CS 4102: Algorithms

Lecture 8: Quickselect, Median of Medians

David Wu

Fall 2019

Warm Up

Guess the solution to this recurrence:

$$T(n) = T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) + c \cdot n$$

where $c \geq 1$ is a constant

Warm Up

$$T(n) = T(n/5) + T(7n/10) + c \cdot n$$

$$\frac{n}{5} + \frac{7n}{10} = \frac{9n}{10} < n$$

If this was $T\left(\frac{9n}{10}\right)$, then can use Master's Theorem to conclude $\Theta(n)$

Guess: $\Theta(n)$

Suffices to show $O(n)$ since non-recursive cost is already $\Omega(n)$

Warm Up

$$T(n) = T(n/5) + T(7n/10) + c \cdot n$$

Claim: $T(n) \leq 10cn$

Base Case: $T(0) = 0$

$T(1) = 1 \leq 10c$ which is true since $c \geq 1$

Strictly speaking, we can handle any $c > 0$, but assuming $c \geq 1$ to simplify the analysis here

Warm Up

$$T(n) = T(n/5) + T(7n/10) + c \cdot n$$

Inductive hypothesis: $\forall n \leq x_0 : T(n) \leq 10cn$

Inductive step:

$$T(x_0+1) = T\left(\frac{1}{5}(x_0+1)\right) + T\left(\frac{7}{10}(x_0+1)\right) + c(x_0+1)$$

$$\leq \left(\frac{1}{5} + \frac{7}{10}\right) 10c(x_0+1) + c(x_0+1)$$

$$= 9c(x_0+1) + c(x_0+1) = 10c(x_0+1)$$

Today’s Keywords

Divide and Conquer

Sorting

Quicksort

Median

Order Statistic

Quickselect

Median of Medians

CLRS Readings: Chapter 7

Homework

HW2 due **today (September 19), 11pm**

- Programming assignment (Python or Java)
- Divide and conquer (Closest pair of points)

HW3 released tonight

- Divide and conquer algorithms
- Written (use LaTeX!) – Submit both **zip** and **pdf**!

Quickselect Algorithm

Algorithm to compute the i^{th} order statistic

- i^{th} smallest element in the list
- 1^{st} order statistic: minimum
- n^{th} order statistic: maximum
- $(n/2)^{\text{th}}$ order statistic: median

Quickselect Algorithm

Finds i^{th} order statistic

General idea: choose a pivot element, partition around the pivot, and recurse on sublist containing index i

Divide: select pivot element p, Partition(p)

Conquer:

- if $i =$ index of p, then we are done and return p
- if $i <$ index of p recurse left. Otherwise, recurse right (with index $i - p$)

Combine: Nothing!

Partition Procedure (Divide Step)

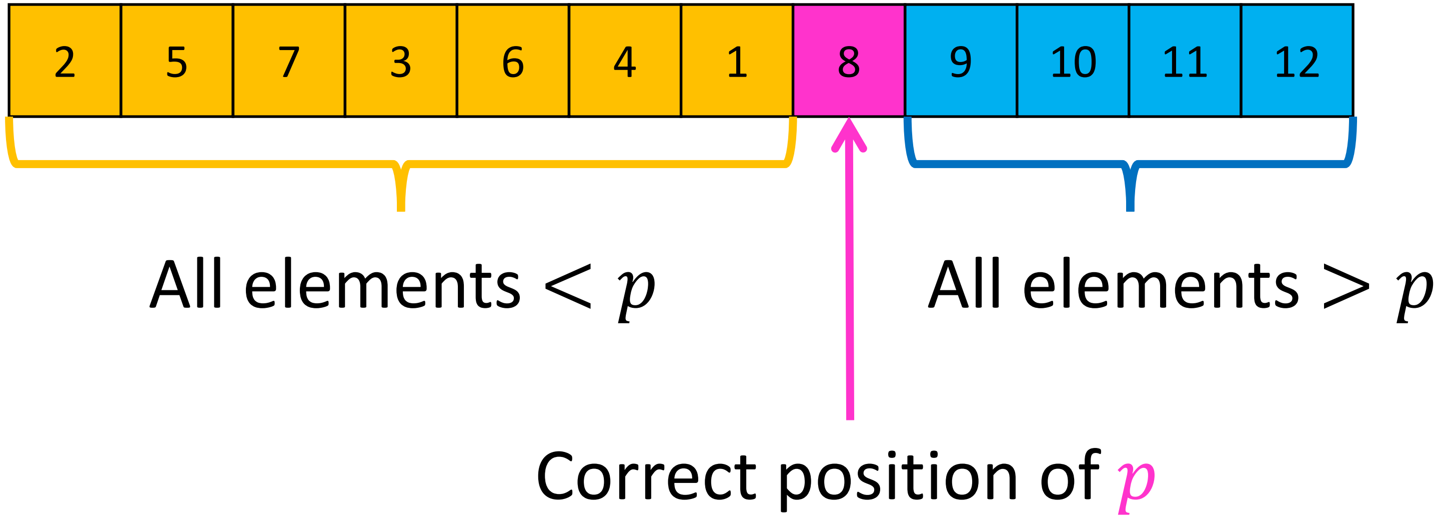
Input: an <u>unordered</u> list, a pivot p

