More Recursion

Lecture11

Sorting

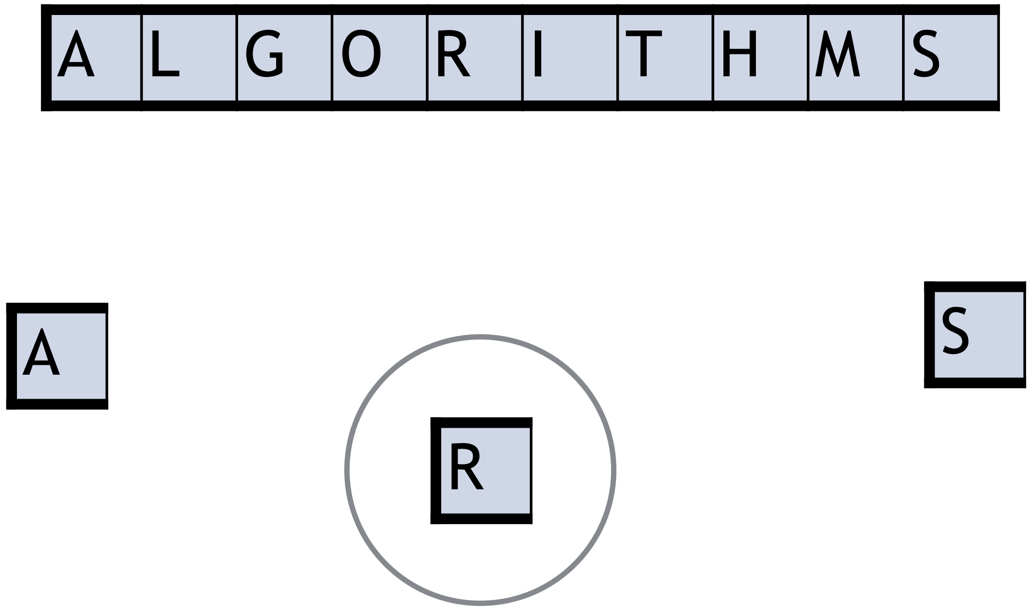
Quicksort:

```
QUICKSORT(A[1 .. n]):
  if (n > 1)
      Choose a pivot element A[p]
      r ← PARTITION(A, p)
      QUICKSORT(A[1 .. r − 1])
      QUICKSORT(A[r + 1 .. n])
```

Running time

- How to choose pivot?
- first/last/middle/median of 3?
- In all cases $T(n) \leq T(n-2)+T(1)+O(n)$

Median of 3



take median of first, last middle

Running time of Quicksort

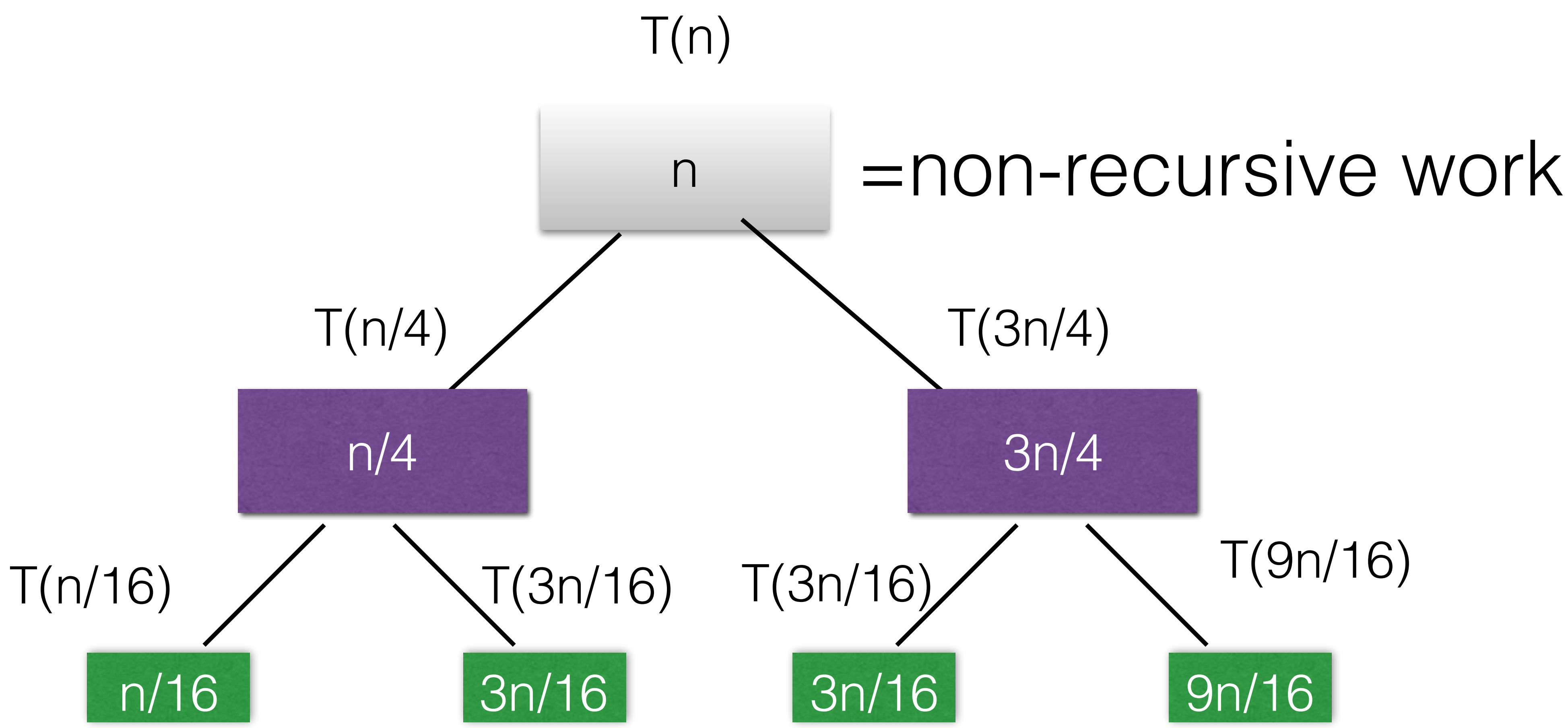
- $O(n^2)$ time!
  - $T(n)=T(n-2)+T(1)+O(n)$

    $= O(n^2)$
- I want
  - Pick an element "near" the middle in $O(n)$ time

Running time of Quicksort

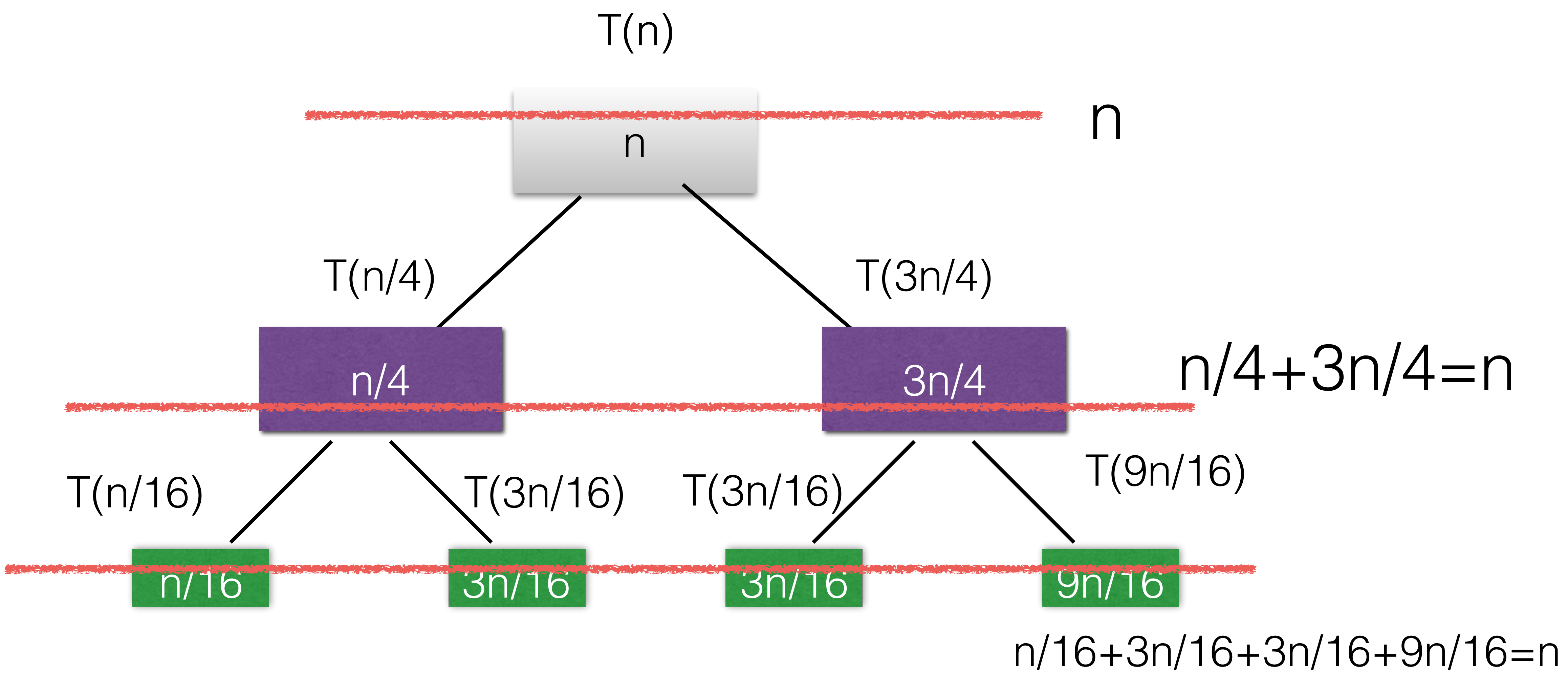
- Then the array partitions in smaller pieces.
- $T(n)=T(n/4)+T(3n/4)+O(n)$
- Solve with recursion trees

