5800

L2 4102

Jan 21 2016

Karatsuba, Recurrences

shelat

warmup

Simplify $(1 + a + a^2 + \cdots + a^L)\underline{(a - 1)} =$ $a^{L+1} - 1$

$$\begin{array}{l} \cancel{a} + \cancel{a^2} + \cancel{a^3} + \cancel{a^4} + \cdots + a^{L+1} \\ -1 - \cancel{a} - \cancel{a^2} - \cancel{a^3} - \cancel{a^4} + \cdots - \cancel{a^L} \\ \hline \end{array}$$

$$\underline{1 + a + a^2 + \cdots a^L} = \left[\frac{a^{L+1} - 1}{a - 1}\right]$$

$$\boxed{\sum_{i=0}^{L} a^i = \left(\frac{a^{L+1} - 1}{a - 1}\right)}$$

warmup

$$\sum_{i=0}^{L} a^i = \frac{a^{L+1} - 1}{a - 1}$$

Logarithm

$$\underline{2^{\log_2(n)} = n}$$

$\log_2(n) =$ the value y such that $2^y = n$

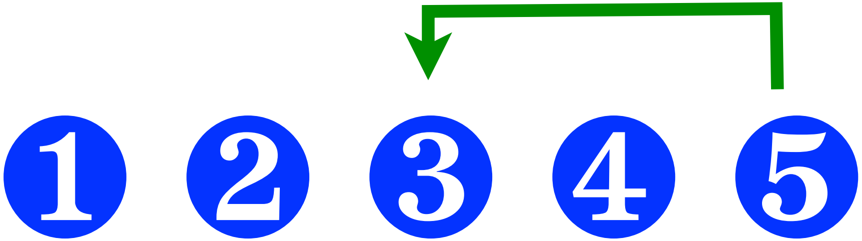
base 2

$$\log_2(16) = 4 \qquad 2^4 = 16$$

$$\log_2(20) \approx 4.3219\ldots$$

$$= \frac{\log_{10}(20)}{\log_{10}(2)} \quad \Leftarrow$$

Recall from last time…

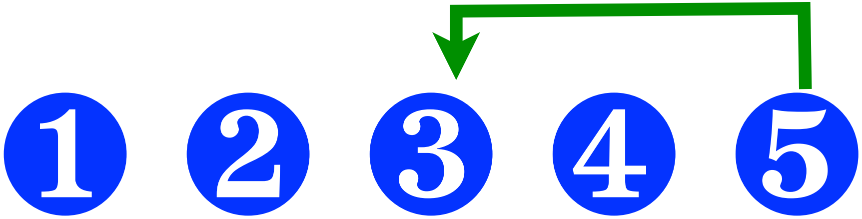


stand **set** **greet** **sit/add** **repeat**

| | || |

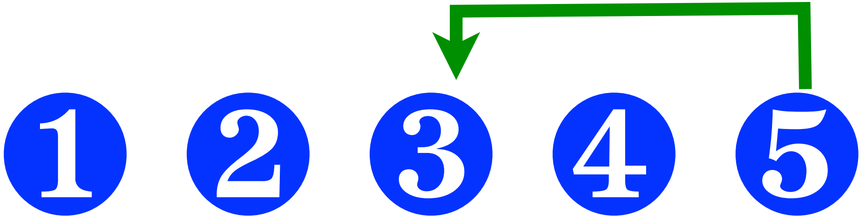
Simple case: 2 people

$$T(2) = 5$$

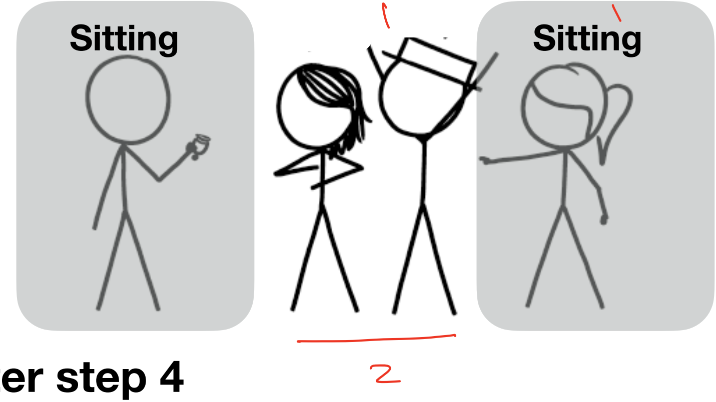


stand set greet sit/add repeat

$$T(4) =$$



stand **set** **greet** **sit/add** **repeat**



After step 4

$$T(4) \leq 4 + T(2)$$

~~① stand~~ ~~② set~~

These steps only happen once.

I1:Approx is OK

③ **greet** ④ **sit/add** ⑤ **repeat**

What about these?

T(n) just counts this part

① stand ② set

③ greet ④ sit/add ⑤ repeat

how fast does it work:

round up

$$T(n) = 1 + 1 + T(\lceil n/2 \rceil)$$

$T(1) = 3$

① stand ② set ③ greet ④ sit/add ⑤ repeat

how fast does it work:

$$T(n) = 1 + 1 + T(\lceil n/2 \rceil)$$

$$T(1) = 3$$

This is a recurrence

$$T(n) = T(\lceil n/2 \rceil) + 2$$

$$T(1) = 3$$

solve a simpler case when n is a power of 2.

$$T(\underline{2}^k) = 2 + \underbrace{T(2^{k-1})}$$

$$= 2 + \left(2 + T(2^{\underline{k-2}})\right)$$

$$= 2 + 2 + 2 + T(2^{k\underline{-3}})$$

$$\vdots$$

$$= \overbrace{2 + \cdots + 2}^{k} + T(2^0)$$

$$= 2k + T(1) = 2k + 3$$

$$T(2^k) = 2 + T(2^{k-1})$$

$$T(2^k) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$T(2^k) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$= \overbrace{2 + 2 + \cdots + 2}^{k} + T(2^0)$$

$$T(2^{\underline{k}}) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$= \overbrace{2 + 2 + \cdots + 2}^{k} + T(2^0)$$

$$= \underline{2\underline{k} + T(1)}$$

intuition here

$$T(2^k) = 2 + T(2^{k-1})$$

$2^{(\log n)} = n$

$\underline{2^{\lceil \log_2 n \rceil} \geqslant n}$

$n = 20$

$\lceil \log_2 20 \rceil = 5$

what if n is not a power of 2 ??

Other cases?

① $\underline{T(n) \leq T(m) \text{ if } m \geqslant n}$

$\underline{T(n)} \leq T(\underbrace{2^{\lceil \log_2 n \rceil}}_{m})$

intuition here

$$T(2^k) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$T(n) = O(\log_2 n)$$

$$= O(\log n)$$

Other cases?

intuition here

$$T(2^k) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$= \overbrace{2 + 2 + \cdots + 2}^{k} + T(2^0)$$

Other cases?

intuition here

$$T(2^k) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$= \overbrace{2 + 2 + \cdots + 2}^{k} + T(2^0)$$

$$= 2k + T(1)$$

Other cases?

Idea1: It is OK to approximate

A good way to do this is to ignore low order terms of our functions, i.e. using asymptotic notation for our functions.

Asymptotic notation

big-Oh

$O(g)$ This notation represents a set

{ set of functions }

the set { functions f such that there exist ∃ constants $c, n' > 0$ such that

$f(n) \leq c \cdot g(n)$ for $n > n'$ }

$\forall\, n > n'$

Asymptotic notation

$$O(g)$$

Set of functions that are at most within constant of g for large n

Asymptotic notation

Big Oh

$O(g)$ Set of functions that are at *most* within const of g for large n

Omega (g)

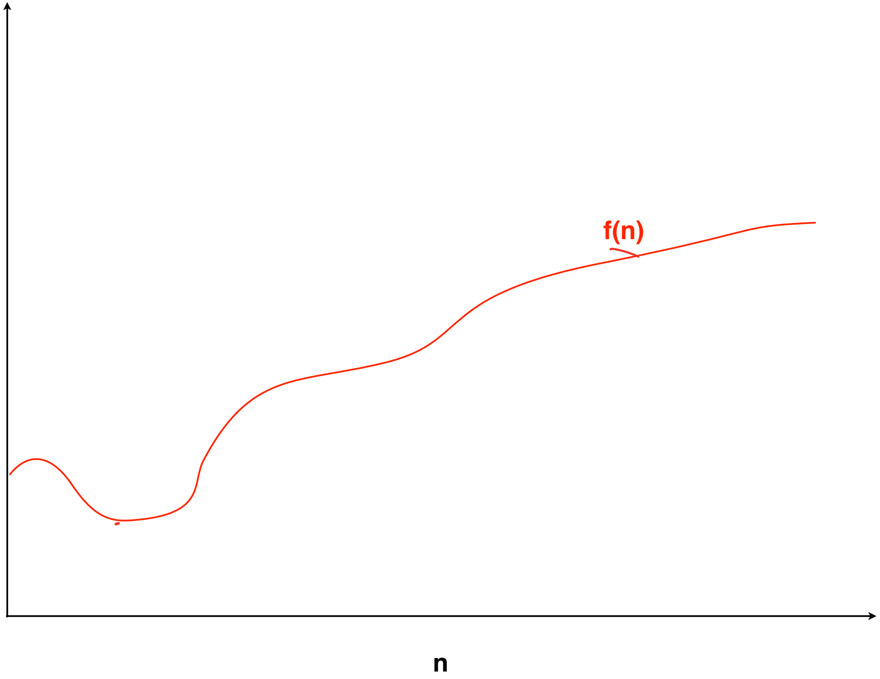
$\Omega(g)$ Set of functions that are at *least* within const of g for large n

Theta (g)

