

CSCI 3110

Algorithm Efficiency

Measuring the Efficiency of Algorithms

Rates

Keeping You Perspective

Recursion?

Iteration vs.

Recursion

Towers of Hand

Summary

Sorting Algorithms

Selection Sort

Bubble Sort

Mergesor

Quick Sort

Summary

OTL O

Sorting Algorithms



Sorting Algorithms

CSCI 3110

Algorithm

Algorithm Growth

Sortina Algorithms

Summary

- Sorting: A process that organizes a collection of data into either ascending or descending order
- The sort key
 - The part of a data item that we consider when sorting a data collection
 - Typically, the operator < must be defined in the type of kev



Categories of sorting algorithms

CSCI 3110

Algorithm

Algorithm Growth

Sorting Algorithms

Summary

An internal sort

■ Requires that the collection of data fit entirely in the computer's main memory

An external sort

■ The collection of data will not fit in the computer's main memory all at once, but must reside in secondary storage



Sorting

CSCI 3110

Algorithm

Algorithm Growth

- 1 Algorithm Efficiency
 - Measuring the Efficiency of Algorithms
 - Algorithm Growth Rates
 - Keeping Your Perspective
- 2 Recursion?
 - Examples
 - Iteration vs. Recursion
 - One Real Example: The Tow
- 3 Summary
- 4 Sorting Algorithms
- Selection Sort
 - Pubble Sor
 - Bubble So
 - Insertion Sort
 - Mergesor
 - Quick Sort
 - Radix Sor
 - Summar
 - STL Sorting Algorithms

Selection Sort





CSCI 3110

Algorithm

Efficiency

Algorithm Growth

Algorithms

Selection Sort

Bubble Sort

Summary

Strategy

- Place the largest item in its correct place
- Place the next largest item in its correct place, and so on



CSCI 3110

Algorithm

Efficiency

Algorithms
Algorithm Growth

Rates

Keeping You Perspective

Recursio

Examples

Iteration vs

Townson of Uses

C.

Sorting Algorithms

Selection Sort

Bubble Sort

Insertion So

Ouick Sort

Quick Sort

Summar

STL Sorting

Strategy

- Place the largest item in its correct place
- Place the next largest item in its correct place, and so on

Initial array:

29 10 14 37 13



CSCI 3110

- Otrac

Algorithm

Algorithms
Algorithm Growth

Rates

Keeping You Perspective

Recursi

Iteration vs

Recursion

TONGIS OF FIBE

Sorting Algorithms

Selection Sort

Bubble Sort

Insertion So Mergesort

Quick Sor

Radix Sor

STL Sorting

Strategy

- Place the largest item in its correct place
- Place the next largest item in its correct place, and so on

Initial array: 29 10 14 37 13

After 1st swap: 29 10 14 13 **37**



CSCI 3110

Algorithm

Algorithm Growth

Strategy

- Place the largest item in its correct place
- Place the next largest item in its correct place, and so on

Initial array: 29 10 14 37 13

After 1st swap: 29 10 14 13 **37**

After 2nd swap: 13 10 14 **29 37**

Sorting Algorithms Selection Sort Bubble Sort Insertion Sort Mergesort



CSCI 3110

Algorithm

Efficiency of Algorithms

Algorithm Growth Rates

Keeping You Perspective

Recursio

Iteration vs

Recursion Towers of H

Sorting Algorithms

Selection Sort

Bubble Sort

Mergeso

Quick Sort Radix Sort

Summary STL Sorting

Strategy

■ Place the largest item in its correct place

20 10 11 07

Place the next largest item in its correct place, and so on

Initial array:	29	10	14	3/	13
After 1 st swap:	29	10	14	13	37

After 2 nd swap:	13	10	14	29	37



CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Growth

Keeping Your Perspective

D - -----

Iteration vs

Recursion Towers of H

O.....

Sorting Algorithms

Selection Sort Bubble Sort

Insertion :

Quick Sort Radix Sort

Summary STL Sorting

Strategy

- Place the largest item in its correct place
- Place the next largest item in its correct place, and so on

Initial array:	29	10	14	37	13

After 1 st swap:	29	10	14	13	37
-----------------------------	----	----	----	----	----



Implementation & Analysis

CSCI 3110

Algorithm Efficiency

Measuring the

Algorithm Growth

Keeping Your

Perspective

Recursion?

Iteration vs.