Running time of Quicksort



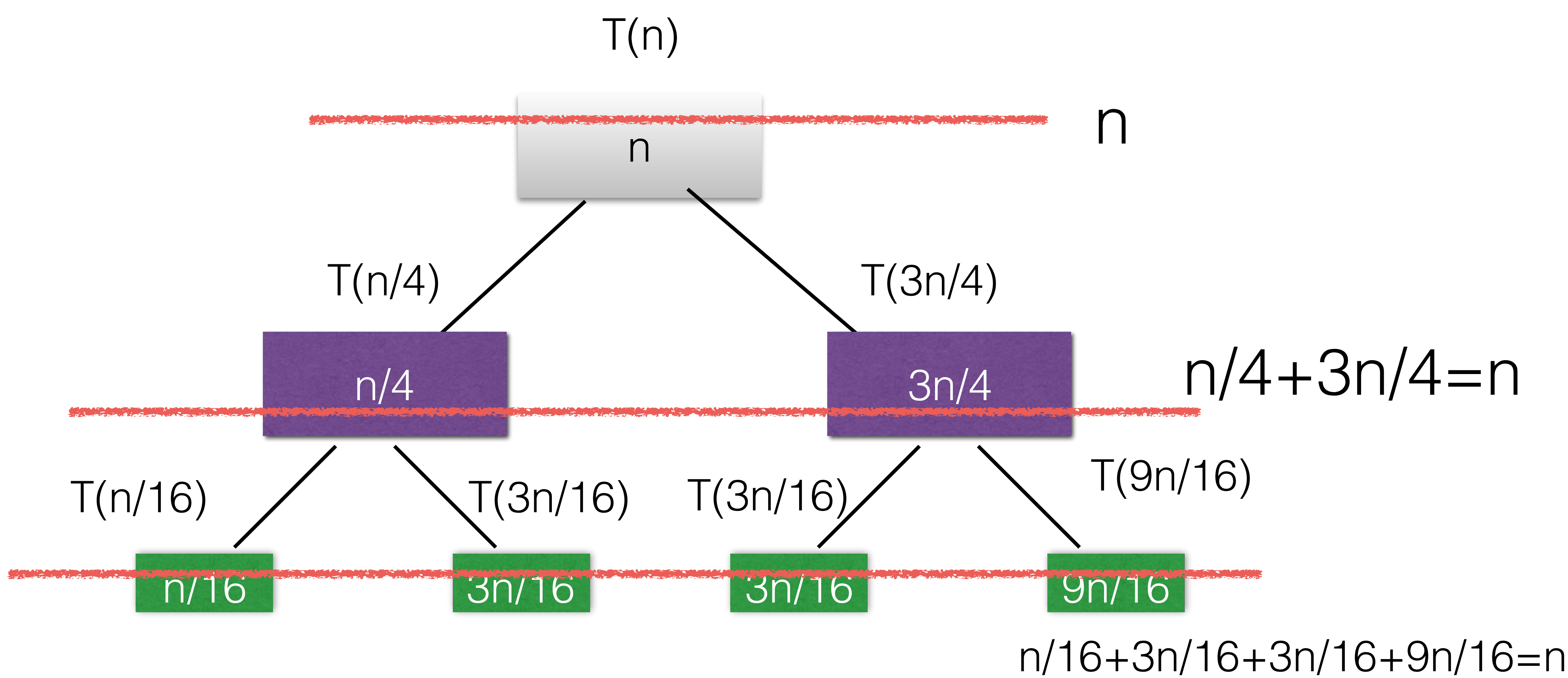
- Leaves on the left are shorter than leaves on the right!
- How do I get a reasonable bound?

Running time of Quicksort



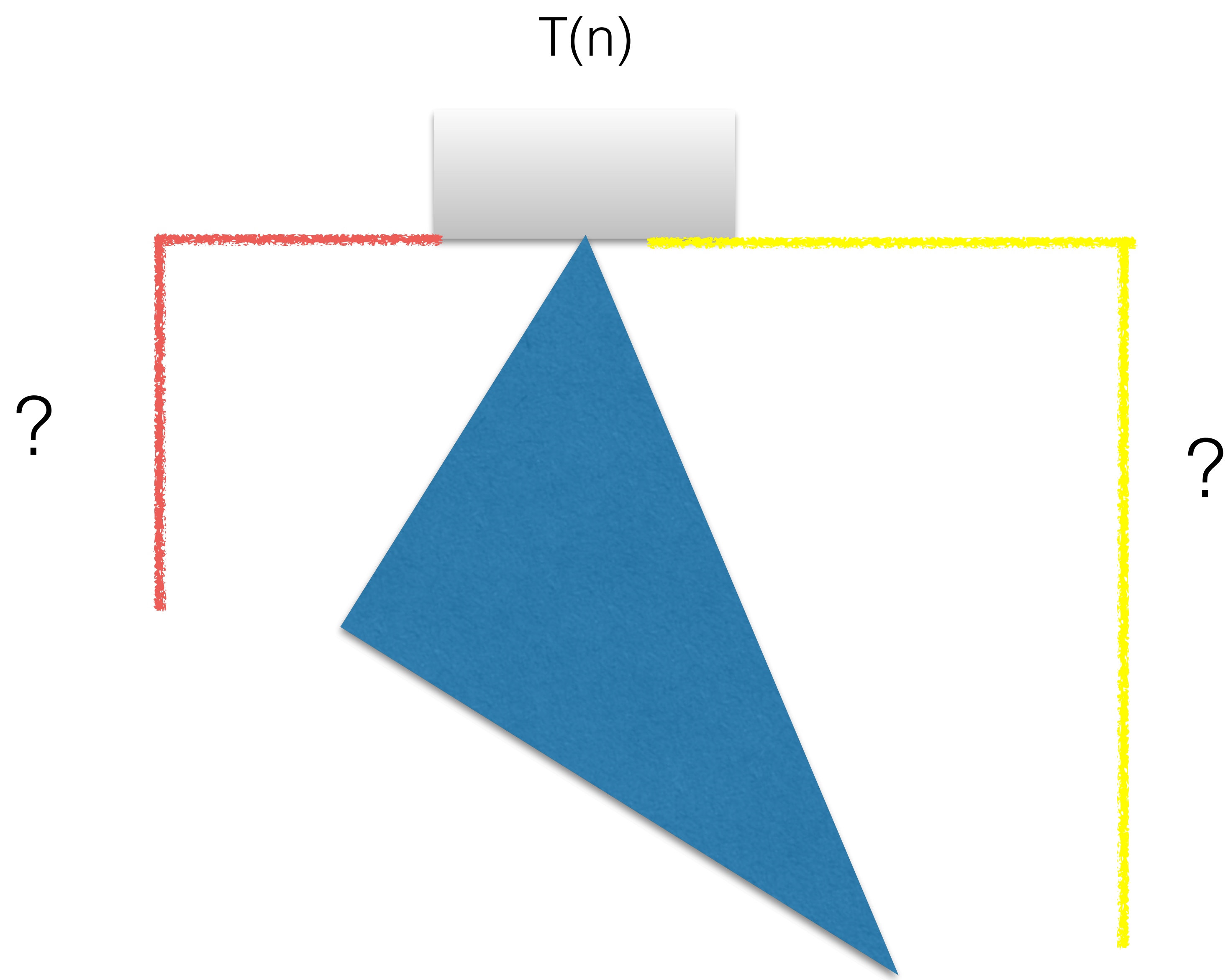
- T(n)=T(n/4)+T(3n/4)+O(n)
- Solve the recurrence by summing up work at each level