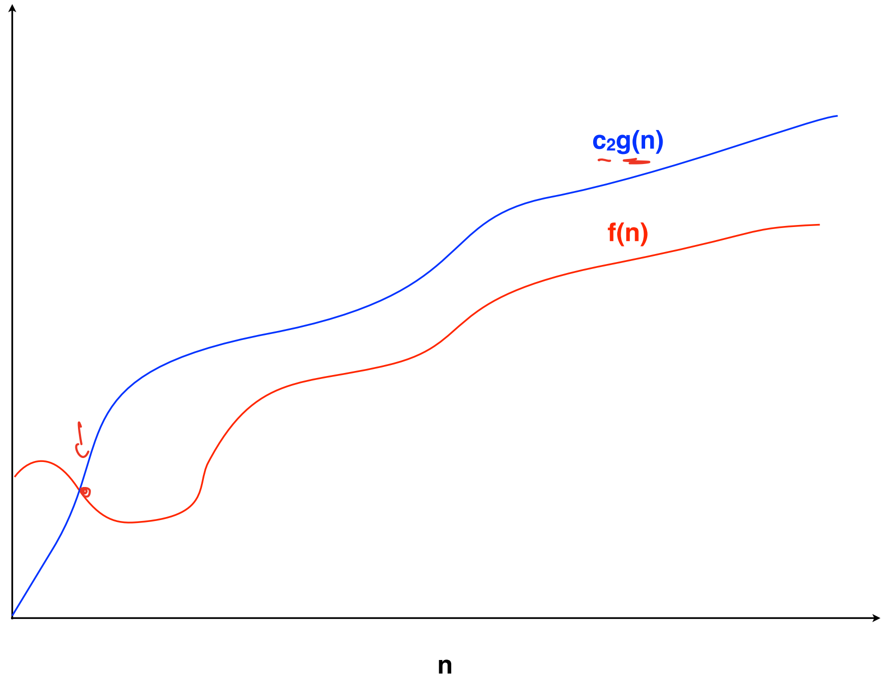
$\Theta(g)$ Set of functions that are at within const of g for large n

combo of O and Ω

Omega sandwich

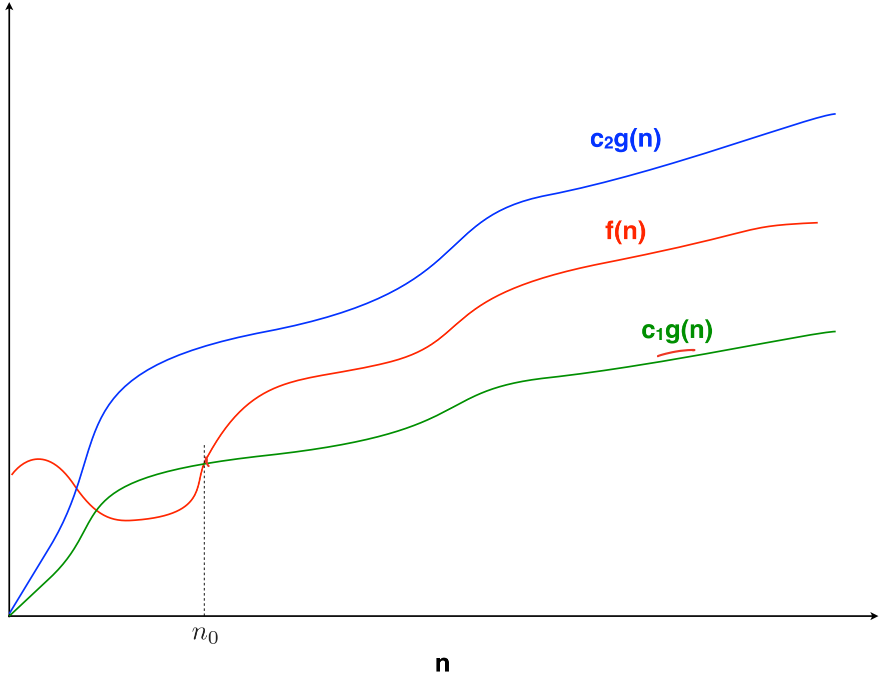


Omega sandwich



$$f(n) = O(g(n))$$

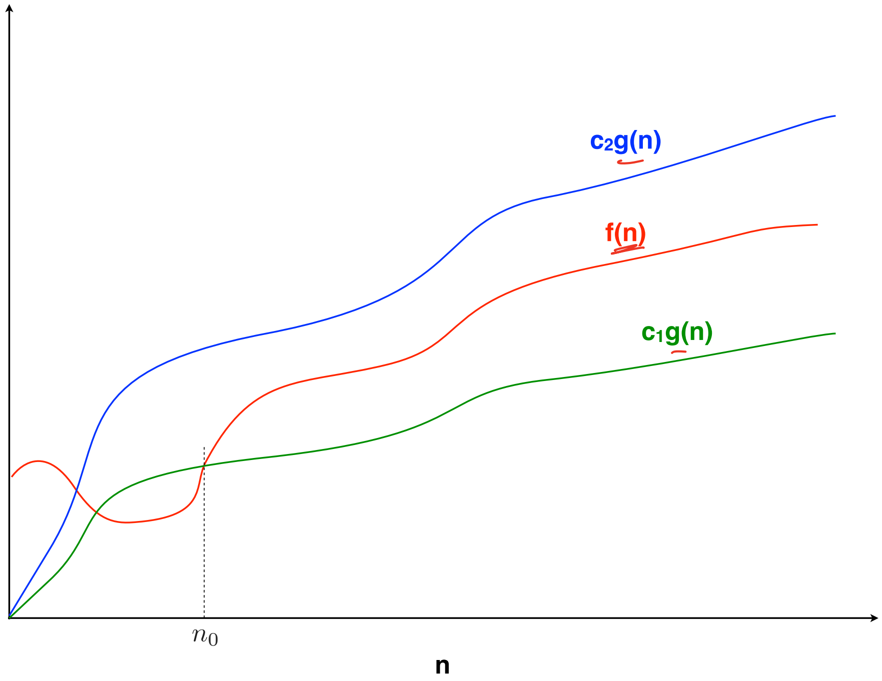
Omega sandwich



$$f(n) = O(g(n))$$

$$f(n) = \Omega(g(n))$$

Omega sandwich



$$f(n) = O(g(n))$$

$$f(n) = \Theta(\underline{g(n)})$$

$$f(n) = \underline{\Omega(g(n)}))$$

Examples of asymptotic notation

this is a function

$$3n \underset{\in}{\overset{?}{=}} \underline{O(n)}$$

this is a set

$$n = \underline{O(2^n)}$$

"n is upper-bounded by 2^n"

$$n = \Theta(n), \qquad \underline{3n = \Theta(n)}$$

$$10{,}000\,n = \Theta(n)$$

$$\Theta(n) = \{\, n,\ 3n,\ 10000n,\ \dots \,\}$$

intuition here

✓ ① $$T(2^k) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$= \overbrace{2 + 2 + \cdots + 2}^{k} + T(2^0)$$

$$= 2k + T(1) = \underline{O(\log(2^k))}$$

✓ ② $T(n) \le T(m)$ if $m > n$

③ $T(n) \le T(2^{\lceil \log n \rceil}) = O(\log n)$

③' $T(n) \ge T(2^{\lfloor \log n \rfloor})$

floor
round down

goal:
what we want to show.

$T(n) = \Theta(\log n)$

what else is needed to show our goal

we need to argue $T(n) = \Omega(\log n)$

intuition here

$$T(2^k) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$= \overbrace{2 + 2 + \cdots + 2}^{k} + T(2^0)$$

$$= 2k + T(1) \;= O(\log(2^k))$$

$$\forall 0 < n < m, T(n) \leq T(m)$$

$$T(m) \leq T(2^{\lceil \log(m) \rceil}) = 2\lceil \log(m) \rceil + 2$$

intuition here

$$T(2^k) = 2 + T(2^{k-1})$$

$$= 2 + 2 + T(2^{k-2})$$

$$= \overbrace{2 + 2 + \cdots + 2}^{k} + T(2^0)$$

$$= 2k + T(1) \;= O(\log(2^k))$$

$$\forall 0 < n < m, T(n) \leq T(m)$$

$$T(m) \leq T(2^{\lceil \log(m) \rceil}) = 2\lceil \log(m) \rceil + 2$$

$$T(m) = \Omega(\log(m))$$

$$= \Theta(\log(m))$$

main ideas:

① Solve big by reducing to smaller problem

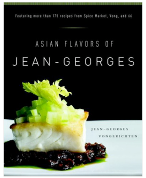
② Approximate well

③ Use asymptotic notation to capture the performance

How to solve recurrence relations

tree method

?-✓ – guess & check method
(induction)

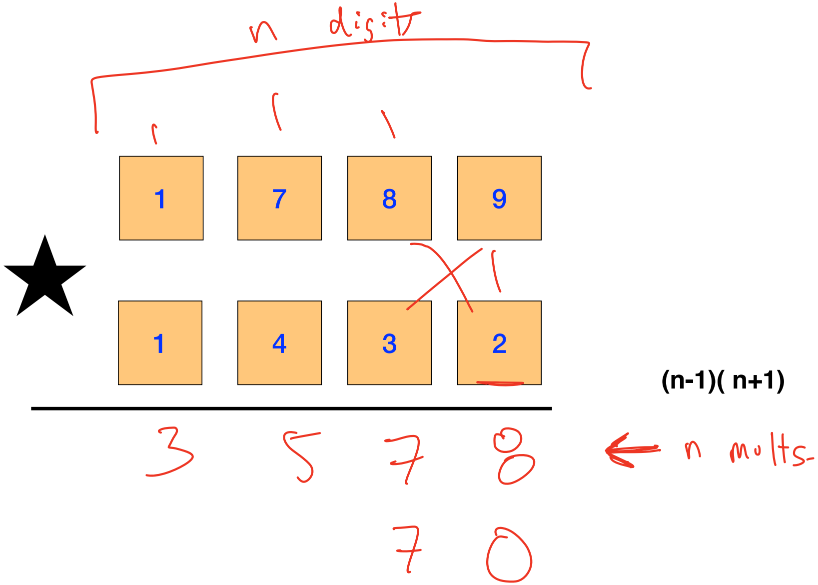


– Masters' theorem.