Towers of Hand

Towers of Hand

Sorting Algorithms

Selection Sort

Bubble Sort

Merges

Quick Sor

Summary

STL Sorting

Analysis:

■ Worst case: O(n²)

■ Average case: O(n²)

Does not depend on the initial arrangement of the data

Only appropriate for small n



Algorithm Analysis, Recursion, and Sorting

CSCI 3110

- Measuring the Et
 - Measuring the Efficiency of Algorithms
 - Algorithm Growth Rates
 - Keeping Your Perspective
 - 2 Recursion
 - Examples
 - Iteration vs. Recursion
 - One Real Example: The Towers of Hand
 - 3 Summary
 - 4 Sorting Algorithms
 - Selection Sor
 - Bubble Sort
 - Insertion Sort
 - Mergesor
 - Quick Sor
 - Radix Sor
 - Summar
 - STL Sorting Algorithms

Algorithm

Efficiency Measuring th

Algorithms
Algorithm Growth

Keeping You Perspective

Perspective

Examples Iteration vs.

Recursion Towers of H

Sorting

Algorithm Selection So

Bubble Sort

Mergeso

Radix Sort





CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Growth

Keeping Your Perspective

- I eropective

- Toodi olo :

Iteration vs. Recursion

Towers of Har

Sorting Algorithm

Selection Sc

Bubble Sort

Mergesort Quick Sort

Quick Sort Radix Sort Summary STI Sorting

Strategy:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order



CSCI 3110

Algorithm

Efficiency Measuring the

Algorithms
Algorithm Growth

Keeping You Perspective

Perspective

Examples

eration vs.

Towers of Har

Summary

Sorting Algorithm

Bubble Sort

Merges

Quick Sor

Radix Sor

Summary STL Sorting

Strategy:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order

(a) Pass 1

29	10	14	37	13
23	-	ľ	51	כי



CSCI 3110

Algorithm

Algorithm Growth

Bubble Sort

Strategy:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order

(a) Pass 1

29	10	14	37	13
10	29	14	37	13



CSCI 3110

Algorithm

Efficiency Measuring the

Algorithms
Algorithm Growt

Keeping Your Perspective

Perspective

Examples

Iteration vs.

Towers of Har

Summary

Sorting Algorithm

Bubble Sort

Mergeso

Radix Sort Summary

Quick Sort

Strategy:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order

(a) Pass 1

29	10	14	37	13
10	29	14	37	13
10	14	29	37	13



CSCI 3110

Strategy:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order

(a) Pass 1

Initial array:

29	10	14	37	13
40	20	4.4	~-	4.0
10	29	14	3/	13
-		20	0.7	4.0
10	14	29	3/	13
10	14	29	37	13

Algorithm

Measuring the Efficiency of Algorithms

Algorithm Growt Rates

Keeping Your Perspective

Recursion Examples

Towers of Hand

Summary

Sorting Algorithm Selection Sor

Bubble Sort

Mergesor

Quick Sort Radix Sort



CSCI 3110

Algorithm

Algorithms
Algorithm Grow

Keeping Your Perspective

Recursion

Examples

Towers of Han

Summary

Sorting Algorithms

Bubble Sort

Mergesort
Quick Sort
Radix Sort

Quick Sort Radix Sort Summary

Strategy:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order

(a) Pass 1

29	10	14	37	13
10	29	14	37	13
10	14	29	37	13
10	14	29	37	13
10	14	29	13	37



CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Grow

Keeping Your Perspective

Dogumion

Examples

Recursion Towers of Hand

Towers or man

Sorting Algorithm

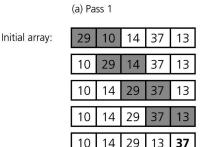
Bubble Sort

Mergesort Quick Sort

Radix Sort
Summary
STL Sorting

Strategy:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order



(b) Pass 2







CSCI 3110

Algorithm

Efficiency of Algorithms Algorithm Growl

Keeping Your Perspective

Recursion

Examples Iteration vs

Towers of Han

Summary

Algorithm: Selection Sor

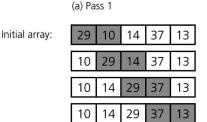
Bubble Sort

Mergesort Quick Sort

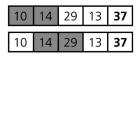
Radix Sort Summary

Strategy:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order



(b) Pass 2





CSCI 3110

Strategy:

Initial array:

 Compare adjacent elements and exchange them if they are out of order

■ Moves the largest elements to the end of the array

Repeating this process eventually sorts the array into ascending order

Algorithm Efficiency

Bubble Sort

(a) Pass 1



4.0	4.4	20	27	40
10	14	29	3/	13
10	14	29	13	37

(b) Pass 2







CSCI 3110

Strategy:

Initial array:

- Compare adjacent elements and exchange them if they are out of order
 - Moves the largest elements to the end of the array
 - Repeating this process eventually sorts the array into ascending order

Algorithm Efficiency Measuring the Efficiency of

Algorithms
Algorithm Growl

Keeping Your Perspective

Recursion?