Running time of Quicksort



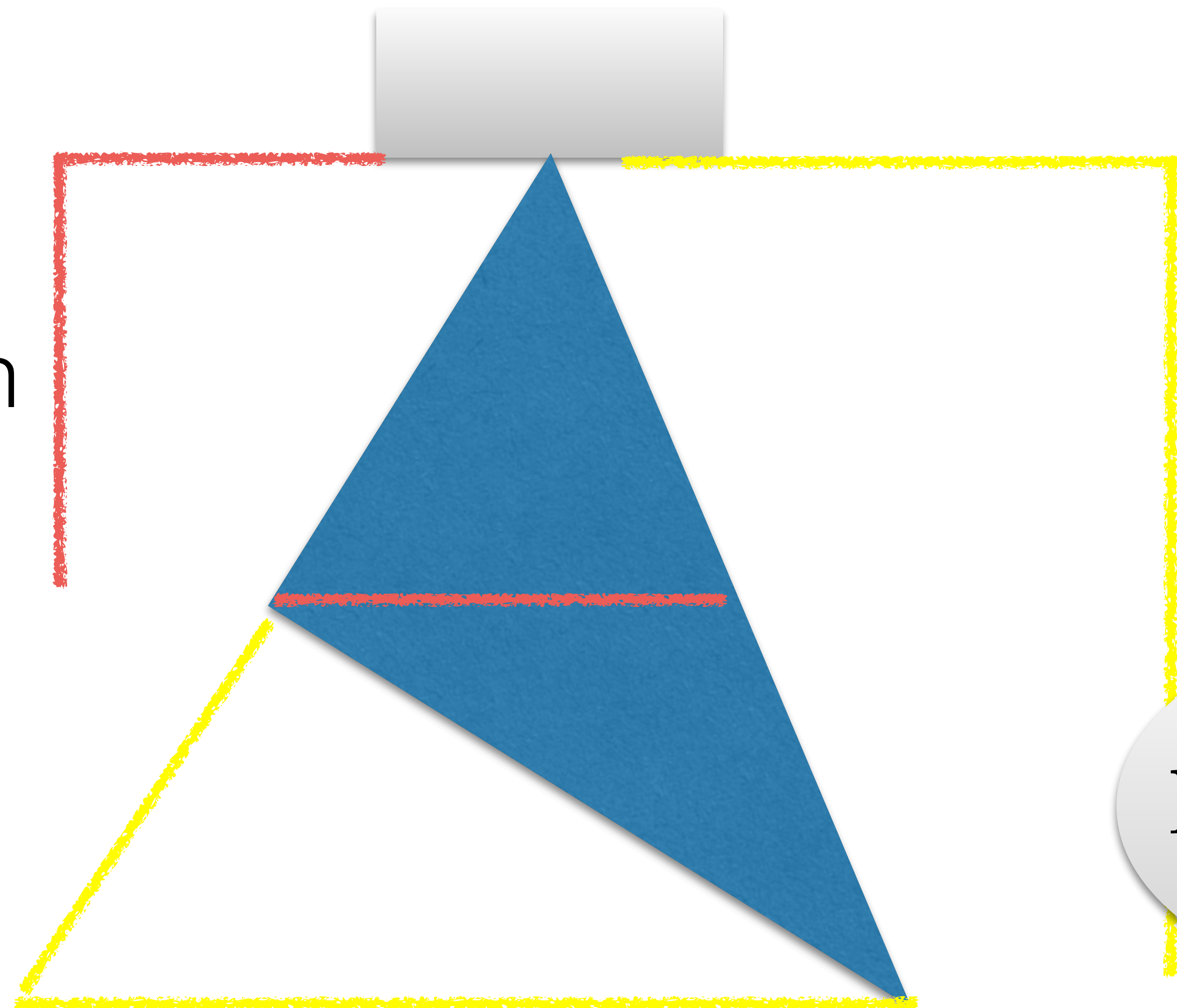
- $T(n)=T(n/4)+T(3n/4)+O(n)$
- What is the depth of shallow and deepest side of tree?

Running time of Quicksort



Running time of Quicksort

T(n)



$$\log_b n = \frac{\log_c n}{\log_c b}$$

$$n \log_4 n \leq T(n) \leq n \log_{4/3} n$$

Running time of Quicksort

- I want AT
  - Pick an element ~~"near"~~ the middle in O(n) time
  - Median Selection : given A[1…n] find |_n/2_| smallest element
  - Doesn't matter exactly which median, if n is even or odd etc

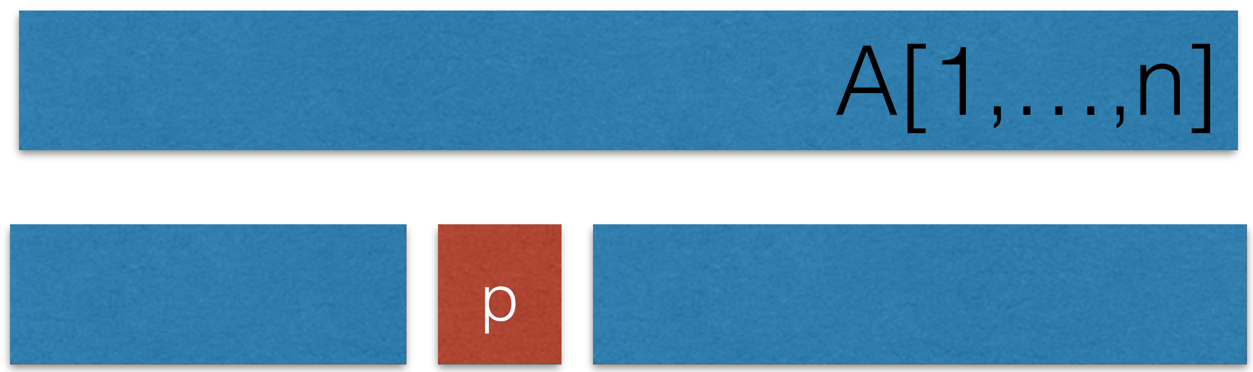
Running time of Quicksort

- I want

  - Pick an element at the middle in O(n) time
  - Median Selection : given A[1…n] find |_n/2_| smallest element
  - Base case: if n<10 brute force
  - Or n<100000000000
  - Inputs less than a googlebyte either constant time or undecidable.

Median Selection

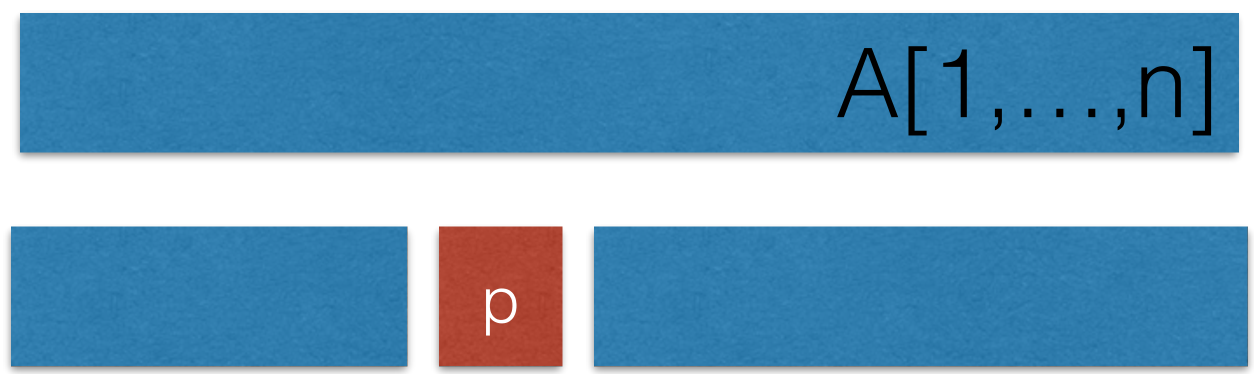
- Median Selection or “One Arm Quicksort”
- Pick a pivot and partition



- If pivot index = n/2, done
- If pivot index < n/2 (left side of the array), then “recurse” on A[p+1,…,n]
- If pivot index > n/2 (right side of the array), then “recurse” on A[1,…,p-1]