8	5	7	3	12	10	1	2	4	9	6	11

Goal: All elements $< p$ on left, all $\geq p$ on right

5	7	3	1	2	4	6	8	12	10	9	11

Conquer Step



Recurse on sublist that contains index i

(add index of the pivot to i if recursing right)

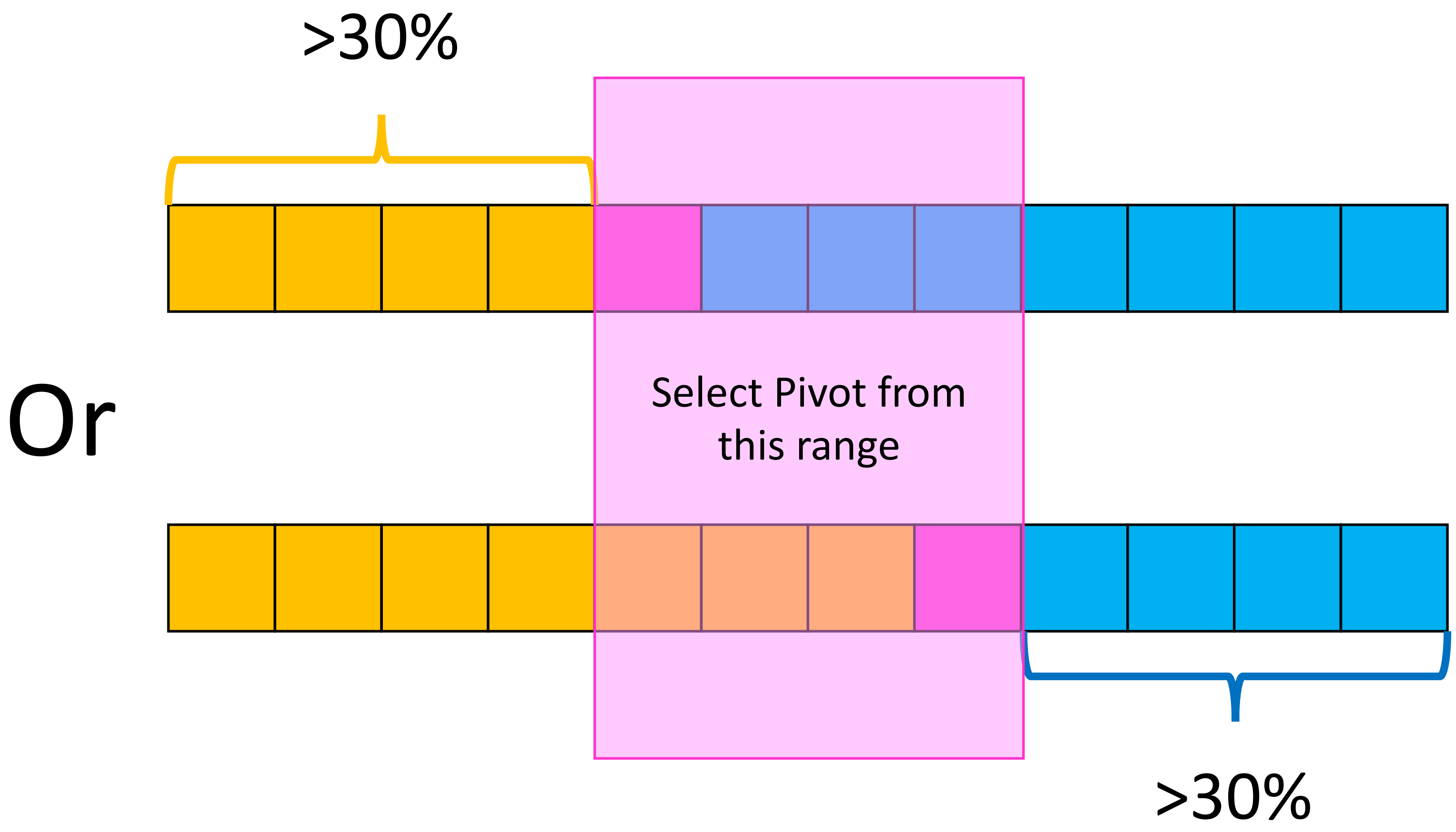
How to Choose the Pivot?