Recursion

Towers or man

Sorting Algorithms

Bubble Sort

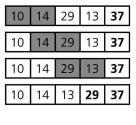
Mergesort
Quick Sort
Radix Sort

Radix Sort Summary STL Sorting

(a) Pass 1



(b) Pass 2



90 Q



Implementation & Analysis

CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Growth

Algorithm Grov Rates

Keeping Yo Perspective

Recursi

Examples

Recursion

Towers of Han

Summary

Sorting Algorithms

Bubble Sort

Mergeson

Quick Sort

Radix Sort

Summary STL Sorting

Analysis:

■ Worst case: O(n²)

■ Average case: O(n²)

■ Best case: O(n)



Algorithm Analysis, Recursion, and Sorting

CSCI 3110

- Algorithm Efficiency
 - Measuring the Efficiency of Algorithm
 - Algorithm Growth Rates
 - Keeping Your Perspective
- 2 Recursion
 - Examples
 - Iteration vs. Recursion
 - One Real Example: The Towers of Hand
- 3 Summary
- 4 Sorting Algorithms
 - Selection Sor
 - Bubble Sor
 - Insertion Sort
 - Mergesor
 - Quick Sor
 - Padiv Sor
 - Summary
 - STL Sorting Algorithms

Algorithm

Efficiency Measuring th

Algorithm Growth Rates

Keeping You Perspective

Examples Iteration vs

Recursion
Towers of Ha

Sorting

Selection Sc

Insertion Sort

Quick Sort

Summary STL Sortin





CSCI 3110

Algorithm

Measuring the Efficiency of

Algorithm Growth

Keeping You Perspective

Recursion?

Iteration vs.

Towers of Har

Summary

Sorting Algorithms

Selection S

Bubble Sort

Insertion Sort Mergesort

Quick Sort Radix Sort

Summary STL Sorting

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region



CSCI 3110

Algorithm

Algorithm Growth

Bubble Sort

Insertion Sort

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region

Initial array:



Copy 10

alf



CSCI 3110

Algorithm

Efficiency

Algorithms
Algorithm Growth

Keeping Your

Perspective

Recursio

Examples Iteration vs.

Towers of Hand

Summary

Sorting Algorithms

Selection S

Bubble Sort Insertion Sort

Mergeso

Radix Sor

Summary STL Sorting

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region

Initial array:

29	10	14	37	13
	V			=

Copy 10

29 29 14 37 13 Shift 29

tself



CSCI 3110

Algorithm

Efficiency Measuring to

Algorithm Growth

Keeping You Perspective

Perspective

Evennoe

Iteration vs. Recursion

Towers of Hand

Sorting

Selection S

Bubble Sort Insertion Sort

Mergeso

Quick Sort

Summar

Summary STL Sorting

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region

Initial array:

3	13	37	14	10	29
3	13	37	14	29	29
	13	37	14	29	29

Copy 10

Shift 29

10 29 14 37 13

Insert 10; copy 14



CSCI 3110

Algorithm

Algorithm Growth

Bubble Sort Insertion Sort

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region

14

Initial array:

29	10	14	37	13	Copy 10
=					
29	29	14	37	13	Shift 29
10	29	14	37	13	Insert 10; copy
•					
10	29	29	37	13	Shift 29

·If



CSCI 3110

Algorithm

Algorithm Growth

Bubble Sort Insertion Sort

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region

Initial array:



10 14 29

Copy 10

Shift 29

Insert 10; copy 14

Shift 29

Insert 14; copy 37, insert 37 on top of itself



CSCI 3110

Algorithm

Algorithm Growth

Bubble Sort Insertion Sort

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region

Initial array:

37 13 14 37 13 13

37 29 29 13

14 29

10 | 14 | 29 | 37

Copy 10

Shift 29

Insert 10: copy 14

Shift 29

Insert 14; copy 37, insert 37 on top of itself

Copy 13



CSCI 3110

Algorithm

Algorithm Growth

Bubble Sort

Insertion Sort

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region

Initial array:

37 13 37 29 29 37 13

13

Insert 10: copy 14

Copy 10

Shift 29

Shift 29

Insert 14; copy 37, insert 37 on top of itself

29 10 | 14 | 29 14 14

14

Copy 13

Shift 37, 29, 14



CSCI 3110

Algorithm

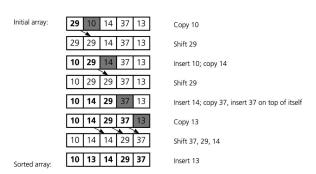
Algorithm Growth

Bubble Sort

Insertion Sort

Strategy:

- Partition the array into two regions: sorted and unsorted
 - Take each item from the unsorted region and insert it into its correct order in the sorted region





Implementation & Analysis

CSCI 3110

Algorithm

Algorithm Growth

Insertion Sort

Mergesort

Summary

■ Analysis:

■ Worst case: $O(n^2)$

Average case: $O(n^2)$

■ Best case: O(n)



