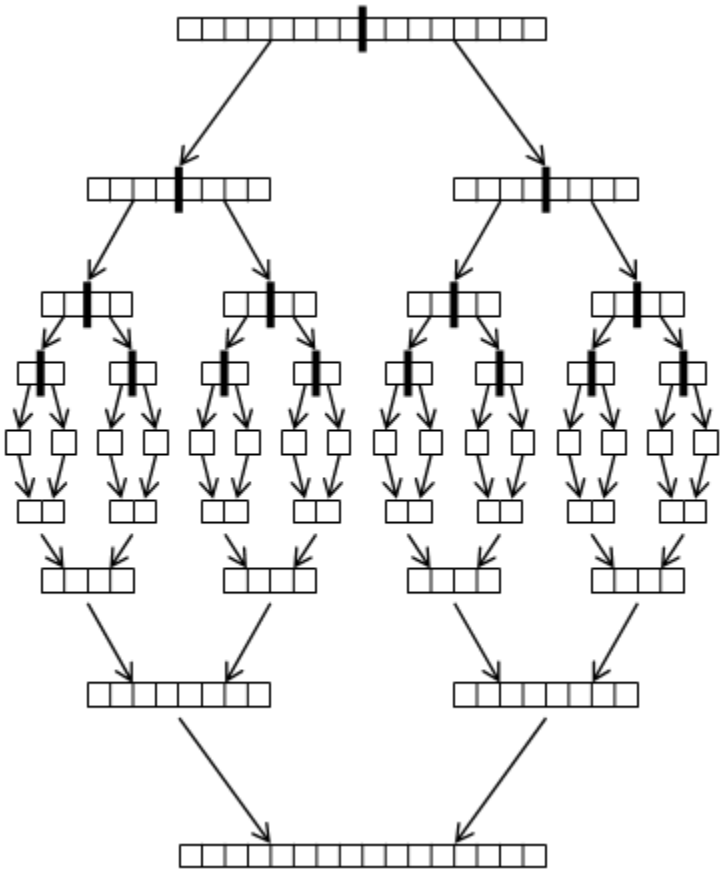
Sorting, Part 2



Agenda

- O(N log N) Algorithms (continued)
- Sub O(N log N) Algorithms
- Summary
- Exercises

O(N log N) Algorithms (continued)

- Quicksort

Quicksort

- Pick a dividing value
- Move values before the divider to the beginning of the array
- Move values after the divider to the end of the array
- Recursively sort the two halves of the array

Quicksort Performance

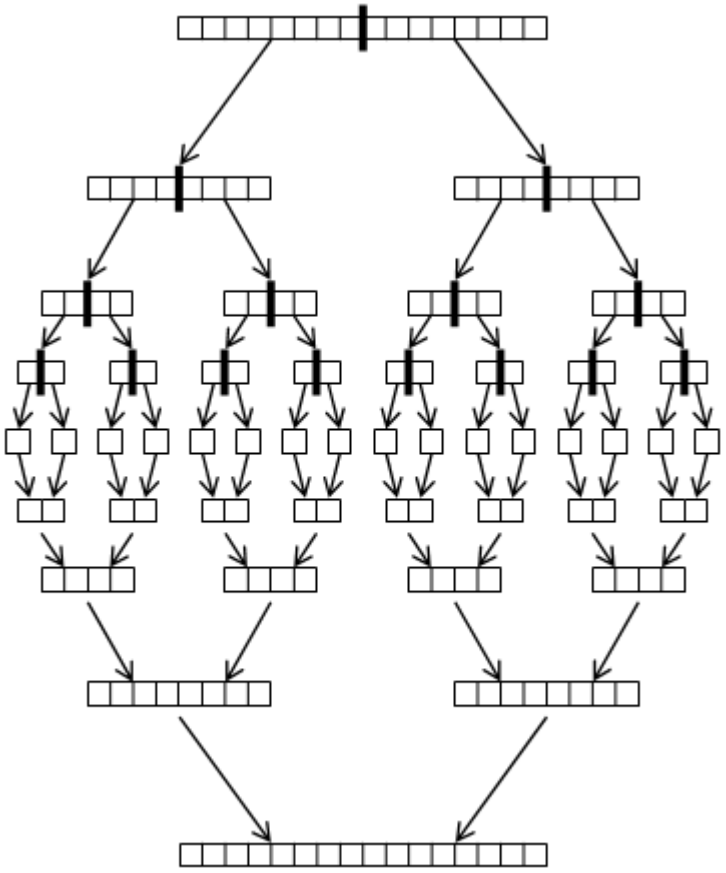
- Normally O(N log N)
- Worst case $O(N^2)$ if:
  - The values are already sorted
  - The values are already sorted in reverse order
  - The values contain many duplicates