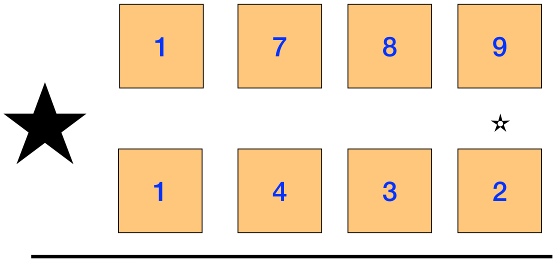
http://www.drblank.com/law301.jpg

– substitutions

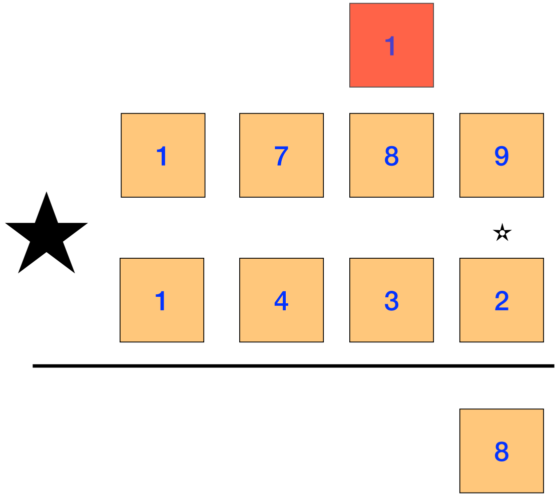
Multiplication



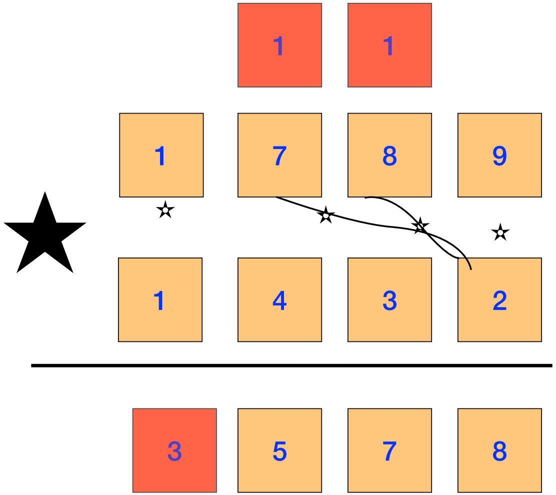
(n-1)(n+1) +



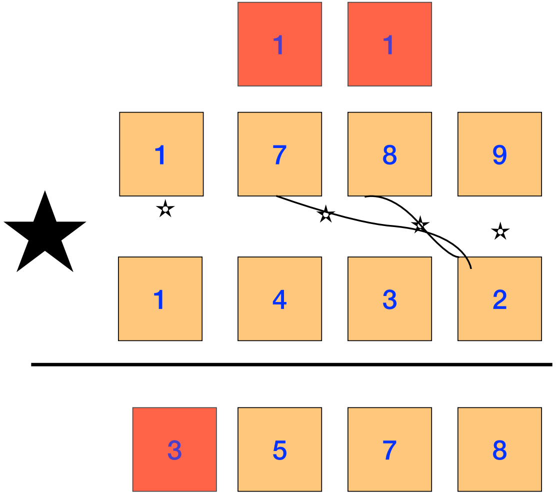
(n-1)(n+1) ✛



(n-1)(n+1) ✛

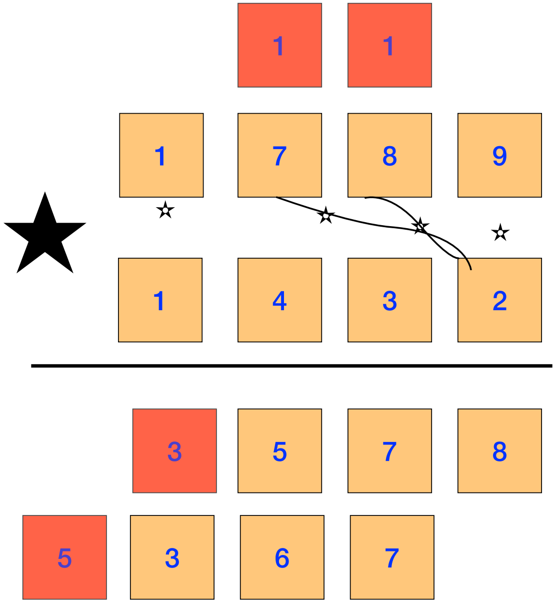


(n-1)(n+1) ✢



(n-1)(n+1) ✢

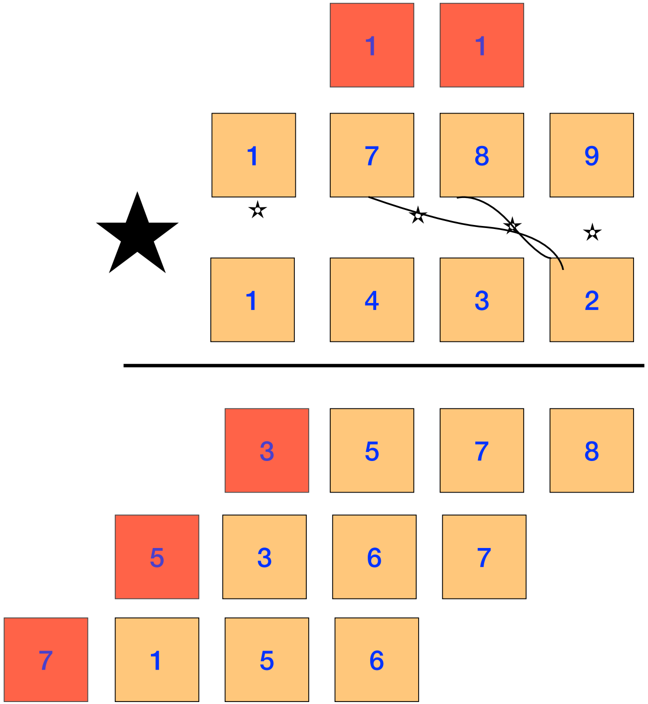
n ☆ n-1 ✢



(n-1)(n+1) ✢

n ☆ $\underline{n-1}$ ✢

$\underline{n}$ ☆ $\underline{n-1}$ ✢

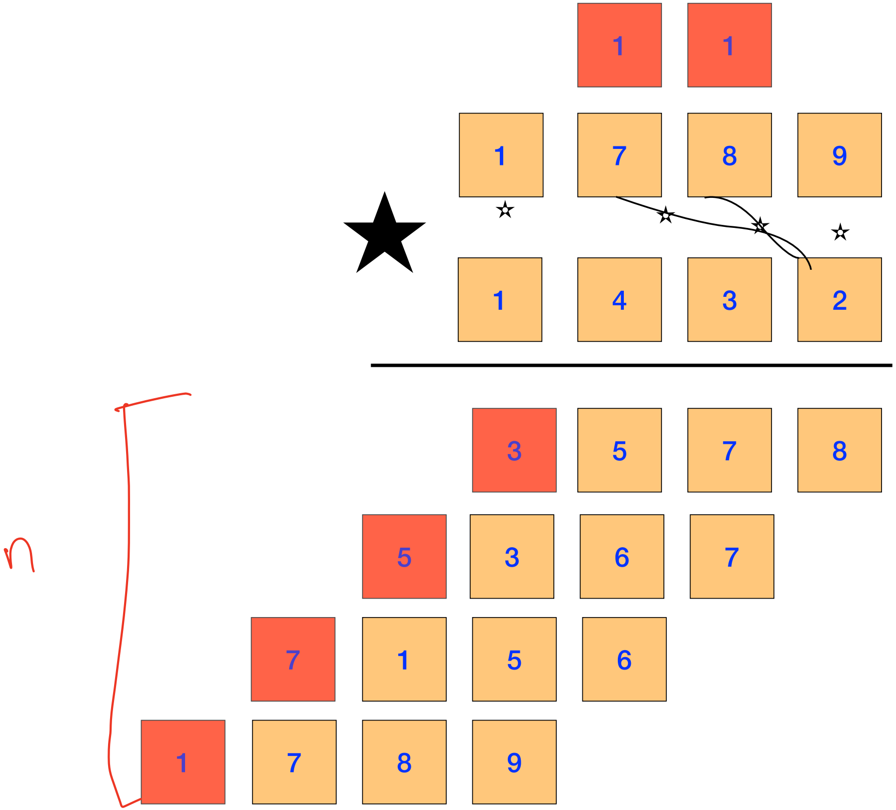


(n-1)(n+1) ✢

n ☆ n-1 ✢

n ☆ n-1 ✢

n ☆ n-1 ✢



$\Theta(n^2)$

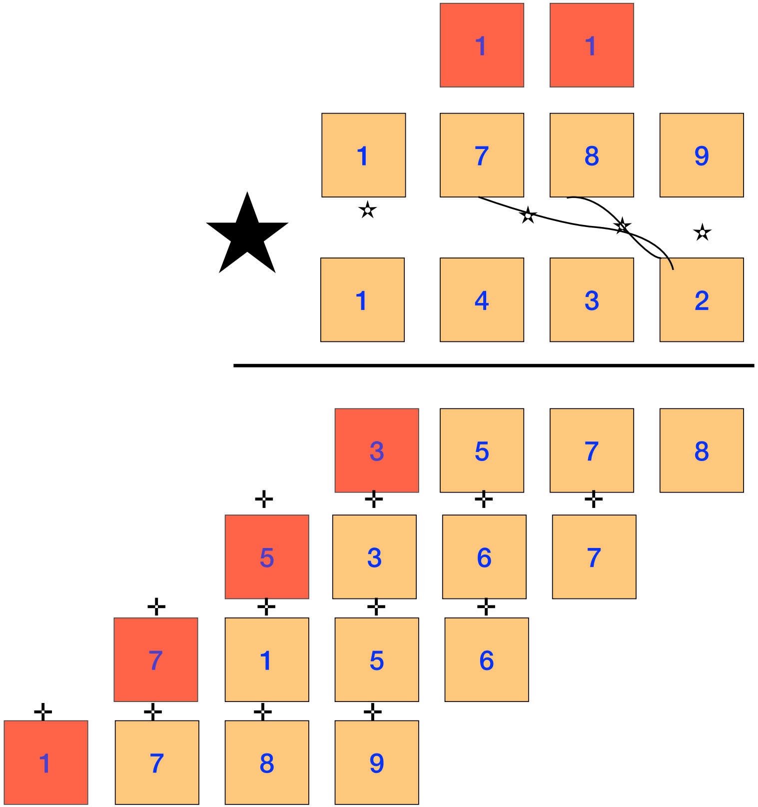
(n-1)(n+1) +

n ☆ n-1 +

n ☆ n-1 +

n ☆ n-1 +

n ☆ n-1 +



$2n^2 > n^2 - n > \left(\frac{1}{2}\right) n^2$

for

$n > 2$

(n-1)(n+1) ✢

n ☆ **n-1** ✢

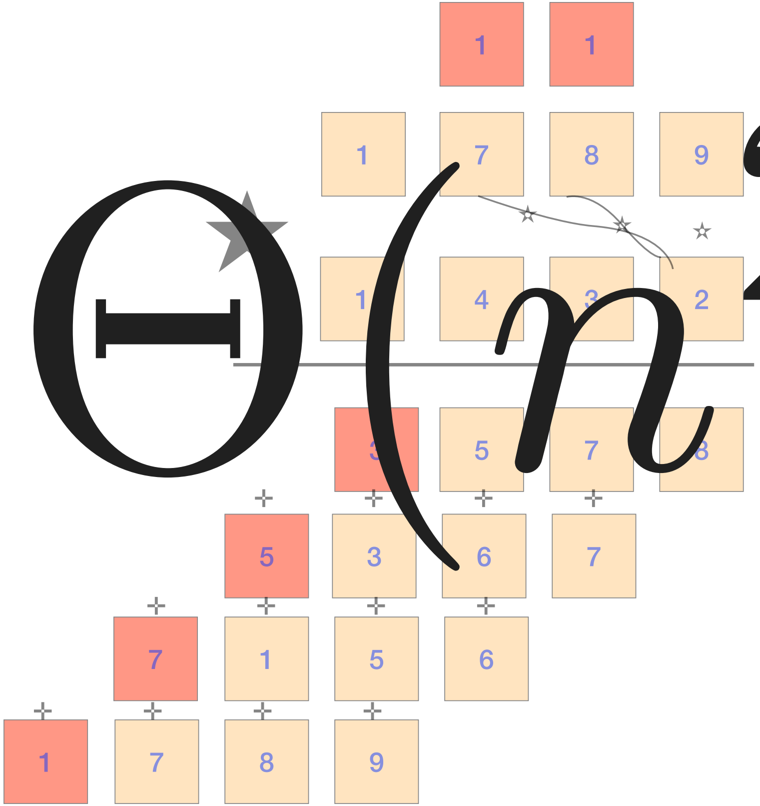
n ☆ **n-1** ✢

n ☆ **n-1** ✢

n ☆ **n-1** ✢

$\underline{\underline{n^2 - n}}$

$$\Theta(n^2)$$



(n-1)(n+1) ✢

n ☆ n-1 ✢

n ☆ n-1 ✢

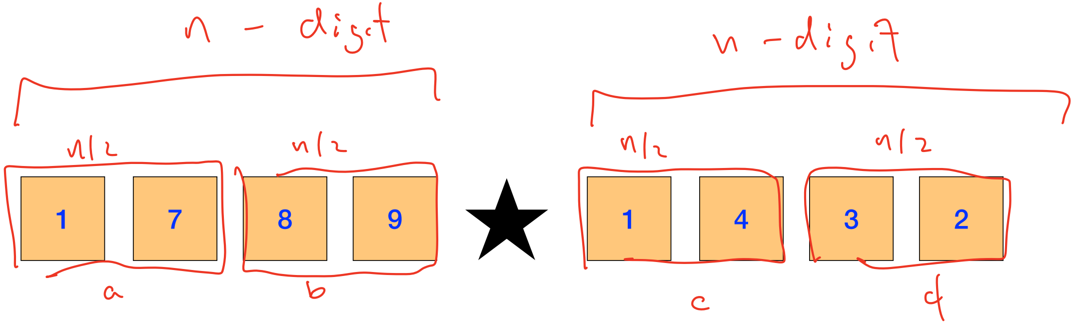
n ☆ n-1 ✢

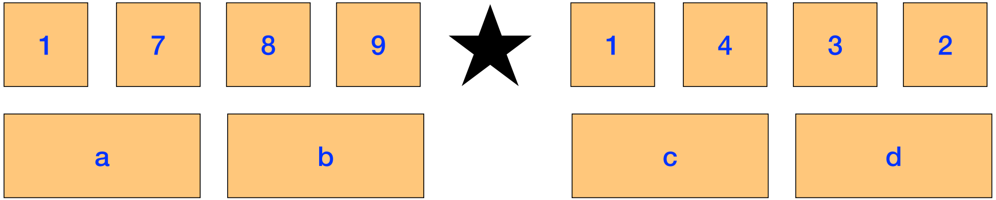
n ☆ n-1 ✢

Lets try to do better by using fewer unit operations

Theme 1

