2$$

Integer multiplication : Karatsuba

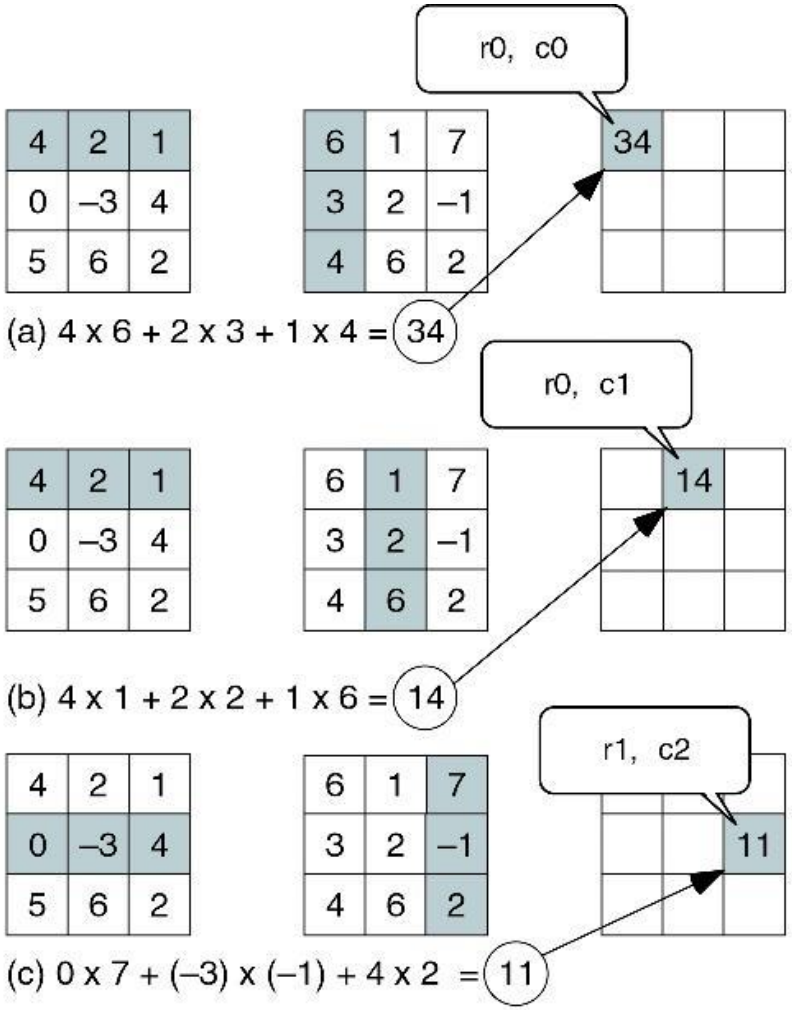
- 2018 * 1234
  - $2018 = (20 * 10^2) + 18 \Rightarrow (a*10^{n/2}+b)$
  - $1234 = (12 * 10^2) + 34 \Rightarrow (c*10^{n/2}+d)$
  - $(a*10^{n/2}+b) * (c*10^{n/2}+d) =$

    $$ac*10^n + ((a+b)*(c+d)-ac-bd)*10^{n/2} + bd$$
  - 2018*1234
  - $= (20*12* 10^4) + ((20+18)*(12+34)-(20*12)-(18*34)) * 10^2) + (18*34)$
  - $= 240*10^4 + (1748-240-612)*10^2 + 612$
  - $= 2490212$

Brute force vs. Karatsuba



Matrix multiplication



Matrix multiplication : Brute force

$O(n^3)$

```
Algorithm multiMatrix (matrix1, matrix2, size, matrix3)
Multiply matrix1 by matrix2 and place product in matrix3
   Pre  matrix1 and matrix2 have data
        size is number of columns and rows in matrix
   Post matrices multiplied--result in matrix3
1  loop (not end of row)
   1  loop (not end of column)
      1  loop (size of row times)
         1  calculate sum of
                (all row cells) * (all column cells)
         2  store sum in matrix3
   2  end loop
2  end loop
3  return
end multiMatrix
```

n = ความยาว

Matrix multiplication : Divide & Conquer

$$A = \left[\begin{array}{ccc|ccc} a_{1,1} & \cdots & a_{1,n/2} & \cdots & & a_{1,n} \\ \vdots & \boxed{A_{1,1}} & \vdots & & \boxed{A_{1,2}} & \vdots \\ a_{n/2,1} & \cdots & a_{n/2,n/2} & \cdots & & a_{n/2,n} \\ \hline \vdots & \boxed{A_{2,1}} & \vdots & & \boxed{A_{2,2}} & \vdots \\ a_{n,1} & \cdots & a_{n,n/2} & \cdots & & a_{n,n} \end{array}\right]$$

$O(n^3)$

$$\begin{bmatrix} C_{1,1} & C_{1,2} \\ C_{2,1} & C_{2,2} \end{bmatrix} = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \times \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix}$$

$$\begin{aligned} C_{1,1} &= A_{1,1} \times B_{1,1} + A_{1,2} \times B_{2,1} \\ C_{2,1} &= A_{2,1} \times B_{1,1} + A_{2,2} \times B_{2,1} \\ C_{1,2} &= A_{1,1} \times B_{1,2} + A_{1,2} \times B_{2,2} \\ C_{2,2} &= A_{2,1} \times B_{1,2} + A_{2,2} \times B_{2,2} \end{aligned}$$

Matrix multiplication :

Divide & Conquer - Strassen

$$\begin{bmatrix} C_{1,1} & C_{1,2} \\ C_{2,1} & C_{2,2} \end{bmatrix} = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \times \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix}$$

$$\begin{aligned}
M_1 &= (A_{12} - A_{22}) \times (B_{21} + B_{22}) \\
M_2 &= (A_{11} + A_{22}) \times (B_{11} + B_{22}) \\
M_3 &= (A_{11} - A_{21}) \times (B_{11} + B_{12}) \\
M_4 &= (A_{11} + A_{12}) \times B_{22} \\
M_5 &= A_{11} \times (B_{12} - B_{22}) \\
M_6 &= A_{22} \times (B_{21} - B_{11}) \\
M_7 &= (A_{21} + A_{22}) \times B_{11}
\end{aligned}$$

$$\begin{aligned}
C_{11} &= M_1 + M_2 - M_4 + M_6 \\
C_{12} &= M_4 + M_5 \\
C_{21} &= M_8 + M_7 \\
C_{22} &= M_2 - M_3 + M_5 - M_7
\end{aligned}$$

1969 : $O(n^{2.81})$ Strassen

1987 : $O(n^{2.376})$ Coppersmith–Winograd

Quiz

- จงแสดงขั้นตอนการหาแก้ปัญหาดังนี้ โดยใช้แนวคิดแบบ Divide and Conquer
  - 2^{1000} mod 10
  - 3^{600} mod 5
- Binary Search จัดเป็น Divide and conquer algorithm หรือไม่ เพราะเหตุใด
- จงกำหนด base case และ recursive case ของอัลกอริทึม binary search