Quicksort Implementation

- Stacks
- In place
- Parallelism

Mergesort

- Divide the array into two equally-sized halves
- Recursively sort the halves
- Merge the sorted halves



Stable Sorting

- Maintains the original relative positioning of equivalent values
- Mergesort is stable
- Quicksort is not stable

Sub O(N log N) Algorithms

- Countingsort
- Pigeonhole Sort
- Bucketsort

Countingsort

- Count the number of entries with each value
- Write out the required number of entries with each value

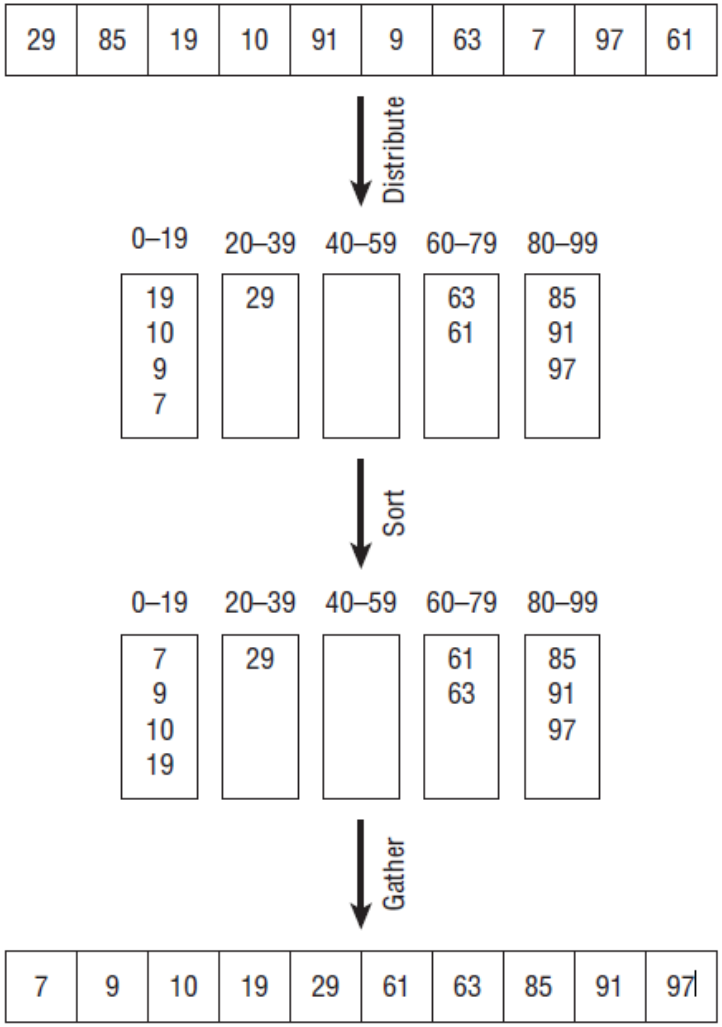
Pigeonhole Sort

- Calculate the pigeonhole where each value belongs
- Sort pigeonholes
- Gather values back into original array

Bucketsort

- Make buckets
- Distribute the items into buckets
- Sort buckets
- Gather values back into original array

Bucketsort


29 85 19 10 91 9 63 7 97 61
Distribute
0–19
20–39
40–59
60–79
80–99
19 10 9 7
29
63 61
85 91 97
Sort
0–19
20–39
40–59
60–79
80–99
7 9 10 19
29
61 63
85 91 97
Gather
7 9 10 19 29 61 63 85 91 97

Algorithm Comparison

ALGORITHM	RUN TIME	TECHNIQUES	WHAT IT IS USEFUL FOR
Insertionsort	$O(N^2)$	Insertion	Very small arrays
Selectionsort	$O(N^2)$	Selection	Very small arrays
Bubblesort	$O(N^2)$	Two-way passes, restricting bounds of interest	Very small arrays, mostly sorted arrays
Heapsort	O(N log N)	Heaps, storing complete trees in an array	Large arrays with unknown distribution
Quicksort	O(N log N) expected, $O(N^2)$ worst case	Divide-and-conquer, swapping items into position, randomization to avoid worst-case behavior	Large arrays without too many duplicates, parallel sorting
Mergesort	O(N log N)	Divide-and-conquer, merging, external sorting	Large arrays with unknown distribution, huge amounts of data, parallel sorting
Countingsort	O(N + M)	Counting	Large arrays of integers with a limited range of values
Bucketsort	O(N + M)	Buckets	Large arrays with reasonably uniform value distribution

Summary

- O(N log N) Algorithms (continued)
  - Quicksort
  - Mergesort
- Sub O(N log N) Algorithms
  - Countingsort
  - Pigeonhole Sort
  - Bucketsort

Exercises

- Chapter 6 Exercises 15 – 21.
- Read *Essential Algorithms, 2e* Chapter 7 pages 201 – 208. (All of Chapter 7.)