Good choice: $\Theta(n)$

Bad choice: $\Theta(n^2)$

Good Pivot

Decent pivot: both sides of Pivot >30%



Median of Medians

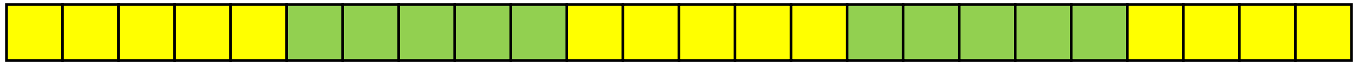
Fast way to select a "good" pivot

Guarantees pivot is greater than $\approx$30% of elements and less than $\approx$30% of the elements

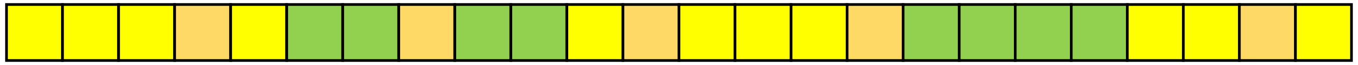
Main idea: break list into blocks, find the median of each blocks, use the median of those medians

Median of Medians

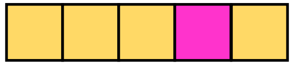
1. Break list into blocks of size 5



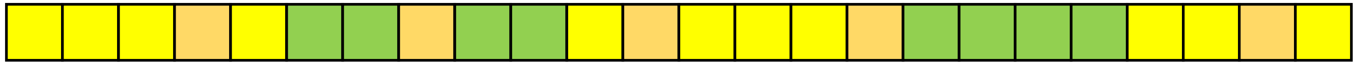