Median Selection

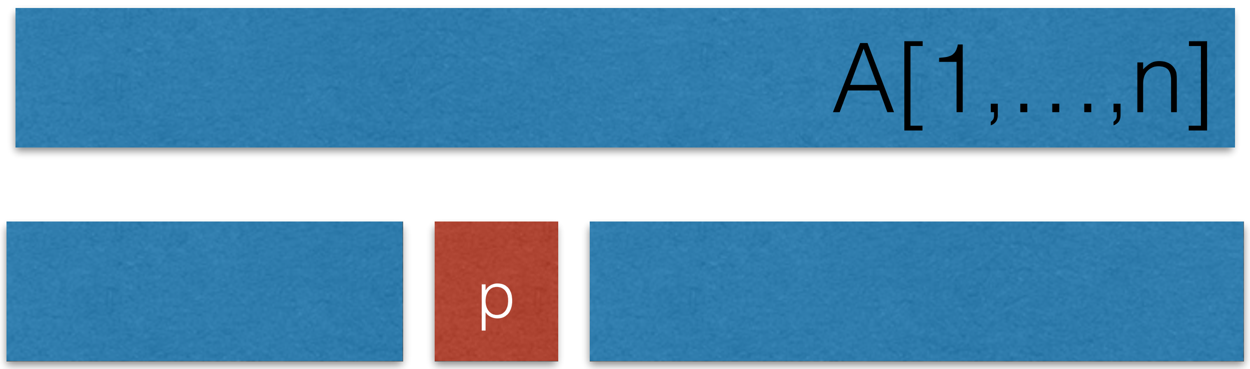
- Median Selection or “One Arm Quicksort”
- Pick a pivot and partition



- If pivot index = n/2, done
- If pivot index < n/2 (left side of the array), then “recurse” on A[p+1,…,n]
- If pivot index > n/2 (right side of the array), then “recurse” on A[1,…,p-1]

Median Selection

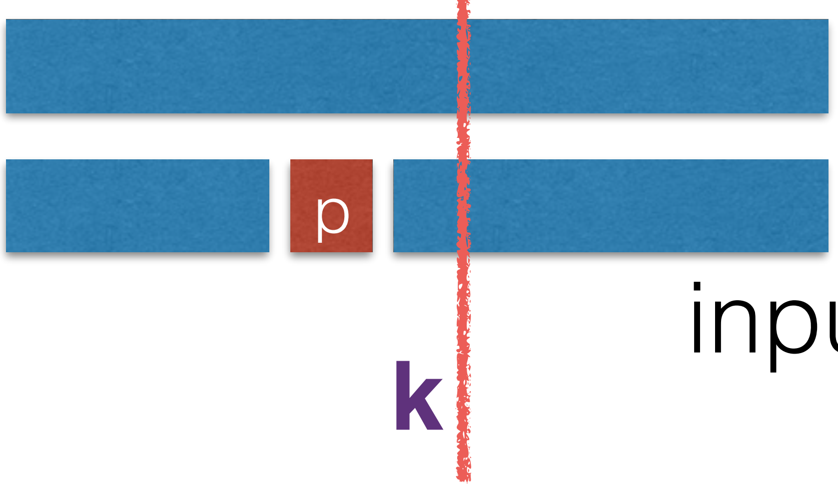
- Recursion would be wrong if I look for median!



- If pivot index = n/2, done
- If pivot index < n/2 (left side of the array), then "recurse" on A[p+1,…,n]
- If pivot index > n/2 (right side of the array), then "recurse" on A[1,…,p-1]

QuickSelect

- More General Problem:



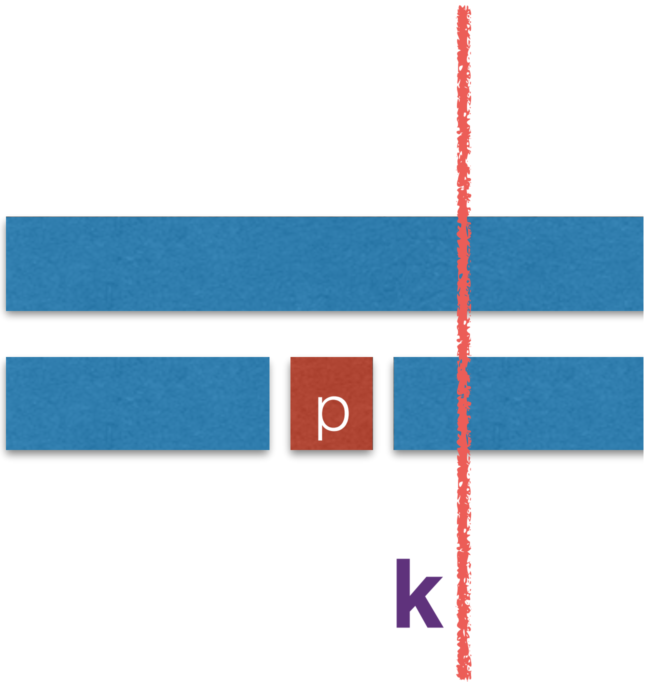
QuickSelect(A[1,…n],k)

input = unsorted array A[1,…n] ,k

output =k th smallest ellement in A

- If pivot index = n/2, done
- If pivot index < n/2 (left side of the array), then recurse on (A[p+1,…,n],k-p)
- If pivot index > n/2 (right side of the array), then recurse on (A[1,…,p-1],k)

QuickSelect



How to choose pivot?
also recursive?

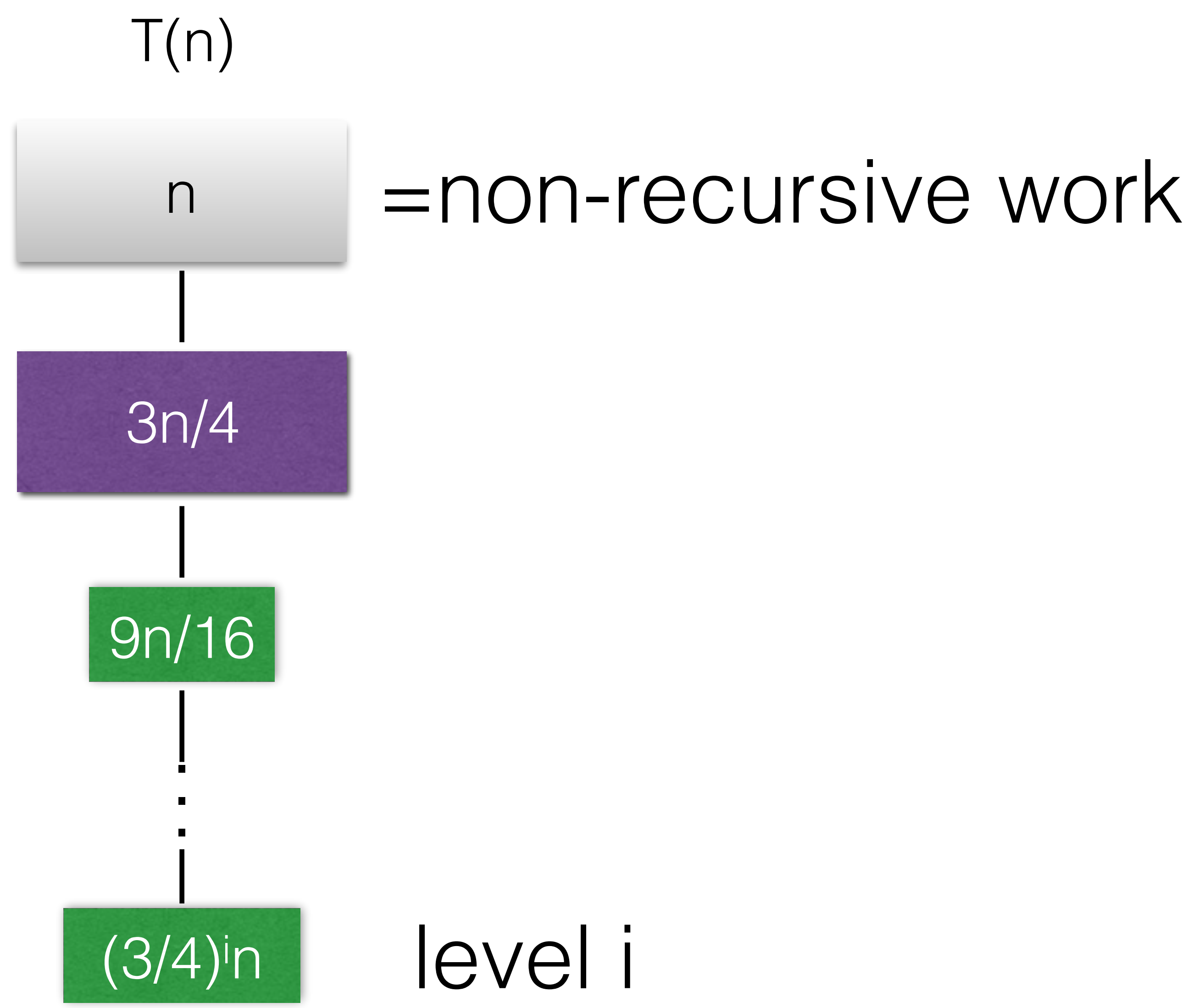
Same result as
Sort A
return A(k)

```
QuickSelect(A[1..n], k):
    if n = 1
        return A[1]
    else
        Choose a pivot element A[p]
        r ← Partition(A[1..n], p)

        if k < r
            return QuickSelect(A[1..r − 1], k)
        else if k > r
            return QuickSelect(A[r + 1..n], k − r)
        else
            return A[r]
```

How to choose pivot?