Solve big problem by turning into smaller one -


n - digit
n/2
n/2
1
7
8
9
a
b
n - digit
n/2
n/2
1
4
3
2
c
d

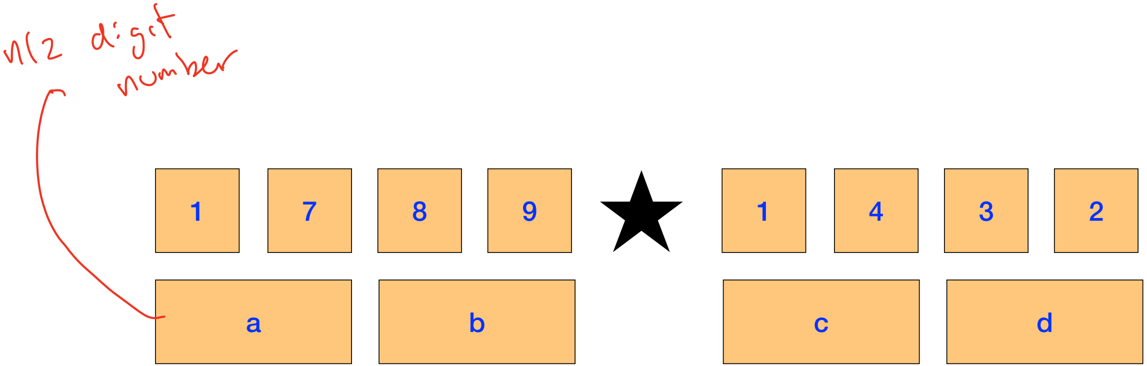


1 7 8 9 ★ 1 4 3 2

a b c d

$(a\cdot 100 \quad + \quad b)$ $\qquad$ $(c\cdot 100) \quad + \quad d$

$$a\cdot c\cdot(100)^2 + \quad (a\cdot d + b\cdot c)\,100$$



n/2 digit number

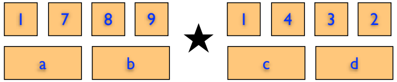
1 7 8 9 ★ 1 4 3 2

a b c d

$$ac100^2 + (ad + bc)100 + bd$$

n-digit inputs

Mult(ab, cd)



$$ac100^2 + (ad + bc)100 + bd$$

Base case: return b*d if inputs are 1-digit

ac = Mult(a, c)

ad = Mult(a, d)

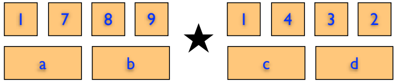
bc = Mult(b, c)

bd Mult(b, d)

Return $ac \cdot 100^2 + (ad + bc) 100 + bd$

n-digit inputs

Mult(ab, cd)



$$ac100^2 + (ad + bc)100 + bd$$

$T(n)$ = # unit operation to compute MULT on n-digit inputs

Base case: return b*d if inputs are 1-digit

Else:

Compute x = Mult(a,c) ← $T(n/2)$ (a and c each n/2)

Compute y = Mult(a,d) "

Compute z = Mult(b,c) "

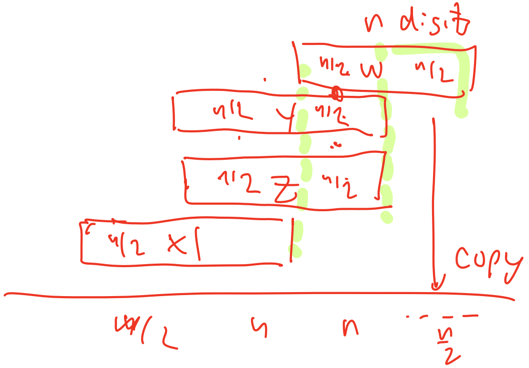
Compute w = Mult(b,d) "

< n digits

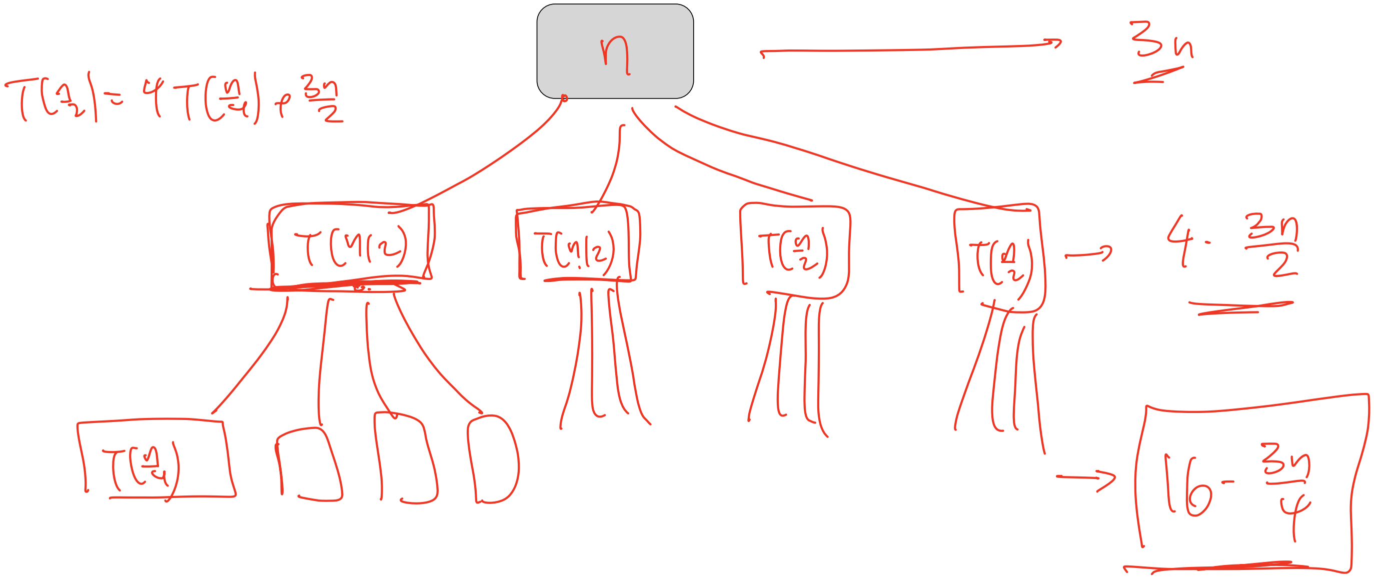
→Return r = x*10^n + (y+z)$10^{n/2}$ + w ←