2. Find the median of each chunk



3. Return median of medians (using Quickselect)

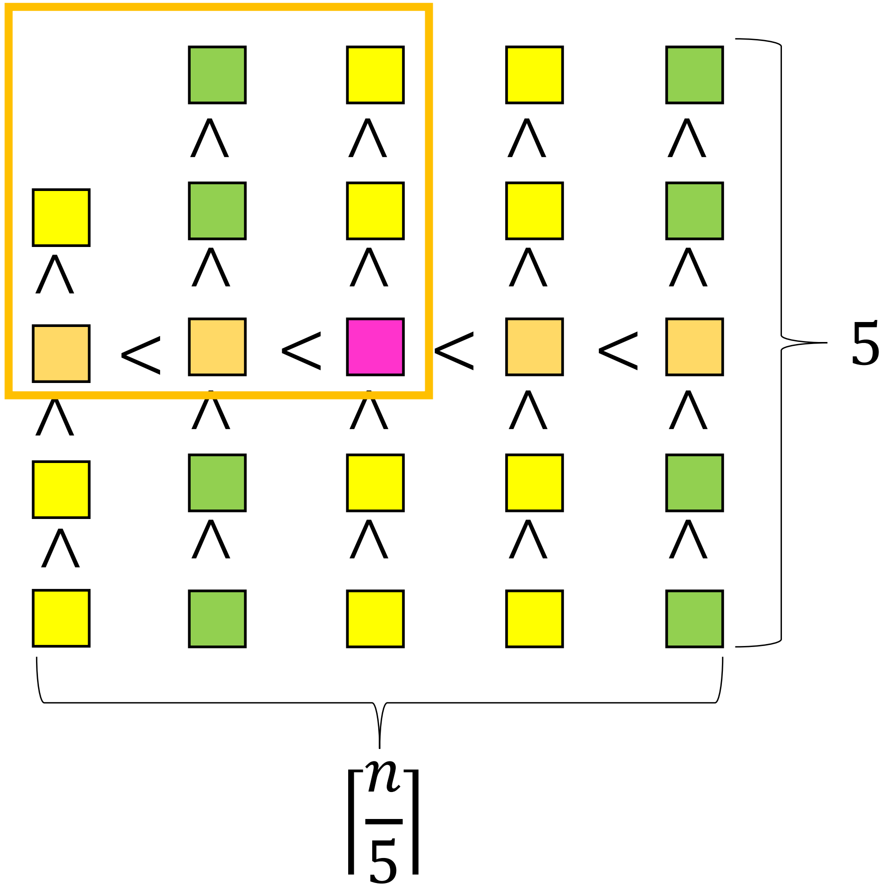


Median of Medians



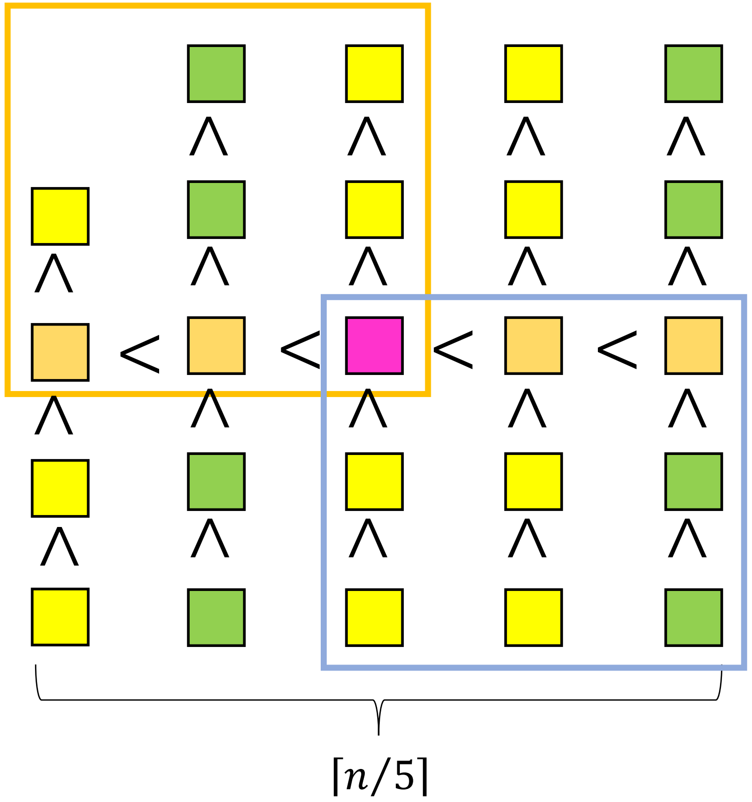
Each chunk sorted, chunks ordered by their medians

MedianofMedians
is larger than all
of these



Median of Medians

MedianofMedians is larger than all of these



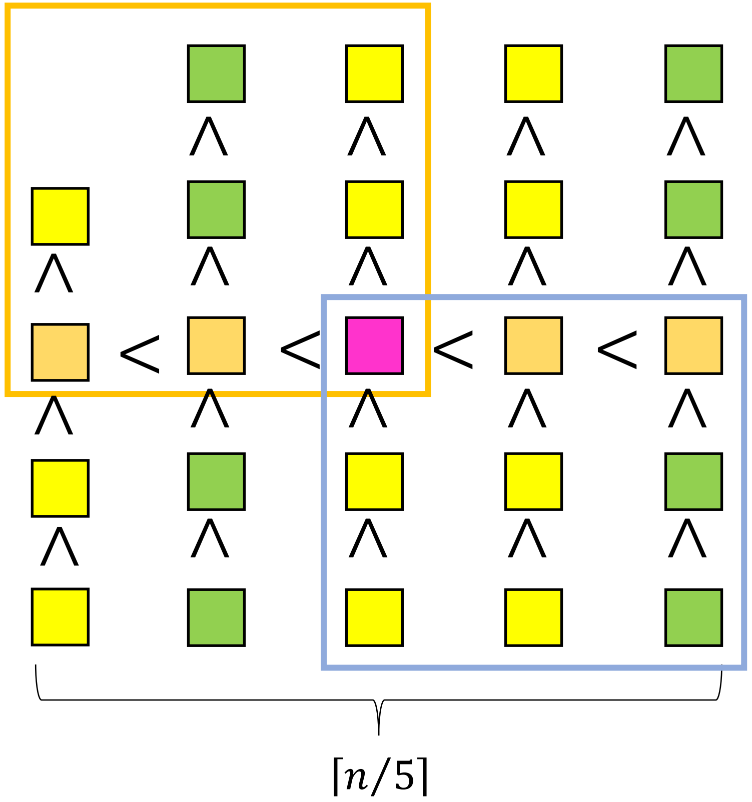
Elements smaller than MedianofMedians: $3\left(\left\lceil\frac{1}{2}\cdot\left\lceil\frac{n}{5}\right\rceil\right\rceil-2\right)\geq\frac{3n}{10}-6$ elements