- If I could choose something near the middle then:
- $T(n) \leq T(3n/4) + O(n)$
- Solve with recursion trees

Recursion Tree



- Sum of all nodes: descending geometric series

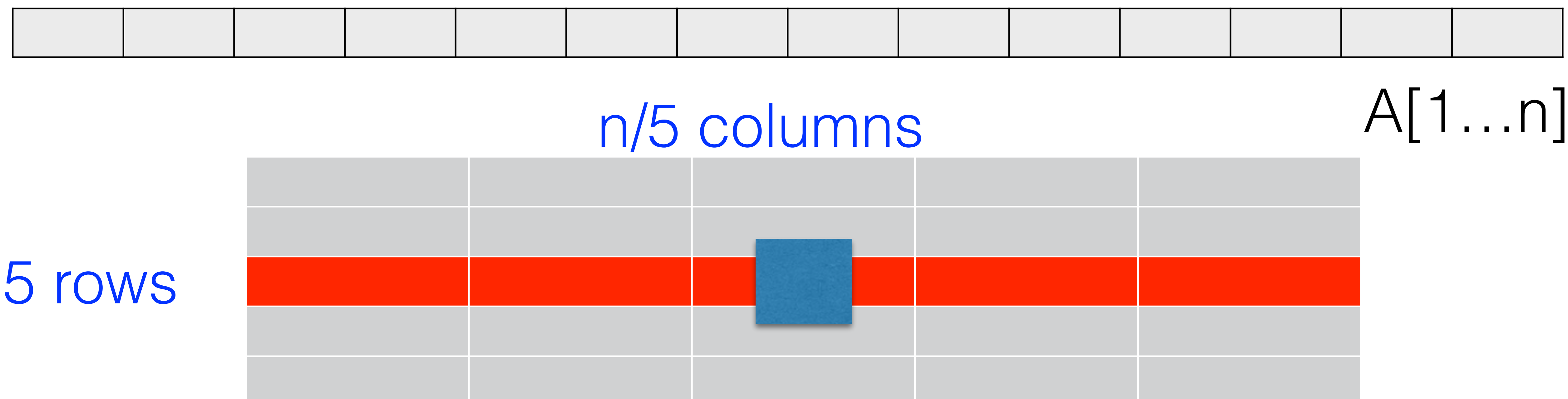
$$n\sum(\frac{3}{4})^i = n\frac{1}{1-3/4} = \frac{4}{3}n = O(n)$$

How to choose pivot?

- If I could choose something near the middle then:
- $T(n) \leq T(3n/4) + O(n)$
- We don't have the magic to choose this pivot yet!

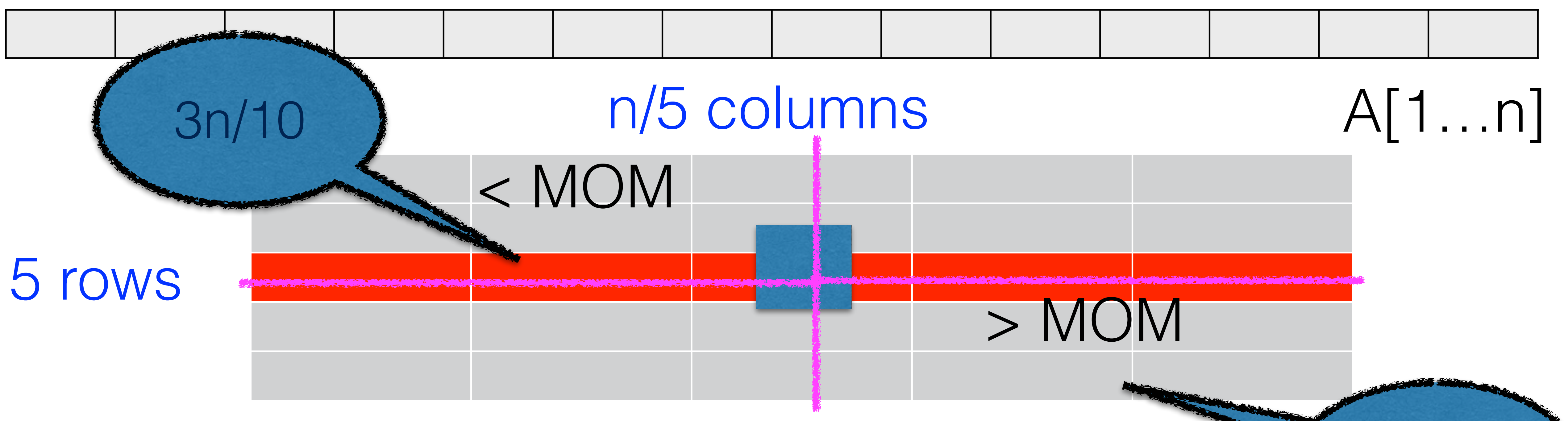
How to choose pivot?

- Imagine I have a two dimensional array:



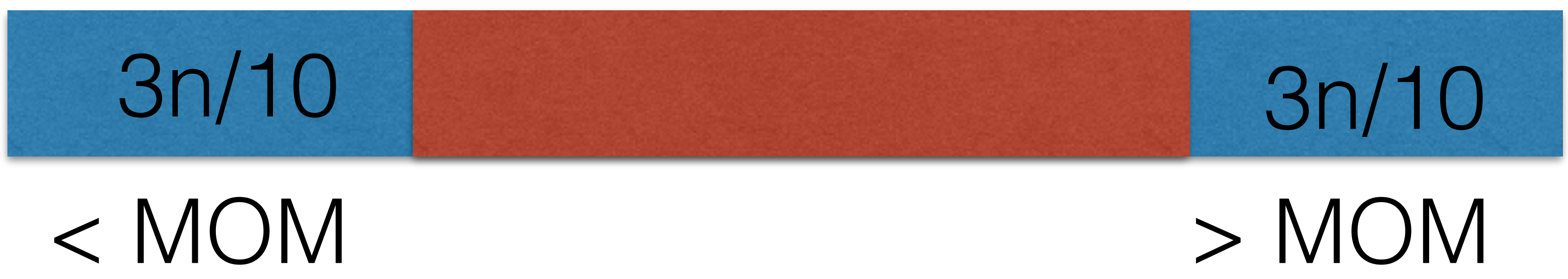
- Find median of each column (time?) O(n) x constant
- Find median of each of those medians (MOM) T(n/5)

How to choose pivot?



- Find median of each column (time?)
- Find median of each of those medians (MOM) T(n/5) time, recursion ferry!
- Use MOM as pivot, then one arm quicksort: $T(n) \leq T(7n/10) + O(n)$

