

# Sorting & Complexity

Hiroki Yasuga, Elisabeth Kolp, Andreas Lang

*25th September 2014, Scientific Programming*

# Agenda

- ▶ What is sorting and complexity?
- ▶ Big O notation
- ▶ Sorting algorithms:
  - Merge sort
  - Quick sort
  - Comparison: Merge sort & Quick sort
- ▶ Searching algorithms
- ▶ Benchmark: Merge sort vs. Quick sort



# Agenda

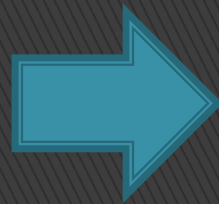
- ▶ What is sorting and complexity?
- ▶ Big O notation
- ▶ Sorting algorithms:
  - Merge sort
  - Quick sort
  - Comparison: Merge sort & Quick sort
- ▶ Searching algorithms
- ▶ Benchmark: Merge sort vs. Quick sort

# What is sorting and complexity?

## Sorting in Kindergarten



Bottle tops



Complexity...  
depends on  
how to sort



# Sorting and complexity in Computer Science

- ▶ Sorting algorithm puts elements of a list in a certain order
- ▶ There are a number of sorting algorithms e.g. Quicksort, Mergesort
- ▶ Choosing efficient sorting algorithm is important
- ▶ Time complexity of sorting algorithm is required time (best, worst and average)
- ▶ To know the complexity, Big O notation

# Agenda

- ▶ What is sorting and complexity?
- ▶ **Big O notation**
- ▶ Sorting algorithms:
  - Merge sort
  - Quick sort
  - Comparison: Merge sort & Quick sort
- ▶ Searching algorithms
- ▶ Benchmark: Merge sort vs. Quick sort



# Big O notation

- ▶ Describes a asymptotic behavior of a function

$$f(x) = O(g(x))$$

$g(x)$  is the dominant term.

$$\text{Ex) } T(x) = 4x^2 - 2x + 2 = O(x^2)$$

- ▶ In time Big-O complexities, this notation indicate how long it takes to complete a sorting

# Agenda

- ▶ What is sorting and complexity?
- ▶ Big O notation
- ▶ Sorting algorithms:
  - Merge sort
  - Quick sort
  - Comparison: Merge sort & Quick sort
- ▶ Searching algorithms
- ▶ Benchmark: Merge sort vs. Quick sort