Number of lists to the "left"

Exclude list on the endpoint, and "middle" list

Median of Medians

MedianofMedians
is larger than all
of these



Elements smaller than MedianofMedians: $3\left(\left\lceil \frac{1}{2} \cdot \left\lceil \frac{n}{5} \right\rceil \right\rceil - 2\right) \geq \frac{3n}{10} - 6$ elements

Elements greater than MedianofMedians: $3\left(\left\lceil \frac{1}{2} \cdot \left\lceil \frac{n}{5} \right\rceil \right\rceil - 2\right) \geq \frac{3n}{10} - 6$ elements

Quickselect

Divide: select an element p using Median of Medians, Partition(p)

$$M(n) + \Theta(n)$$

median of medians algorithm

partition algorithm

Quickselect

Divide: select an element p using Median of Medians, Partition(p)

$$M(n) + \Theta(n)$$

Conquer: if $i = $ index of p, done, if $i <$ index of p recurse left. Else recurse right (with index $i - p$)

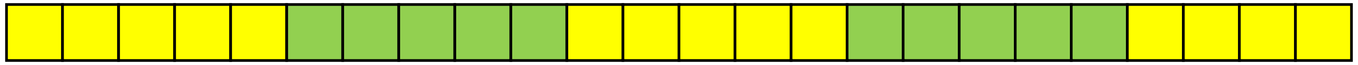
$$\leq S\left(\frac{7n}{10}\right)$$

Combine: Nothing!

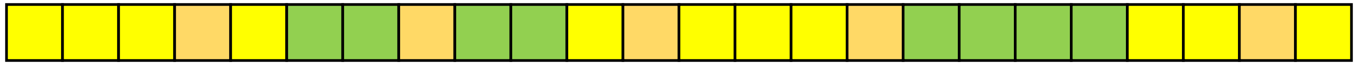
$$S(n) \leq S\left(\frac{7n}{10}\right) + M(n) + \Theta(n)$$

Median of Medians

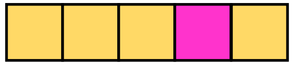
1. Break list into blocks of size 5 $\Theta(n)$



2. Find the median of each chunk $\Theta(n)$



3. Return median of medians (using Quickselect) $S\left(\frac{n}{5}\right)$



$$M(n) = S\left(\frac{n}{5}\right) + \Theta(n)$$

Quickselect

Divide: select an element p using Median of Medians, Partition(p)

$$M(n) + \Theta(n)$$

Conquer: if $i = $ index of p, done, if $i <$ index of p recurse left. Else recurse right

$$\leq S\left(\frac{7n}{10}\right)$$

Combine: Nothing!

$$S(n) \leq S\left(\frac{7n}{10}\right) + M(n) + \Theta(n)$$

Quickselect

Divide: select an element p using Median of Medians, Partition(p)

$$M(n) + \Theta(n)$$

Conquer: if $i = $ index of p, done, if $i <$ index of p recurse left. Else recurse right

$$\leq S\left(\frac{7n}{10}\right)$$

Combine: Nothing!

$$S(n) \leq S\left(\frac{7n}{10}\right) + S\left(\frac{n}{5}\right) + \Theta(n) = \Theta(n)$$

Phew! Back to Quicksort

Divide: Select a pivot element, and partition about the pivot

2	5	1	3	6	4	7	8	10	9	11	12

Using Quickselect, always pivot about the median

2	1	3	5	6	4	7	8	9	10	11	12

Conquer: Recursively sort left and right sublists

If pivot is the median, list is split in half each iteration

Phew! Back to Quicksort

Divide: Select a pivot element, and partition about the pivot

2	5	1	3	6	4	7	8	10	9	11	12

Using Quickselect, always pivot about the median

2	1	3	5	6	4	7	8	9	10	11	12

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

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

A Worthwhile Choice?