MOMSelect

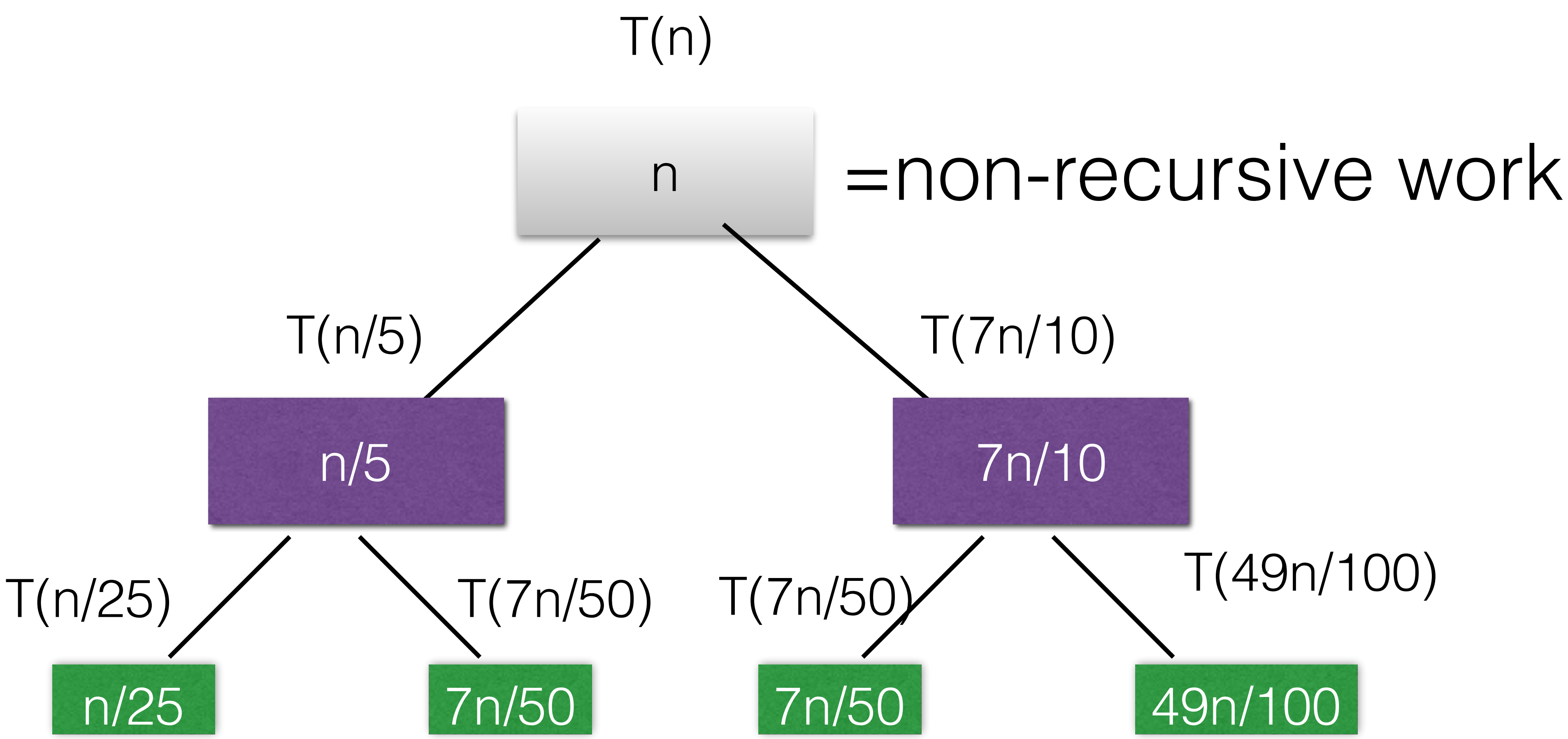



```
MOM5SELECT(A[1..n], k):
  if n ≤ 25
      use brute force
  else
      m ← ⌈n/5⌉
      for i ← 1 to m
          M[i] ← MEDIANOFFIVE(A[5i − 4..5i])    ⟨⟨Brute force!⟩⟩
      mom ← MOMSELECT(M[1..m], ⌊m/2⌋)           ⟨⟨Recursion!⟩⟩

      r ← PARTITION(A[1..n], mom)

      if k < r
          return MOMSELECT(A[1..r − 1], k)      ⟨⟨Recursion!⟩⟩
      else if k > r
          return MOMSELECT(A[r + 1..n], k − r)  ⟨⟨Recursion!⟩⟩
      else
          return mom
```

How to choose pivot?

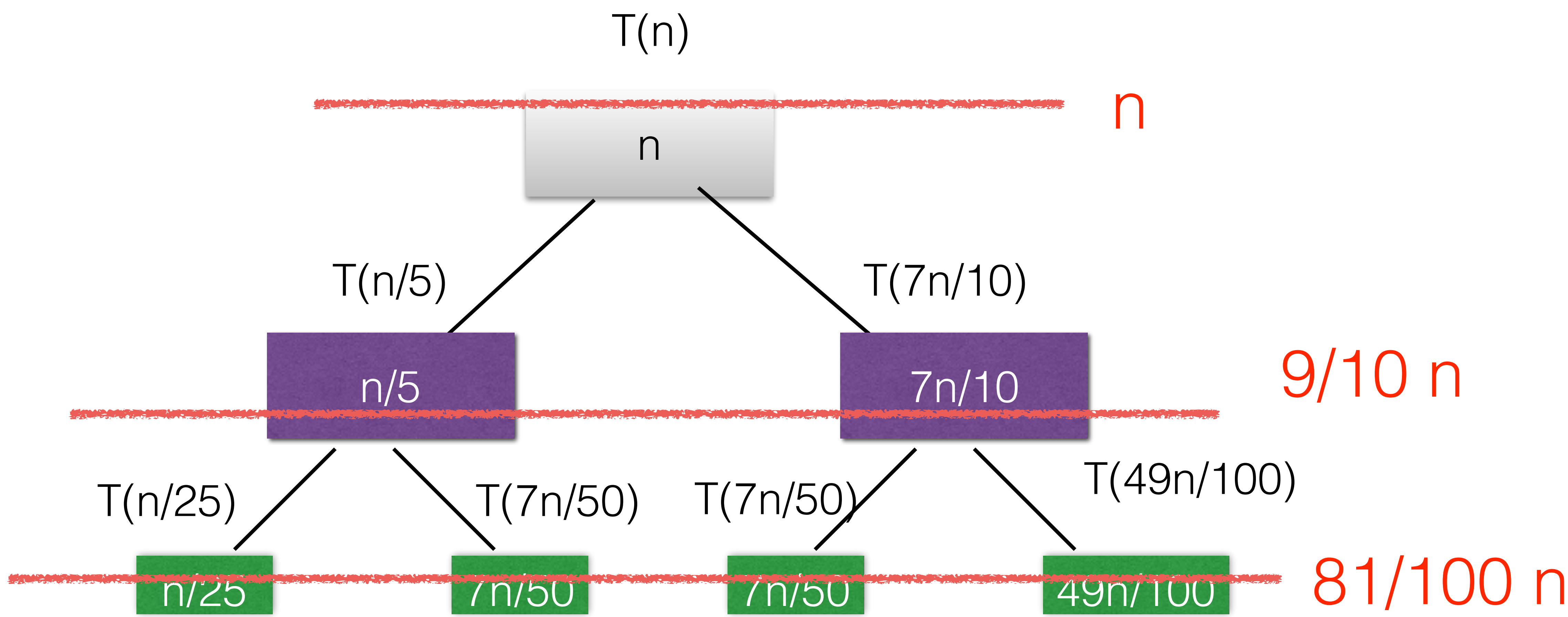
- Subproblem I recurse on has at most 70 per cent of original array.
- Why the number 5?
- $T(n)=T(n/5)+T(7n/10)+O(n)$

Running time of MOM



- $T(n)=T(n/5)+T(7n/10)+O(n)$

Running time of MOM

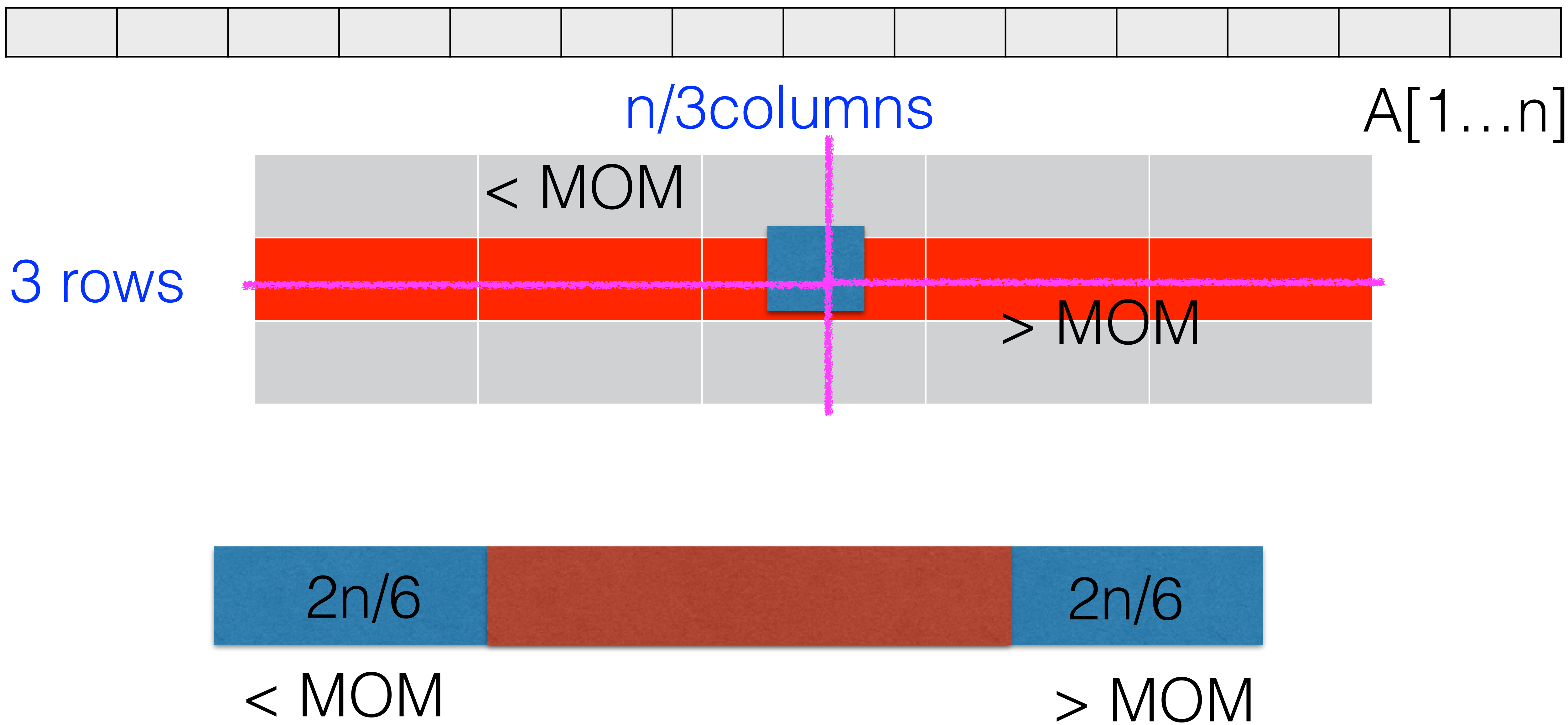


level i: $(9/10)^i n$

$T(n)=T(n/5)+T(7n/10)+O(n)$

Sum is geometric series, adds up to O(n) but large constant!