$T(1) = 1$

$T(n) = 4T(n/2) + 3n$

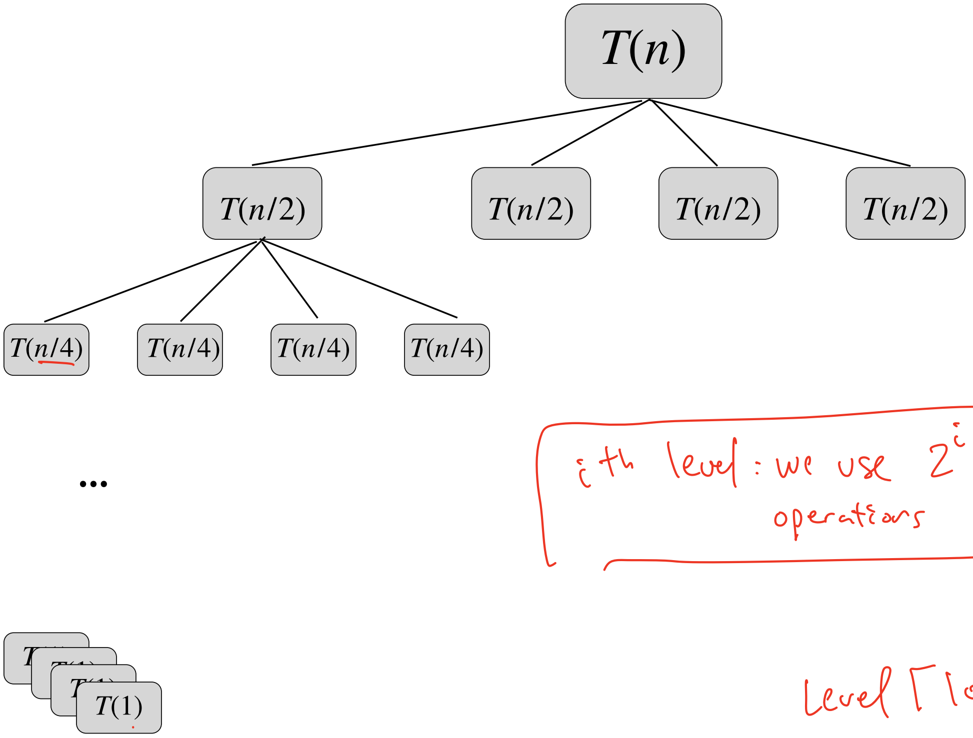


$$T(n) = 4T(\lceil n/2 \rceil) + 3n$$



$T(\frac{n}{2}) = 4T(\frac{n}{4}) + \frac{3n}{2}$

$$T(n) = 4T(\lceil n/2 \rceil) + 3n$$



L2

ith level : we use $2^i \cdot 3n$ operations

Level $\lceil \log n \rceil$

$2^{\log n}$

$4^0 \, 3n$

$4^1 \cdot (3n/2) = 2 \cdot 3n$

$4^2 = 16 \cdot (3n/4) = 2^2 \cdot 3n$

$64 \cdot (3n/8) = 2^3 \cdot 3n$

$\frac{3n}{2^{\lceil \log n \rceil}}$

$\cancel{2}4^{\lceil \log(n) \rceil} \cdot (\cancel{1}3)$

calculations:

$$T(n) = \text{total \# of unit operations} =$$

$$= 3n + 2\cdot 3n + 2^2\cdot 3n + \cdots \underline{2^{\lceil \log n \rceil}\cdot 3n}$$

$$= 3n\left[1 + 2 + 2^2 + \cdots + 2^{\lceil \log n \rceil}\right]$$

$$= 3n\cdot\left[\frac{2^{\lceil \log n \rceil + 1} - 1}{\cancel{2-1}}\right]$$

$$2^{\lceil \log 33 \rceil} = \underline{64}$$

$$\leq 3n\cdot(2n\cdot 2) - 3n \overset{n}{=} 12n^2 - 3n = O(n^2)$$

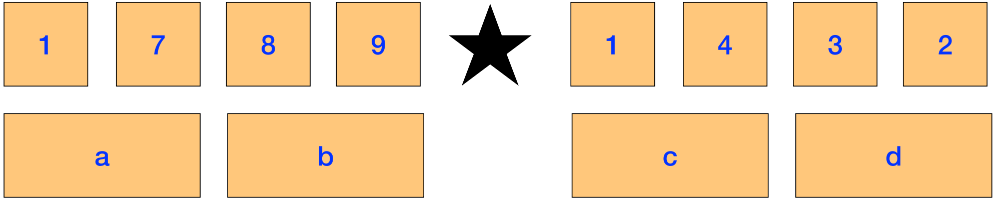
$$\Rightarrow \Theta(n^2)$$

$$\geq 3n\cdot n\cdot 2 - 3n = 6n^2 - 3n = \Omega(n^2)$$

How can we improve?

$$4^{\lceil \log n \rceil} = \underline{2^{\lceil \log n \rceil}} \cdot \cancel{2^{\lceil \log n \rceil}} \cdot \left(\frac{\underline{\underline{3n}}}{\cancel{2^{\lceil \log n \rceil}}} \right)$$

Karatsuba Algorithm



1 7 8 9 ★ 1 4 3 2

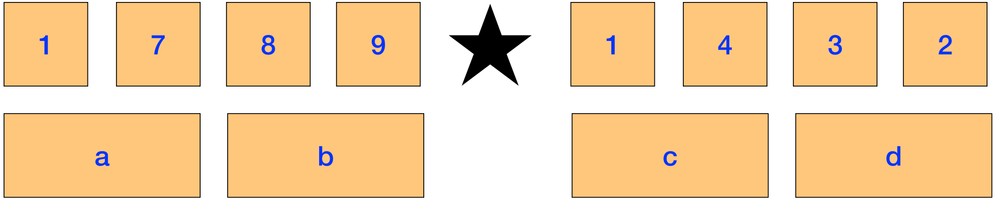
a b c d

$$\underline{ac}100^2 + (\underline{ad} + \underline{bc})100 + \underline{bd}$$

$$(a+b)(c+d) = ac + (ad+bc) + bd$$

$$\Rightarrow (ad+bc) = (a+b)(c+d) - ac - bd$$

Karatsuba Algorithm

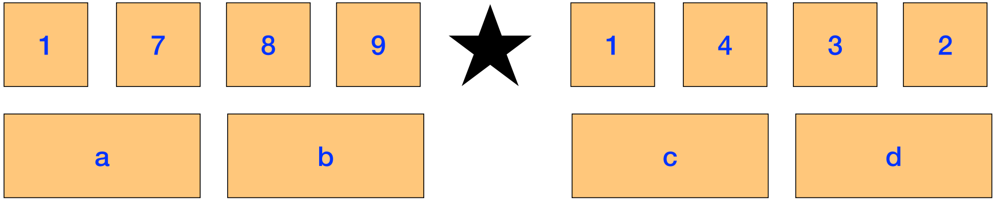


$$ac100^2 + (ad + bc)100 + bd$$

$$(a + b)(c + d) = ac + ad + bc + bd$$

$$ad + bc = (a + b)(c + d) - ac - bd$$

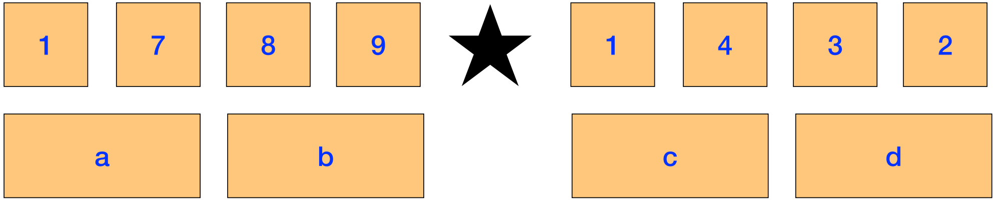
Karatsuba Algorithm



1 7 8 9 ★ 1 4 3 2

a b c d

Karatsuba Algorithm



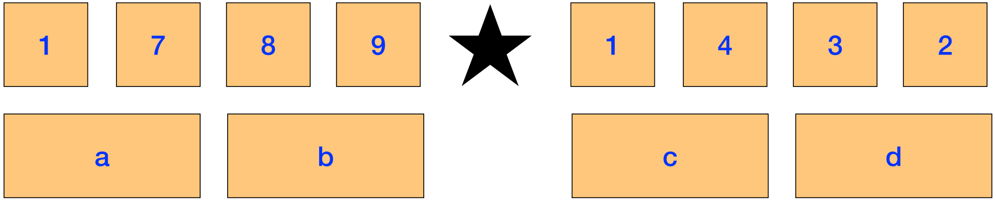
1 7 8 9 ★ 1 4 3 2

a b c d

Recursively compute

① $ac, bd, (a+b)(c+d)$

Karatsuba Algorithm

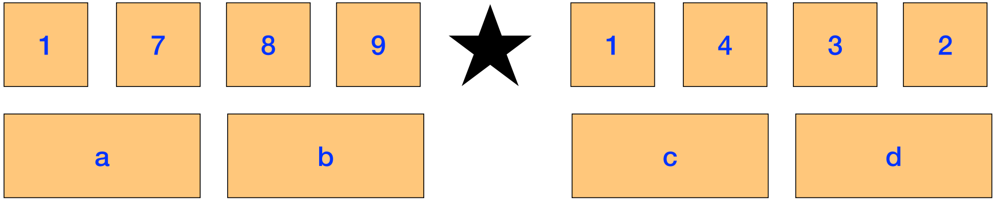


Recursively compute

❶ $$ac, bd, (a+b)(c+d)$$

❷ $$ad + bc = \underline{(a+b)(c+d)} - \underline{ac} - \underline{bd}$$

Karatsuba Algorithm



Recursively compute

❶ $ac, bd, (a+b)(c+d)$

❷ $ad + bc = (a+b)(c+d) - ac - bd$

❸ $\underline{ac}100^2 + (\underline{ad + bc})100 + \underline{bd}$

Karatsuba(ab, cd)

Base case: return b*d if inputs are 1-digit ✓ $T(1) = 1$

ac = Karatsuba(a,c) | $T(\frac{n}{2})$

bd = Karatsuba(b,d) | $T(n/2)$

$n/2 + n/2 = n$

extra operation

t = Karatsuba((a+b),(c+d)) | $T(\frac{n}{2}+1) \sim T(\frac{n}{2})$

cheating, but ok.

mid = t - $\overset{n}{\text{ac}}$ - bd n - digit

2n extra operations

RETURN ac*100^2 + mid*100 + bd

$$T(n) = 3T(\frac{n}{2}) + 6n$$

Karatsuba(ab, cd)

Base case: return b*d if inputs are 1-digit

ac = Karatsuba(a,c)

bd = Karatsuba(b,d)