Using Quickselect to pick median guarantees $\Theta(n \log n)$ worst-case run-time

Approach has very large constants

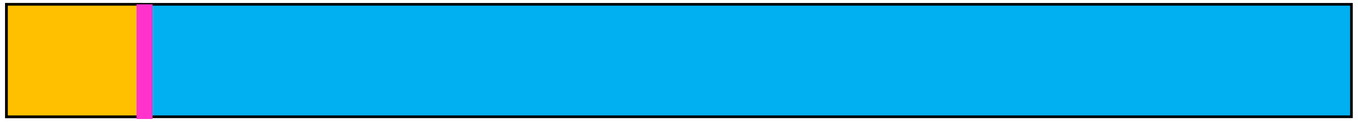
- If you really want $\Theta(n \log n)$, better off using MergeSort

More efficient approach: Random pivot

- Very small constant (very fast algorithm)
- Expected to run in $\Theta(n \log n)$ time
  - Why? Unbalanced partitions are very unlikely

Quicksort Running Time

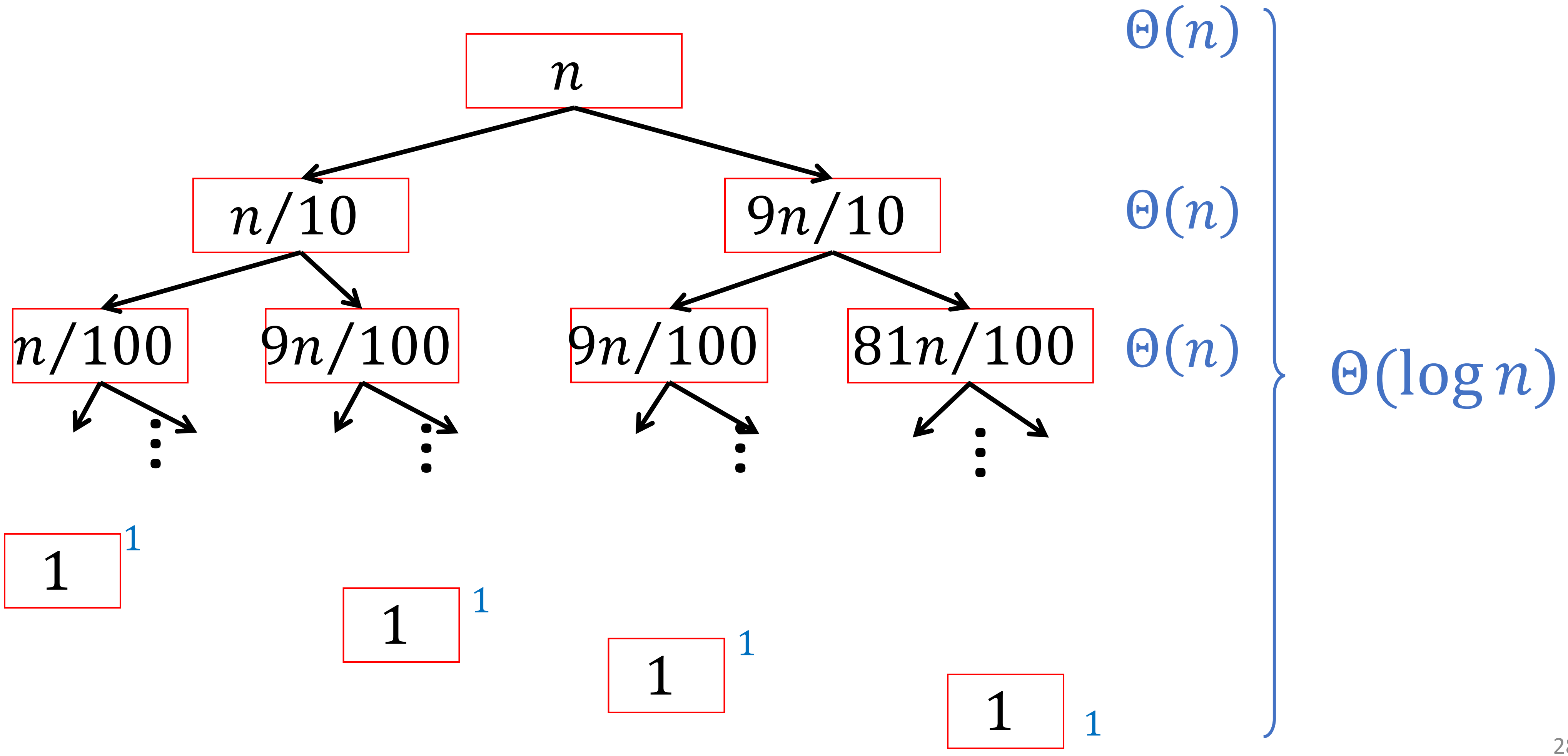
If the pivot is always $(n/10)^{\text{th}}$ order statistic:



$$T(n) = T(n/10) + T(9n/10) + \Theta(n)$$

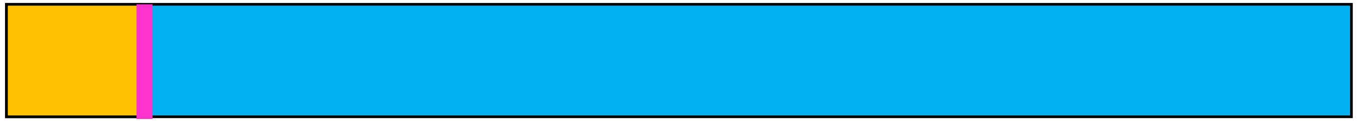
Quicksort Running Time

$$T(n) = T(n/10) + T(9n/10) + \Theta(n)$$



Quicksort Running Time

If the pivot is always $(n/10)^{\text{th}}$ order statistic:



$$T(n) = T(n/10) + T(9n/10) + \Theta(n)$$
$$= \Theta(n \log n)$$

This is true if the pivot is any $(n/k)^{\text{th}}$ order statistic for any constant $k > 1$ (as long as the size of the smaller list is a constant fraction of the full list, we get $\Theta(n \log n)$ running time)

Quicksort Running Time

If the pivot is always d^{th} order statistic:

1	5	2	3	6	4	7	8	10	9	11	12

Then we shorten by d each time

$$T(n) = T(n-d) + n$$

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

What's the probability of this occurring (for a random pivot)?

Probability of Always Choosing d^{th} Order Statistic

We must consistently select pivot from within the first d terms

Probability first pivot is among d smallest: $\frac{d}{n}$

Probability second pivot is among d smallest: $\frac{d}{n-d}$

Probability all pivots are among d smallest:

Very small probability!

$$\frac{d}{n} \times \frac{d}{n-d} \times \frac{d}{n-2d} \times \cdots \times \frac{d}{2d} \times 1 = \left(\frac{n}{d} \times \left(\frac{n}{d} - 1\right) \times \cdots \times 1\right)^{-1} = \frac{1}{\left(\frac{n}{d}\right)!}$$

Formal Argument for $n \log n$ Average

We will focus on counting the number of comparisons

For simplicity: suppose all elements are distinct

Quicksort only compares against a pivot

- Element i only compared to element j if one of them was the pivot

Formal Argument for $n \log n$ Average

What is the probability of comparing two given elements?

1	2	3	4	5	6	7	8	9	10	11	12

Consider the sorted version of the list

Observation: Adjacent elements must be compared

- **Why?** Otherwise I would not know their order
- **Every** sorting algorithm **must** compare adjacent elements

In quicksort: adjacent elements always end up in same sublist, unless one is the pivot

Formal Argument for $n \log n$ Average

What is the probability of comparing two given elements?

1	2	3	4	5	6	7	8	9	10	11	12

Consider the sorted version of the list

$$\Pr[\text{we compare 1 and 12}] = \frac{2}{12}$$

Assuming pivot is chosen uniformly at random

Elements only compared if 1 or 12 was chosen as the first pivot since otherwise they are in <u>different</u> sublists

Formal Argument for $n \log n$ Average

What is the probability of comparing two given elements?