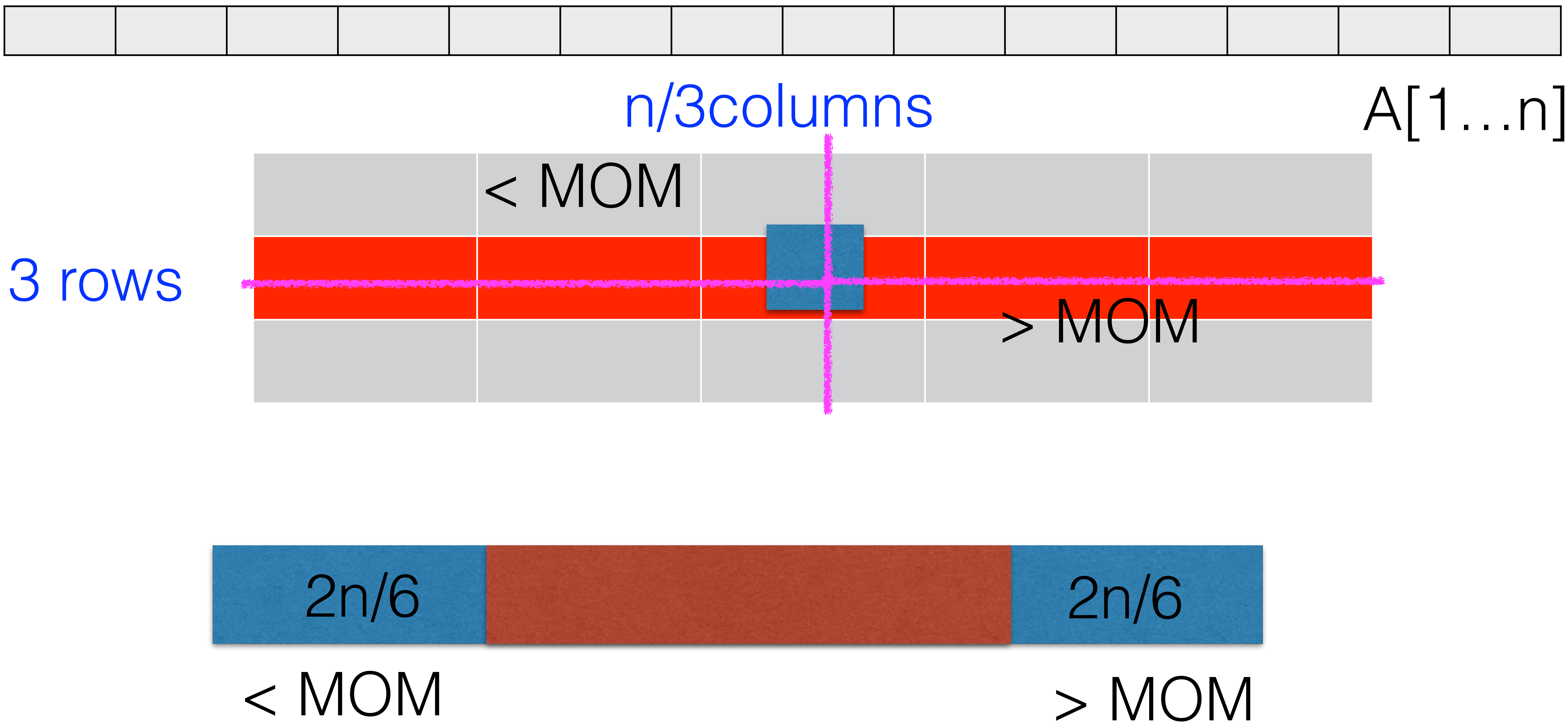
Why 5?



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

what is the solution?

Why 5?



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

n log n, like sorting already!

Multiplication

- How to multiply two n digit numbers?

```
        31415962
      × 27182818
       251327696
       31415962
     251327696
     62831924
   251327696
   31415962
 219911734
 62831924
 853974377340916
```

- two nested for loops
- runs in $O(n^2)$ time
- Is this the best you can do?

Multiplication

- How to multiply two n digit numbers?

n/2	n/2	$10^m a+b$
n/2	n/2	$10^m c+d$

$$(10^m a + b)(10^m c + d) = 10^{2m} ac + 10^m (bc + ad) + bd$$

reduce to multiply 4 n/2 digit numbers!

Multiplication

$10^m a+b$	n/2	n/2
$10^m c+d$	n/2	n/2

```
MULTIPLY(x, y, n):
    if n = 1
        return x · y
    else
        m ← ⌈n/2⌉
        a ← ⌊x/10^m⌋;  b ← x mod 10^m
        d ← ⌊y/10^m⌋;  c ← y mod 10^m
        e ← MULTIPLY(a, c, m)
        f ← MULTIPLY(b, d, m)
        g ← MULTIPLY(b, c, m)
        h ← MULTIPLY(a, d, m)
        return 10^{2m} e + 10^m (g + h) + f
```