t = Karatsuba((a+b),(c+d)) n

mid = $\underline{t - ac - bd}$ 2n

RETURN $\underline{ac*100^2 + mid*100 + bd}$

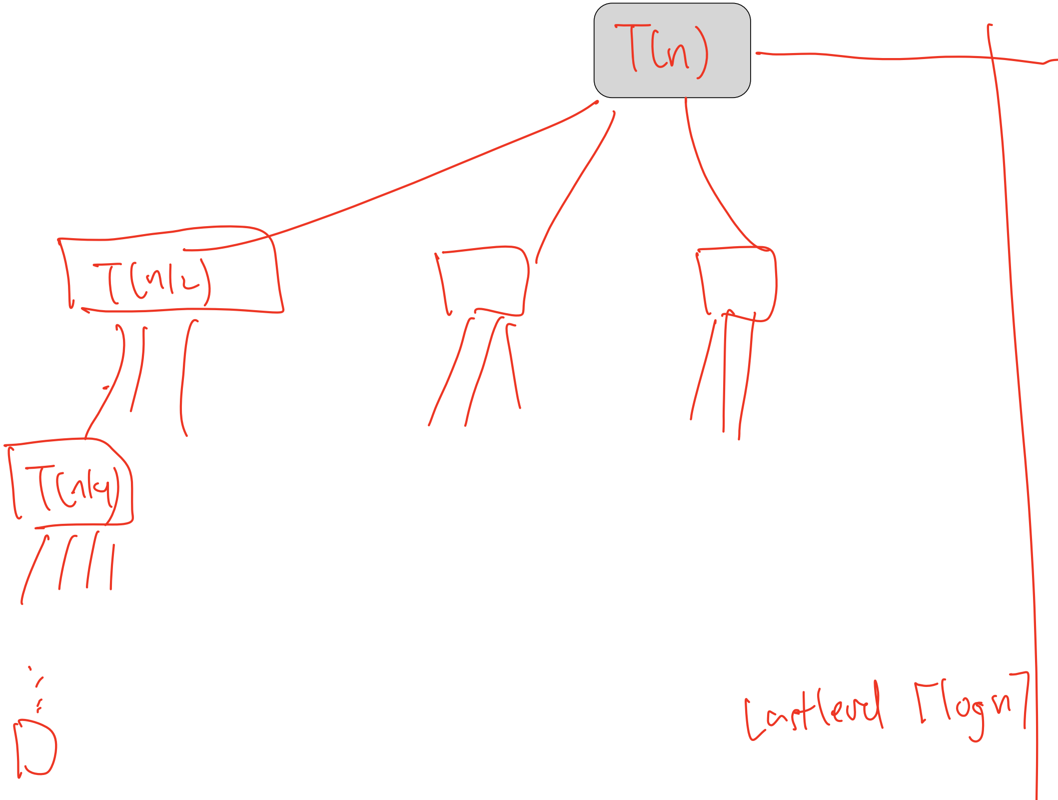
$3T(n/2) + \underline{2n}$

Ignoring issue of carries

$\underline{4n}$

$\underline{\cancel{4n}}$ $\underline{3n}$

$$T(n) = \underline{3}T(n/2) + \underline{9n}$$



$$9n$$

$$3 \cdot \frac{9n}{2}$$

$$3^2 \cdot \frac{9n}{4}$$

$$3^3 \frac{9n}{2^3}$$

$$3^{\lceil \log n \rceil} \cdot \frac{9n}{2^{\lceil \log n \rceil}}$$

calculations:

$$T(n) = 9n + 3 \cdot \frac{9n}{2} + 3^2 \cdot \frac{9n}{2^2} + \dots + 3^{\lceil \log n \rceil} \cdot \frac{9n}{2^{\lceil \log n \rceil}}$$

$$= 9n \left[1 + \frac{3}{2} + \left(\frac{3}{2}\right)^2 + \dots \left(\frac{3}{2}\right)^{\lceil \log n \rceil} \right]$$

$$= 9n \left[\frac{\left(\frac{3}{2}\right)^{\lceil \log n \rceil + 1} - 1}{\left(\frac{3}{2}\right) - 1} \right] = 9n \cdot \cancel{2} \cdot \frac{3}{\cancel{2}} \cdot \left(\frac{3}{2}\right)^{\lceil \log n \rceil} - 18n$$

$$= 27\cancel{n} \cdot \frac{2^{(\log_2 3)\lceil \log n \rceil}}{\cancel{2^{\lceil \log n \rceil}}} - 18n = 27 \cdot n^{\log_2 3} - 18n$$

$$= O\left(n^{\log_2 3}\right)$$

calculations:

$$T(n) = 9n + \left(\frac{3}{2}\right) \cdot 9n + \left(\frac{3}{2}\right)^2 \cdot 9n + \cdots + \left(\frac{3}{2}\right)^{\lceil \log_2 n \rceil} \cdot 9n$$

$$= 9n\left[1 + \left(\frac{3}{2}\right) + \left(\frac{3}{2}\right)^2 + \cdots + \left(\frac{3}{2}\right)^{\log_2 n}\right] = 9n\left[\frac{\left(\frac{3}{2}\right)^{\lceil \log_2 n \rceil + 1} - 1}{\underline{\left(\frac{3}{2}\right) - 1}}\right]$$

$$= (9n)(2)\left[\left(\frac{3}{2}\right)^{\log_2 n + 1} - 1\right]$$

$$3 = 2^{\log_2 3}$$

$$= (9n)(2)\left(\frac{3}{2}\right)\left[\frac{3^{\log_2 n}}{\underbrace{\boxed{2^{\log_2 n}}}_{n}}\right] - 18n$$

$$= 27 \cdot \underbrace{3^{\log_2 n}}_{\text{aka}} - 18n = 27 \cdot n^{\log_2 3} - 18n = O\left(n^{\log_2 3}\right)$$

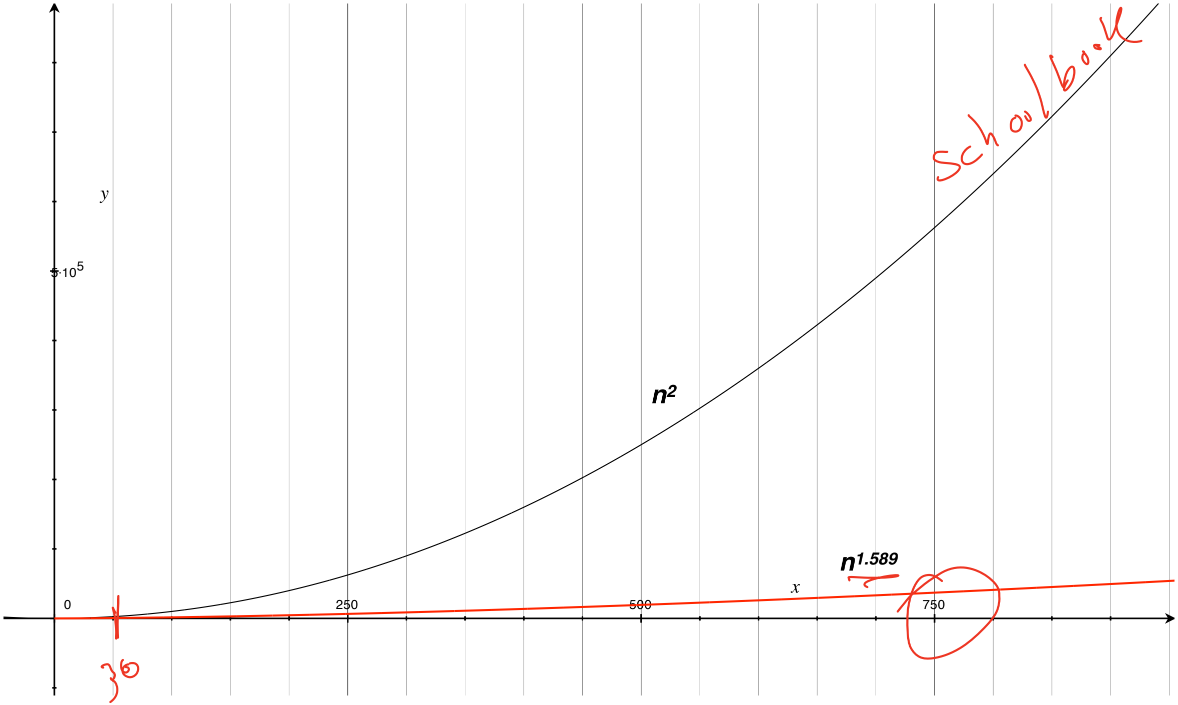
$$= \left(2^{\log_2 3}\right)^{\log_2 n} = \left(2^{\log_2 n}\right)^{\log_2 3} = \left(n^{\log_2 3}\right)$$

$$T(n) = 3T(n/2) + 9n$$

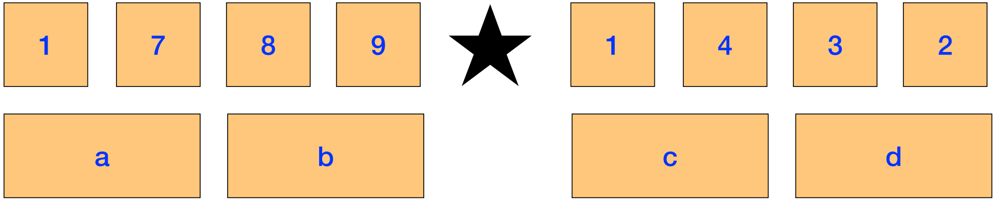
$$O(n^{\log_2(3)})$$

$$T(n) = 3T(n/2) + 9n$$

$$O(n^{\log_2(3)}) \quad O(n^{1.589})$$



y
5·10^5
0
250
500
750
x
n^2
n^1.589
Schoolbook
30



1 7 8 9 ★ 1 4 3 2

a b c d

$$T(n) = 3T(n/2) + 9n$$

$$T(n) = 4T(n/2) + 3n$$

simpler proof technique?

❶

induction redux

classic goal: prove that some property P(k) is true for all k

$\forall k, P(k)$ holds

❶

one long proof...

classic
goal: prove that some property P(k) is true for all k

$\forall k, P(k)$ holds

❶ Induction

classic
base case: $$P(1)$$ is true.

classic
inductive step: $$\left.\begin{matrix}P(1)\\ \cdots\\ P(k)\end{matrix}\right\}$$ implies $$P(k+1)$$ is true

❶ Induction, asymptotic style

classic
base case: $P(n^*)$ is true.

classic
inductive step: $\left.\begin{array}{c} P(n^*) \\ \cdots \\ P(k) \end{array}\right\}$ implies $P(k+1)$ is true

simpler proof **(guess +chk)**

$$T(n) = 3T(n/2) + 9n$$

simpler proof

simpler proof

$$T(n) = 3T(n/2) + cn$$

Induction hypothesis: `T(n) < dn`$^{1.59}$

It is true for n=1. suppose it is true for n<n0.

$$T(n_0+1) = 3T((n_0+1)/2) + c(n_0+1)$$

$$< 3d[(n_0+1)/2]^{1.59} + c(n_0+1)$$

By the induction hypothesis

$$< 3/2^{1.59} d(n_0+1)^{1.59} + c(n_0+1)$$

$$< 0.997d(n_0+1)^{1.59} + c(n_0+1)$$

Another example: sorting

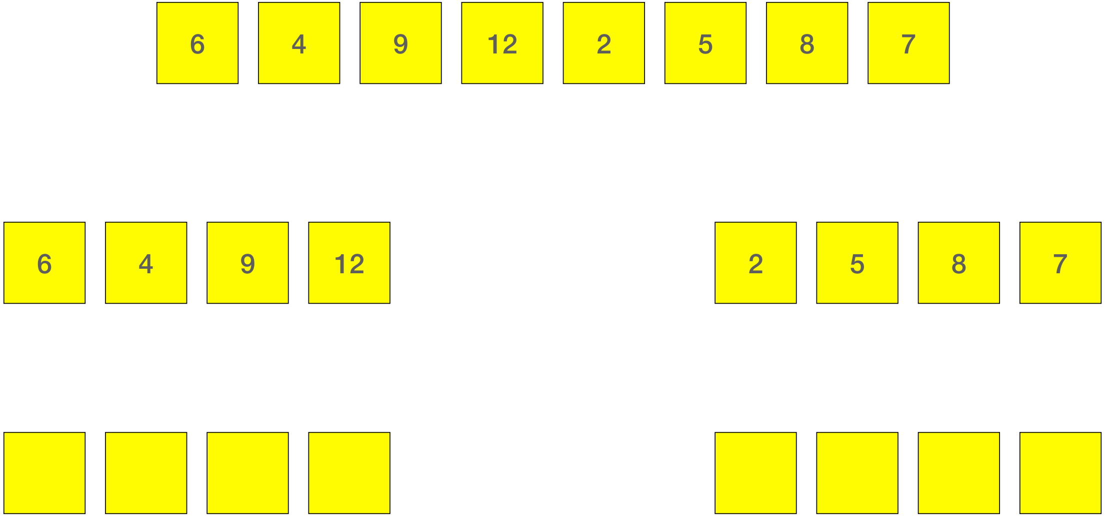
mergesort

goal:

technique:

6	4	9	12	2	5	8	7

mergesort

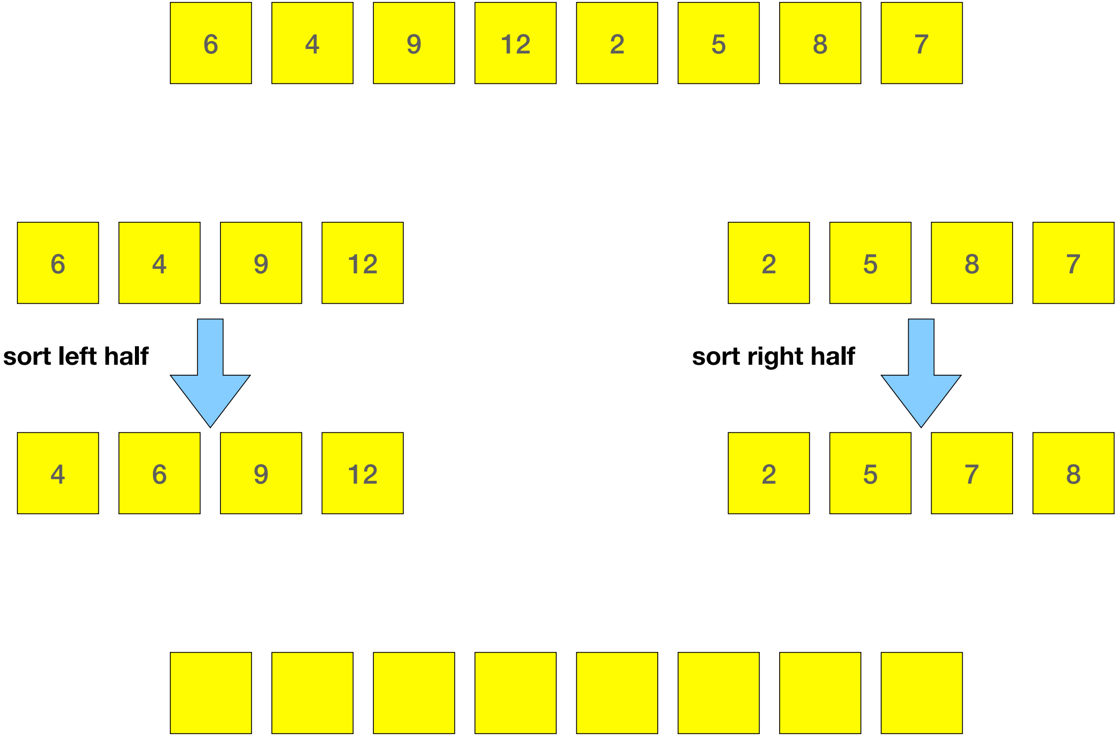


6	4	9	12	2	5	8	7

6	4	9	12

2	5	8	7

mergesort



6	4	9	12	2	5	8	7

6	4	9	12

sort left half

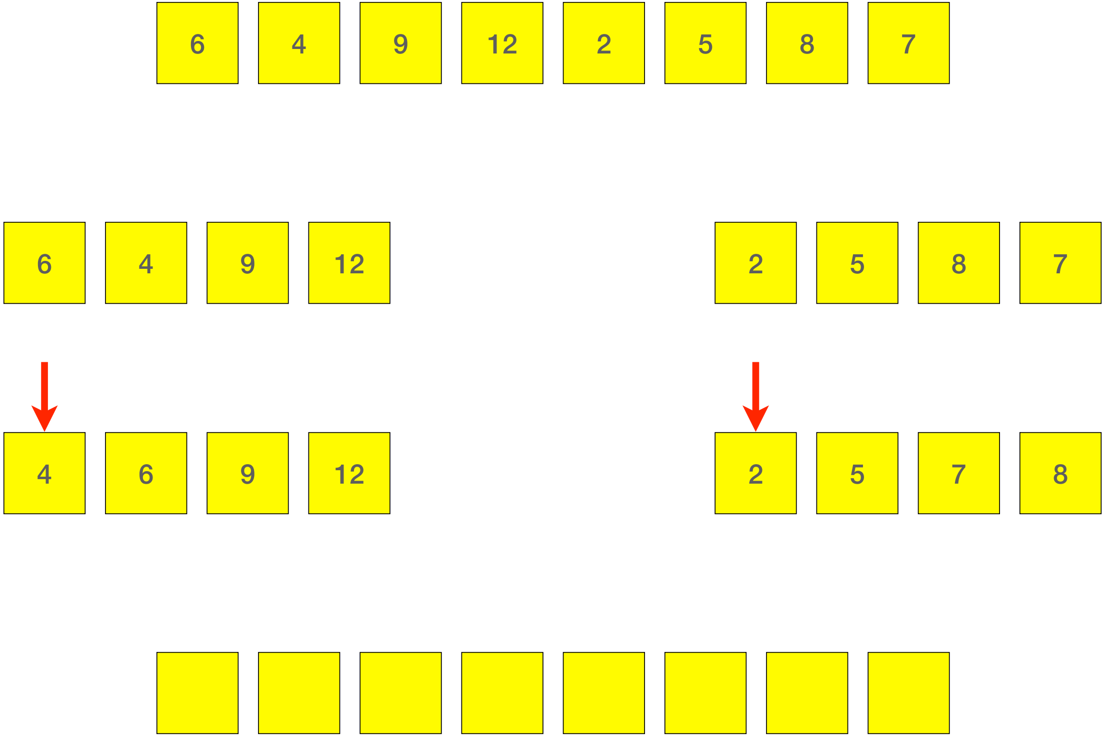
4	6	9	12

2	5	8	7

sort right half

2	5	7	8

mergesort



6 4 9 12 2 5 8 7

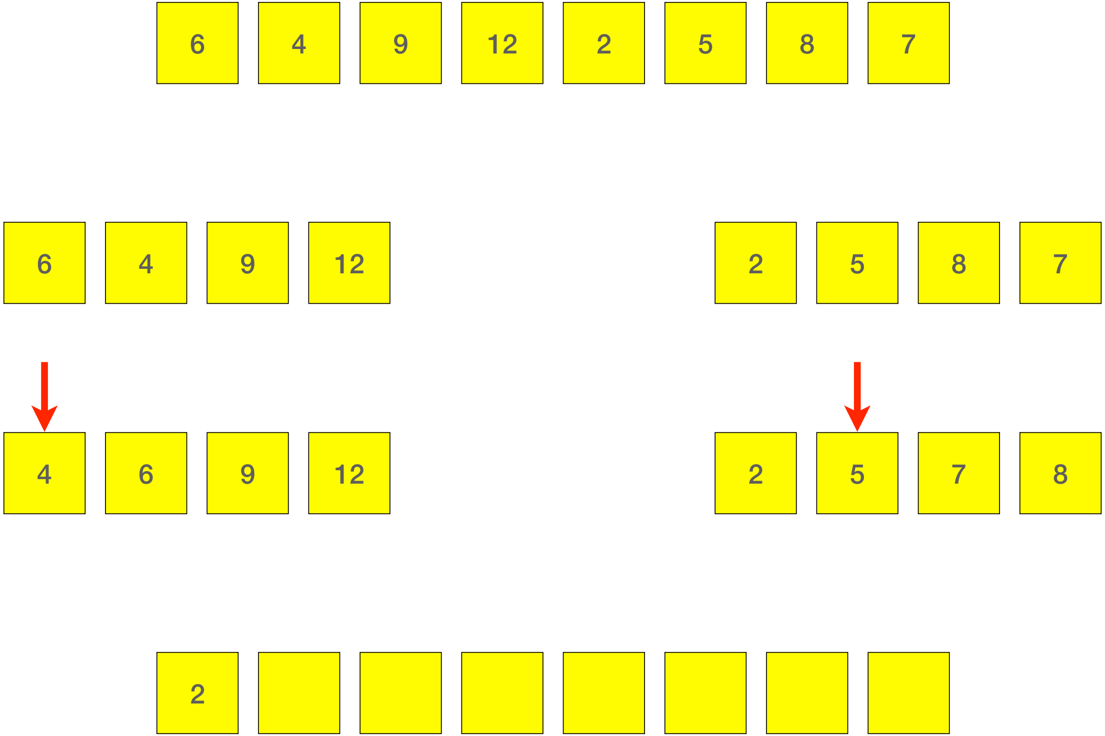
6 4 9 12

2 5 8 7

4 6 9 12

2 5 7 8

mergesort



6 4 9 12 2 5 8 7

6 4 9 12

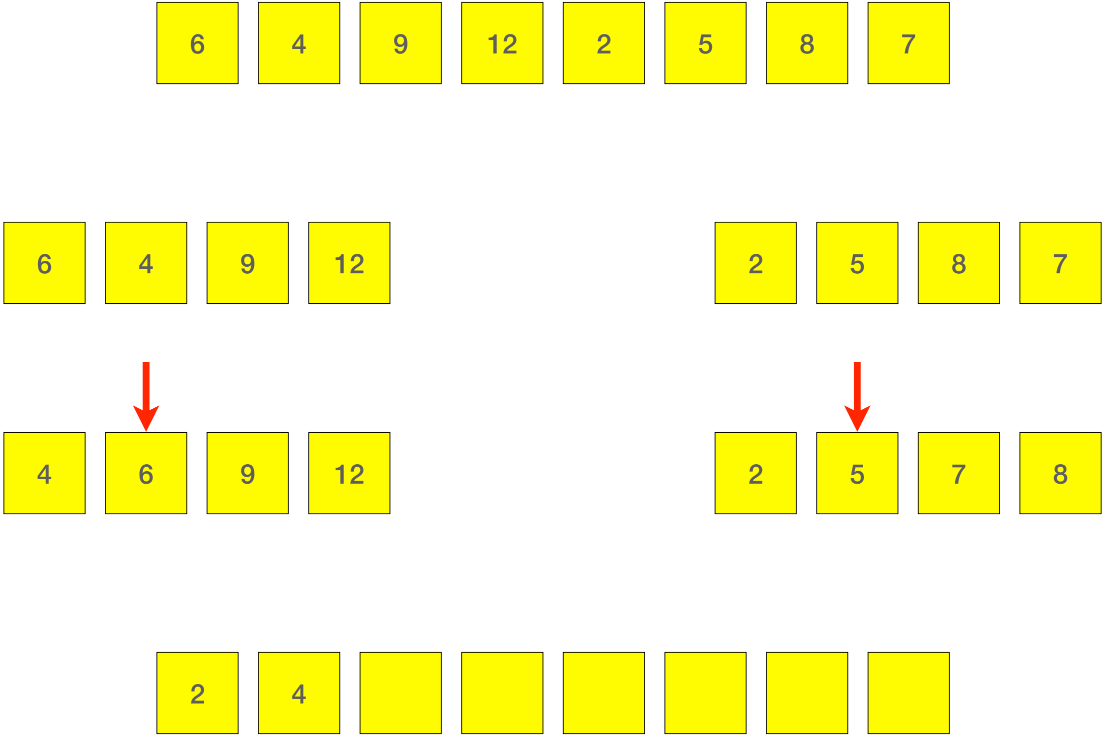
2 5 8 7

4 6 9 12

2 5 7 8

2

mergesort



6 4 9 12 2 5 8 7

6 4 9 12

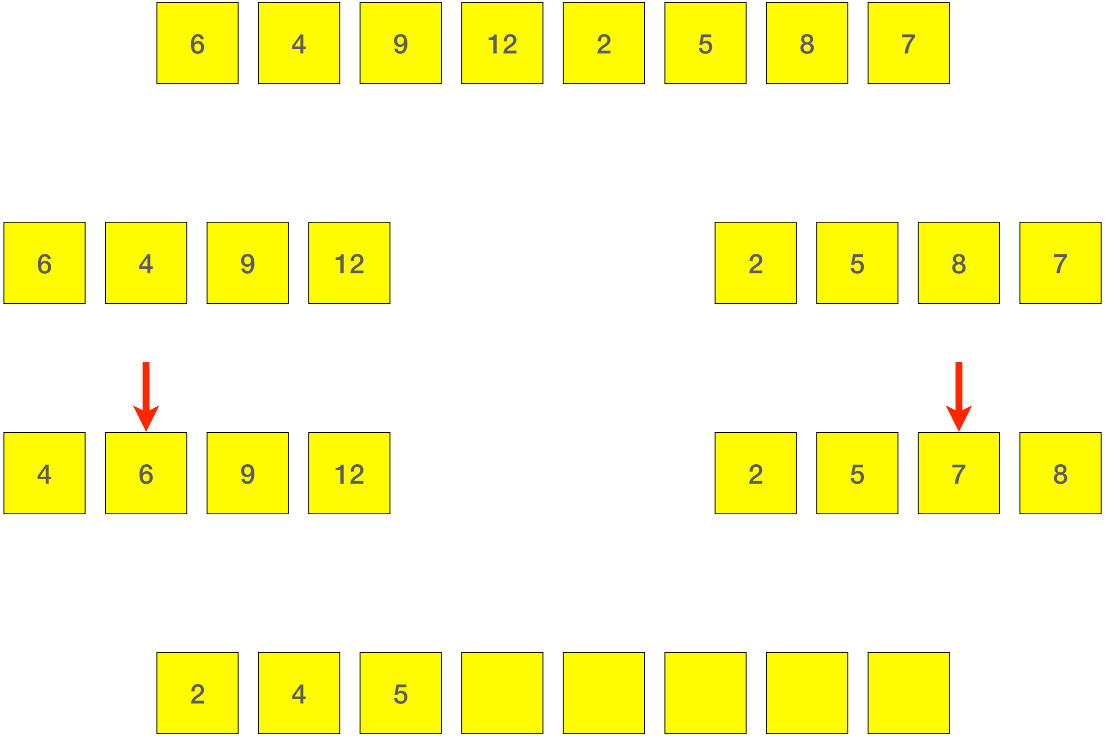
2 5 8 7

4 6 9 12

2 5 7 8

2 4

mergesort



6 4 9 12 2 5 8 7

6 4 9 12

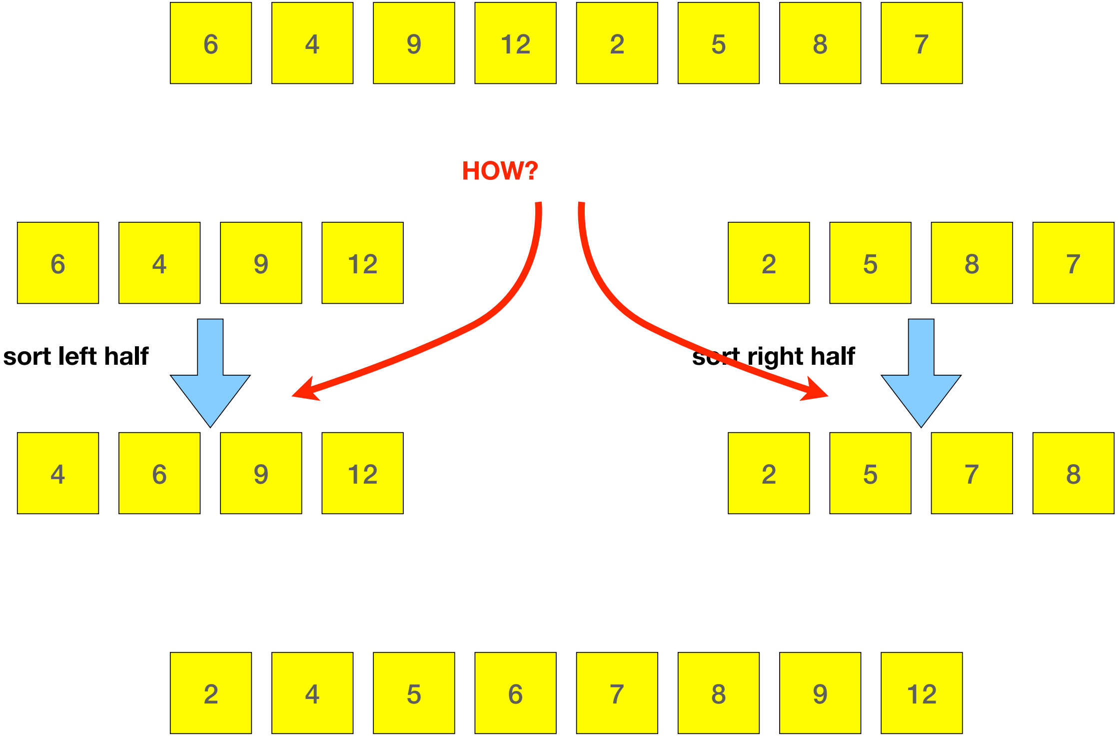
2 5 8 7

4 6 9 12

2 5 7 8

2 4 5

mergesort



6 4 9 12 2 5 8 7

HOW?

6 4 9 12

2 5 8 7

sort left half

sort right half

4 6 9 12

2 5 7 8

2 4 5 6 7 8 9 12

mergesort(`A,start,end`)

❶

❷

❸

❹

❺

mergesort(`A,start,end`)

```
❶ if start < end
❷     $q \leftarrow \lfloor(\mathrm{start} + \mathrm{end})/2\rfloor$
❸     mergesort(A,start,q)
      mergesort(A,q+1,end)
❹     merge(A,start,q,end)
❺         else ...
```

mergesort(A,start,end)

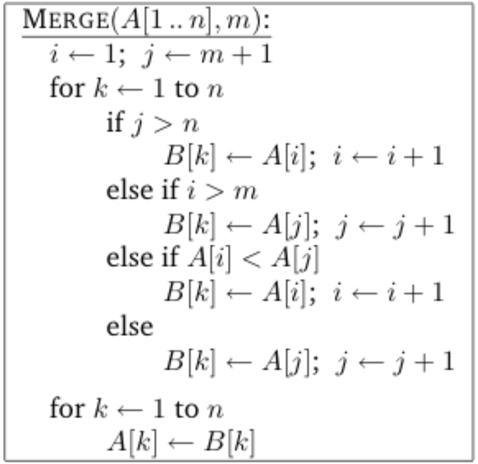
```
❶ if start < end

❷     $q \leftarrow \lfloor(\text{start} + \text{end})/2\rfloor$

❸     mergesort(A,start,q)
      mergesort(A,q+1,end)

❹     merge(A,start,q,end)

❺         else ...
```



```
MERGE($A[1\,..\,n], m$):
  $i \leftarrow 1$; $j \leftarrow m+1$
  for $k \leftarrow 1$ to $n$