1	2	3	4	5	6	7	8	9	10	11	12
				i				j			

Case 1: Pivot less than i

Then sublist $[i, i+1, \dots, j]$ will be in right sublist and will be processed in future invocation of Quicksort

$$\Pr[\text{we compare } i \text{ and } j] = \Pr[\text{we compare } i \text{ and } j \text{ in Quicksort}([p+1, \dots, n])]$$

Formal Argument for $n \log n$ Average

What is the probability of comparing two given elements?

1	2	3	4	5	6	7	8	9	10	11	12
				i				j			

Case 1: Pivot less than i

Then sublist $[i, i+1, \ldots, j]$ will be processed in future invocation of (

$[p+1, \ldots, n]$ denotes the right sublist (in some order) that we are recursively sorting

$$\Pr[\text{we compare } i \text{ and } j] = \Pr[\text{we compare } i \text{ and } j \text{ in Quicksort}([p+1, \ldots, n])]$$

Formal Argument for $n \log n$ Average

What is the probability of comparing two given elements?

1	2	3	4	5	6	7	8	9	10	11	12
				i				j			

Case 2: Pivot greater than j

Then sublist $[i, i+1, \dots, j]$ will be in left sublist and will be processed in future invocation of Quicksort

$$\Pr[\text{we compare } i \text{ and } j] = \Pr[\text{we compare } i \text{ and } j \text{ in Quicksort}([1, \dots, p])]$$

Formal Argument for $n \log n$ Average

What is the probability of comparing two given elements?

1	2	3	4	5	6	7	8	9	10	11	12
				i				j			

Case 3.1: Pivot contained in $[i+1, \ldots, j-1]$

Then i and j are in different sublists and will never be compared

$$\Pr[\text{we compare } i \text{ and } j] = 0$$

Formal Argument for $n \log n$ Average

What is the probability of comparing two given elements?

1	2	3	4	5	6	7	8	9	10	11	12
				i				j			

Case 3.2: Pivot is either i or j

Then we will always compare i and j

$$\Pr[\text{we compare } i \text{ and } j] = 1$$

Formal Argument for $n \log n$ Average

What is the probability of comparing two given elements?

1	2	3	4	5	6	7	8	9	10	11	12
				i				j			

Case 1: Pivot less than i

$$\Pr[\text{we compare } i \text{ and } j] = \Pr[\text{we compare } i \text{ and } j \text{ in Quicksort}([p+1,\ldots,n])$$

Case 2: Pivot greater than j

$$\Pr[\text{we compare } i \text{ and } j] = \Pr[\text{we compare } i \text{ and } j \text{ in Quicksort}([1,\ldots,p])$$

Case 3: Pivot in $[i, i+1, \ldots, j]$

$$\Pr[\text{we compare } i \text{ and } j] = \Pr[i \text{ or } j \text{ is selected as pivot}] = \frac{2}{j-i+1}$$

Formal Argument for $n \log n$ Average

Probability of comparing element i with element j:

$$\Pr[\text{we compare } i \text{ and } j] = \frac{2}{j - i + 1}$$

Formal Argument for $n \log n$ Average

Probability of comparing element i with element j:

$$\Pr[\text{we compare } i \text{ and } j] = \frac{2}{j - i + 1}$$

<u>Expected</u> number of comparisons:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j - i + 1} = \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} < 2 \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{1}{k} < 2 \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{1}{k}$$

Substitution:
$k = j - i$

$$\frac{1}{k+1} < \frac{1}{k}$$

Formal Argument for $n \log n$ Average

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} = \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} < 2 \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{1}{k} < 2 \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{1}{k}$$

Substitution:
$k = j - i$

$$\frac{1}{k+1} < \frac{1}{k}$$

Useful fact: $\sum_{k=1}^{n} \frac{1}{k} = \Theta(\log n)$

Intuition (not proof!):

$$\sum_{k=1}^{n} \frac{1}{k} \approx \int_{1}^{n} \frac{1}{x} dx = \ln n$$

Formal Argument for $n \log n$ Average

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} = \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} < 2 \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{1}{k} < 2 \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{1}{k}$$

$$= 2 \sum_{i=1}^{n-1} \Theta(\log n) = \Theta(n \log n)$$

Useful fact: $\sum_{k=1}^{n} \frac{1}{k} = \Theta(\log n)$