$T(n)=3T(n/2)+O(n)=O(n^2)$

Didn't help much :(

Multiplication

$10^m a+b$	n/2	n/2
$10^m c+d$	n/2	n/2

$$(10^m a + b)(10^m c + d) = 10^{2m} ac + 10^m (bc + ad) + bd$$

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

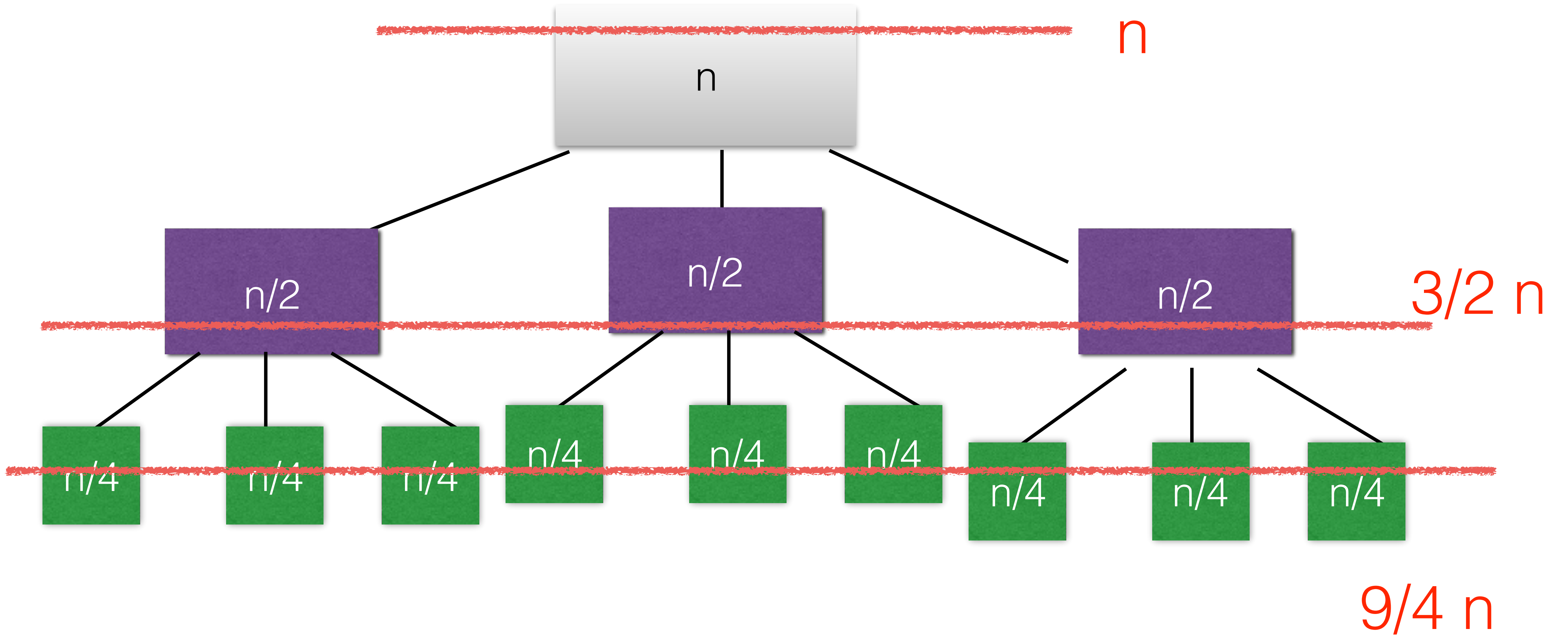
Multiplication

$10^m a+b$	n/2	n/2
$10^m c+d$	n/2	n/2

FASTMULTIPLY(x, y, n):
  if $n = 1$
    return $x \cdot y$
  else
    $m \leftarrow \lceil n/2 \rceil$
    $a \leftarrow \lfloor x/10^m \rfloor$; $b \leftarrow x \bmod 10^m$
    $d \leftarrow \lfloor y/10^m \rfloor$; $c \leftarrow y \bmod 10^m$
    $e \leftarrow$ FASTMULTIPLY(a, c, m)
    $f \leftarrow$ FASTMULTIPLY(b, d, m)
    $g \leftarrow$ FASTMULTIPLY$(a - b, c - d, m)$
    return $10^{2m} e + 10^m (e + f - g) + f$

Multiplication Running time

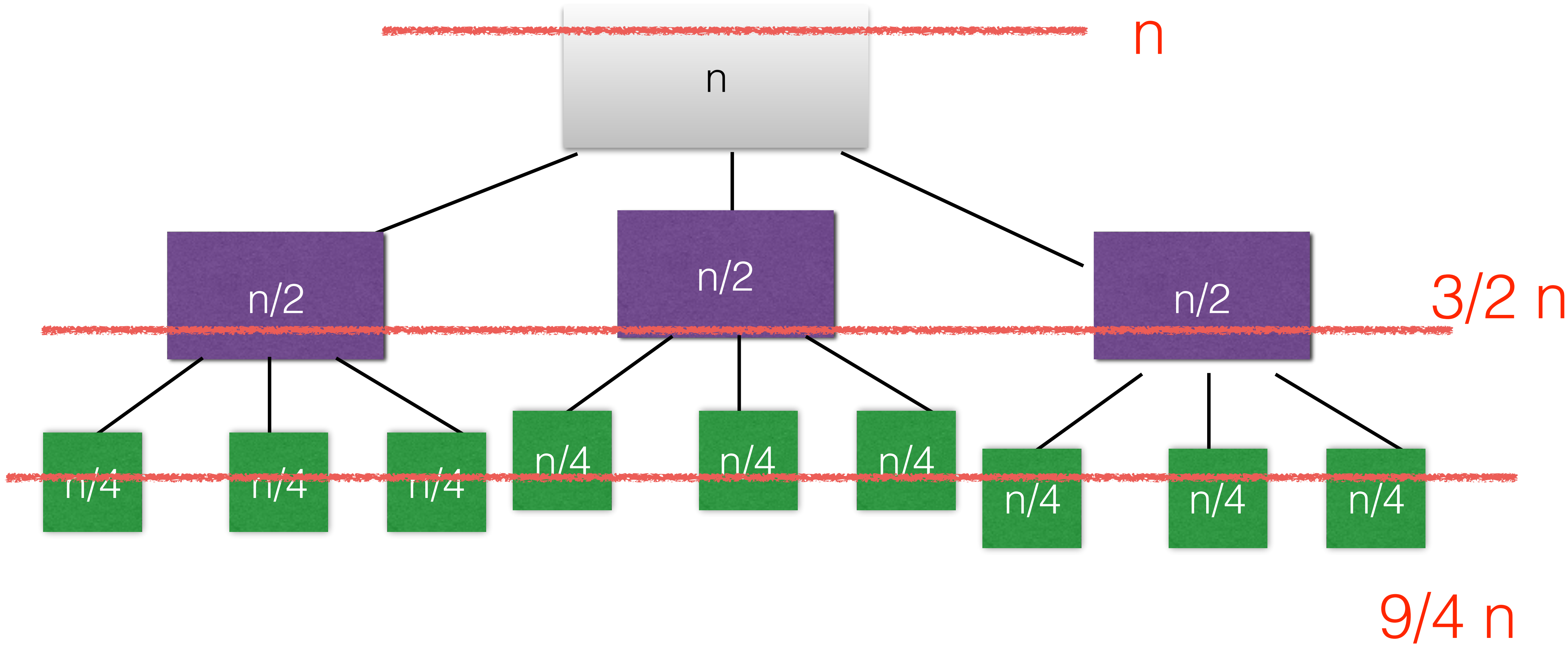
$T(n)=3T(n/2)+O(n)$



- Ascending geometric series,every level has 50% more work than the previous one
- the only work that matters is at the last level (leaves)

Multiplication Running time

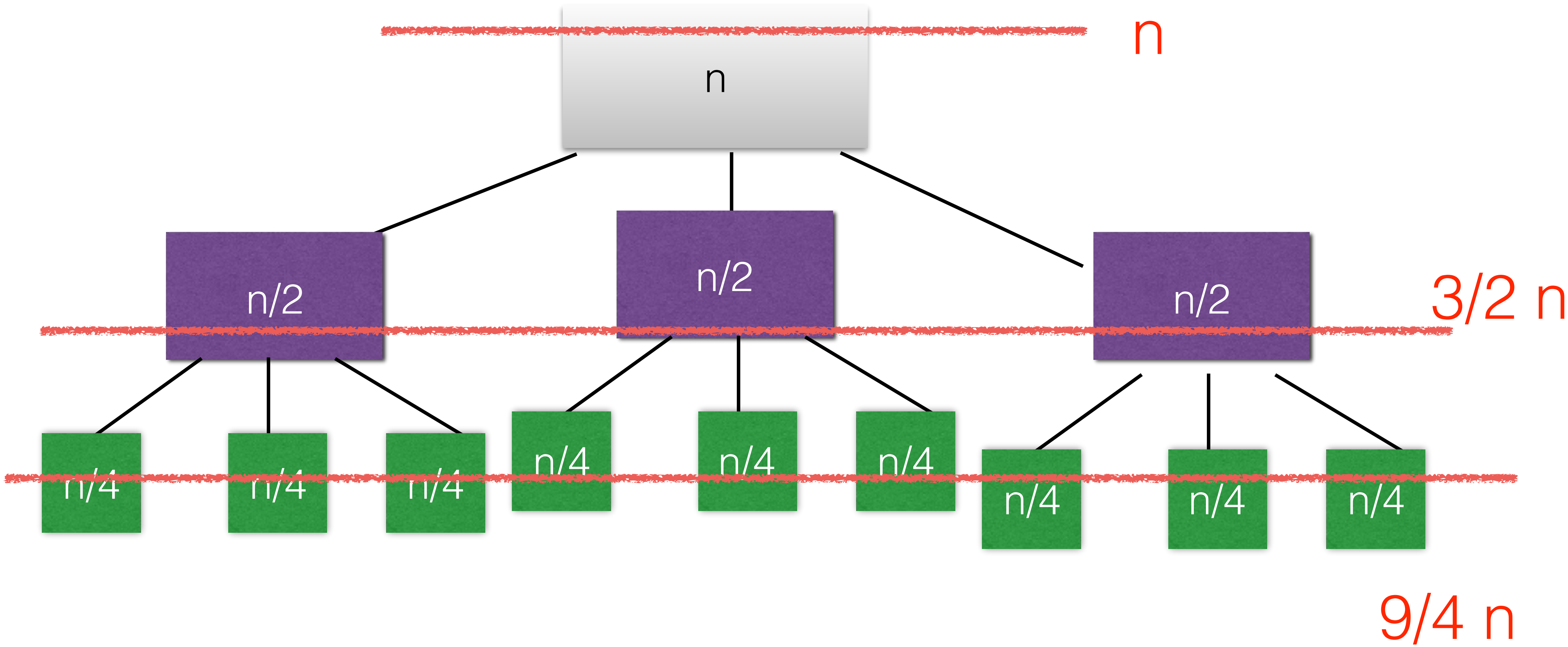
$T(n)=3T(n/2)+O(n)$



- At the leaves, we have constant size, assume it's 1.
- total work ≈ # leaves x 1.
- how many leaves?

Multiplication Running time

T(n)=3T(n/2)+O(n)



- # leaves = $3^{\text{depth}} = 3^{\log_2 n} = n^{\log_2 3} = n^{1.6}$