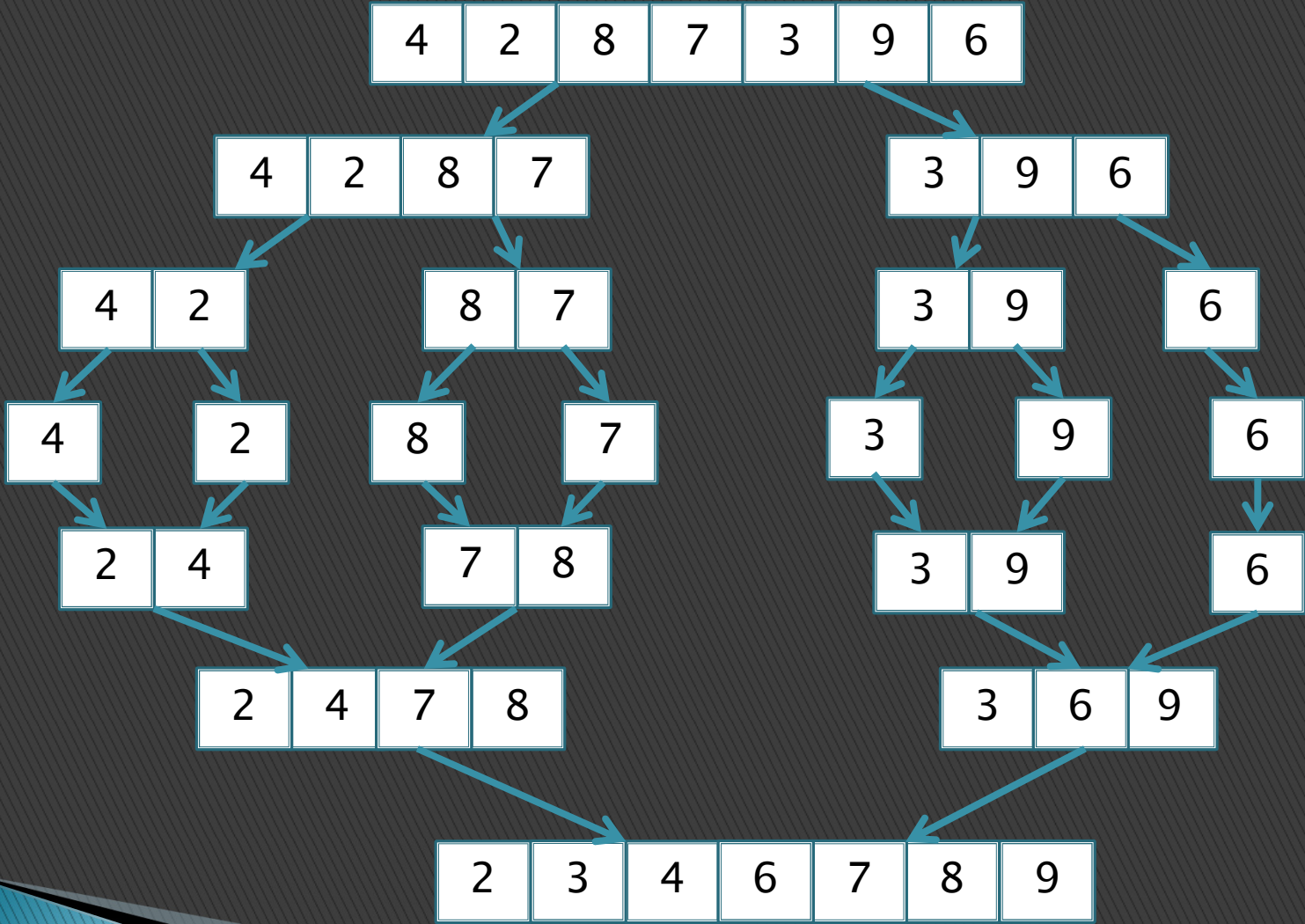


# Merge sort

- ▶ Divide-and-Conquer algorithm
- ▶ Data structure: Array with  $n$  elements
- ▶ Stable sort
- ▶ Complexity:

Best	Average	Worst
$O(n \log(n))$	$O(n \log(n))$	$O(n \log(n))$

# Merge sort: example



Divide-steps

Merge-steps




# Quick sort

- ▶ Divide-and-Conquer algorithm
- ▶ Data structure: Array with  $n$  elements
- ▶ Not a stable sort
- ▶ Complexity:

Best	Average	Worst
$O(n \log(n))$	$O(n \log(n))$	$O(n^2)$

# Quick sort: example

 current pivot

 fixed element

6	2	8	7	3	9	4
---	---	---	---	---	---	---

3	2	4	6	7	9	8
---	---	---	---	---	---	---

2	3	4	6	7	9	8
---	---	---	---	---	---	---

2	3	4	6	7	9	8
---	---	---	---	---	---	---

2	3	4	6	7	9	8
---	---	---	---	---	---	---

2	3	4	6	7	9	8
---	---	---	---	---	---	---

2	3	4	6	7	8	9
---	---	---	---	---	---	---

2	3	4	6	7	8	9
---	---	---	---	---	---	---



# Comparison: Merge sort & Quick sort

## *Similarities:*

- Divide-and-Conquer algorithm
- Recursion
- Same average complexity

## *Differences:*

- Quick sort: sorting step is done during the splitting, no merging step
- Merge sort: stable sort, better worst case complexity

# Agenda

- ▶ What is sorting and complexity?
- ▶ Big O notation
- ▶ Sorting algorithm:
  - Merge sort
  - Quick sort
  - Comparison: Merge sort & Quick sort
- ▶ Searching algorithms
- ▶ Benchmark: Merge sort vs. Quick sort