      if $j > n$
          $B[k] \leftarrow A[i]$; $i \leftarrow i+1$
      else if $i > m$
          $B[k] \leftarrow A[j]$; $j \leftarrow j+1$
      else if $A[i] < A[j]$
          $B[k] \leftarrow A[i]$; $i \leftarrow i+1$
      else
          $B[k] \leftarrow A[j]$; $j \leftarrow j+1$

  for $k \leftarrow 1$ to $n$
      $A[k] \leftarrow B[k]$
```

jeff erickson

mergesort(`A,start,end`)

Running time?

```
① if start < end
②     $q \leftarrow \lfloor(\text{start} + \text{end})/2\rfloor$
③     mergesort(A,start,q)
      mergesort(A,q+1,end)
④     merge(A,start,q,end)
⑤        else ...
```

$$T(n) = 2T(n/2) + n$$

show:

$$T(n) = 2T(n/2) + n$$

prove:

hypothesis:

base case:

inductive step:

$$T(n) = 2T(n/2) + n$$

prove: $T(n) = O(n \log n)$

property: $T(n) < cn \log n$ **for c>1**

base case:

inductive step:

$$T(n) = 2T(n/2) + n$$

goal is to show $T(n) = \theta(n \log n)$

show: $T(n) \le n \log n$

Proof: Base case holds for $n \le 5$. Assume that the hypothesis holds for all $k \le n$. Consider

$$T(n+1) = 2T\left(\frac{n+1}{2}\right) + (n+1)$$

$$\le 2\left(\frac{n+1}{2}\right)\log\left(\frac{n+1}{2}\right) + n+1$$

$$= (n+1)\left[\log(n+1) - 1\right] + n+1$$

$$= (n+1)\log(n+1) - \cancel{(n+1)} \cancel{+ n+1}$$

$$= (n+1)\log(n+1)$$

$\frac{n+1}{2} < n, \Rightarrow T\left(\frac{n+1}{2}\right) \le \left(\frac{n+1}{2}\right)\log\left(\frac{n+1}{2}\right)$

by ind hypothesis

$$\log\left(\frac{a}{b}\right) = \log(a) - \log(b)$$