# Searching algorithms

## Binary Search

- Sorted arrays of  $n$  elements
- Complexity

Average	Worst
$O(\log(n))$	$O(\log(n))$

## Linear (Brute Force Search)

- unsorted & sorted arrays of  $n$  elements
- Complexity

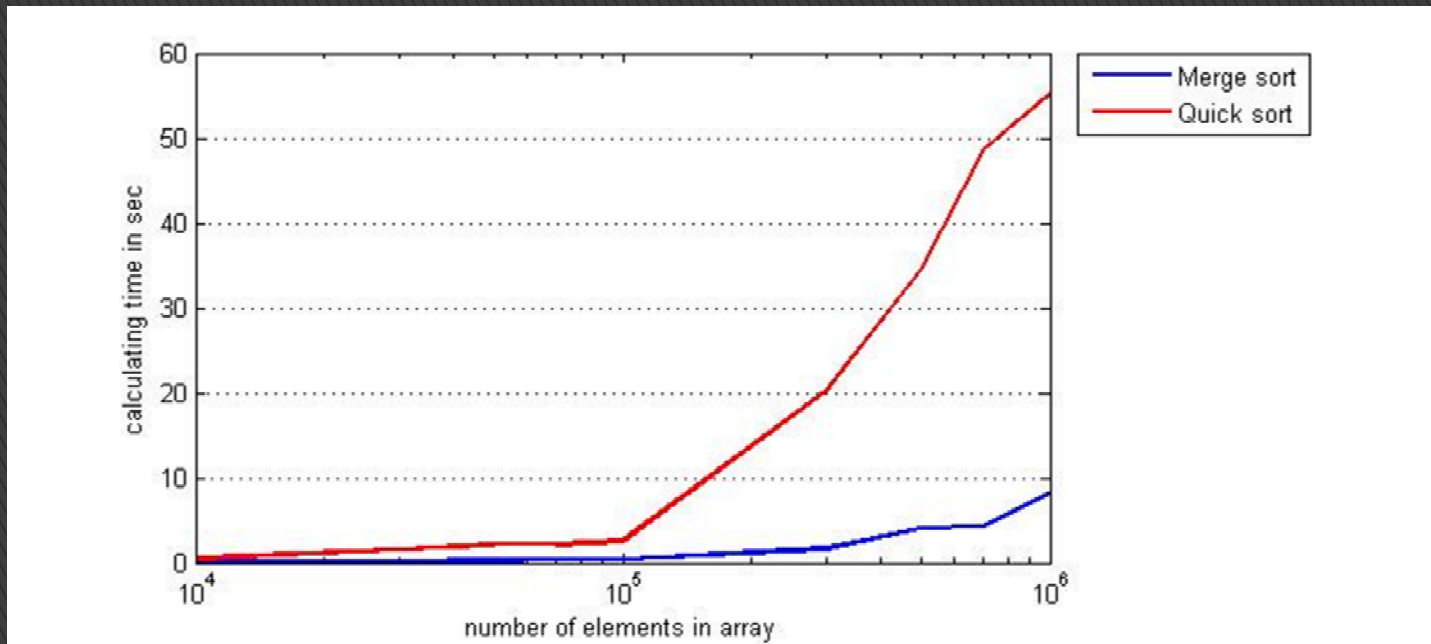
Average	Worst
$O(n)$	$O(n)$

# Agenda

- ▶ What is sorting and complexity?
- ▶ Big O notation
- ▶ Sorting algorithms:
  - Merge sort
  - Quick sort
  - Comparison: Merge sort & Quick sort
- ▶ Searching algorithms
- ▶ Benchmark: Merge sort vs. Quick sort



# Benchmarking: Merge sort vs. Quick sort



# Discussion & Questions

- ▶ How would you sort a terabyte of data?
- ▶ What about parallelization?
- ▶ What if you are running out of RAM?



# Backup: further sorting algorithm

- Heap sort
- Insertion sort
- Bubble sort
- Select sort
- Bucket sort
- Radix sort

# Backup: stable sort

Stable sort: „if two items compare as equal [...] then their relative order will be preserved, so that if one came before the other in the input, it will also come before the other in the output.” ([http://en.wikipedia.org/wiki/Sorting\\_algorithm#Stability](http://en.wikipedia.org/wiki/Sorting_algorithm#Stability))

e.g.: sorting a list of employees twice (first: department, second: lastname)



# Backup: sources

*<http://bigocheatsheet.com/>*

*[http://en.wikipedia.org/wiki/Sorting\\_algorithm](http://en.wikipedia.org/wiki/Sorting_algorithm)*

*<http://www.mathworks.com/matlabcentral/fileexchange/>*

*[http://web.mit.edu/16.070/www/lecture/big\\_o.pdf](http://web.mit.edu/16.070/www/lecture/big_o.pdf)*