Algorithm Analysis, Recursion, and Sorting

CSCI 3110

Algorithm

Algorithm Growth

- Algorithm Efficiency
 - Measuring the Efficiency of Algorithms
 - Algorithm Growth Rates
 - Keeping Your Perspective
- 2 Recursion
 - Examples
 - Iteration vs. Recursion
 - One Real Example: The Towers of Hand
- 3 Summary
- 4 Sorting Algorithms
 - Selection Sor
 - Bubble So
 - Insertion Sort
 - Mergesort
 - Quick Sor
 - Radix Sor
 - Summary
 - STL Sorting Algorithms



Radix Sort Summary







Merge Sort

CSCI 3110

Algorithm

Measuring t Efficiency of

Algorithm Growth

Keeping You Perspective

Pocursion'

1 (CCUI SIOII)

Iteration vs. Recursion

Towers of Har

Summary

Sorting Algorithm

Selection S

Bubble Sort

Mergesort

Quick Sort Radix Sort

Radix Sort Summary STL Sorting

Strategy (Divide-and-conquer):

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array



CSCI 3110

Algorithm

Algorithm Growth

Bubble Sort

Mergesort

Strategy (Divide-and-conquer):

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array

38 16 27 39 12



CSCI 3110

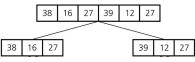
Algorithm

Algorithm Growth

Bubble Sort

Mergesort

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array





CSCI 3110

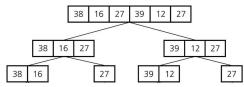
Algorithm

Algorithm Growth

Bubble Sort

Mergesort

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array





CSCI 3110

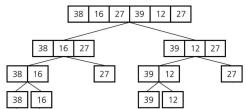
Algorithm

Algorithm Growth

Bubble Sort

Mergesort

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array





CSCI 3110

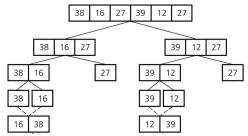
Algorithm

Algorithm Growth

Bubble Sort

Mergesort

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array





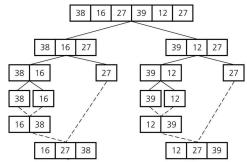
CSCI 3110

Algorithm

Bubble Sort

Mergesort

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array





CSCI 3110

Algorithm

Measuring I

Algorithms
Algorithm Grow

Keeping Your

Perspective

Recursion

Iteration vs. Recursion

Towers of Har

Summa

Sorting Algorithm

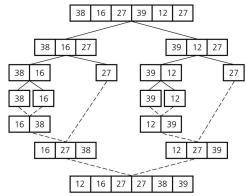
Selection Sc

Bubble Sort

Mergesort

Quick Sort Radix Sort Summary

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array





CSCI 3110

Algorithm Efficiency

Measuring t

Algorithm Grow

Keeping Your

Recursion'

Examples

Recursion

TONGIS OF FIRE

Sorting Algorithm

Algorithm Selection So

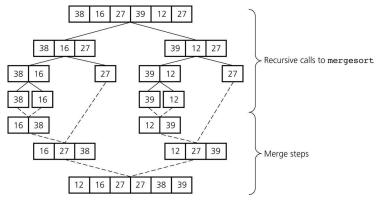
Bubble Sort

Insertion Sor Mergesort

Quick Sort Radix Sort

Summary STL Sorting

- Divide an array into halves
- Sort each half
- Merge the sorted halves into one sorted array





How to Merge Two Arrays

CSCI 3110

Algorithm Efficiency

Efficienc Measuring

Algorithm Grov

Keeping You Perspective

Recursion?

Iteration vs.

Towers of Han

Summary

Algorithms

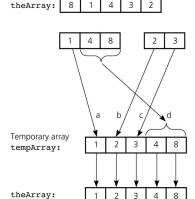
Selection Sc

Bubble Sort

Mergesort

Quick Sort Radix Sort

Summary STL Sortin



Divide the array in half

Sort the halves

Merge the halves:

a. 1 < 2, so move 1 from left half to tempArray

b. 4 > 2, so move 2 from right half to tempArray

c. 4 > 3, so move 3 from right half to tempArray

d. Right half is finished, so move rest of left half to tempArray

Copy temporary array back into original array



Implementation & Analysis

CSCI 3110

Algorithm Efficiency

Algorithms

Rates Keeping Your

Keeping Your Perspective

Recursion?

Iteration vs.

Towers of Hand

Summa

Sorting Algorithm

Selection Sor Bubble Sort

Mergesort

Quick Sort Radix Sort Summary STL Sorting

Analysis:

■ Worst case: $O(n * log_2 n)$

■ Average case: $O(n * log_2 n)$

Performance is independent of the initial order of the array items

Advantage:

Mergesort is an extremely fast algorithm

Disadvantage:

 Mergesort requires a second array as large as the original array



Algorithm Analysis, Recursion, and Sorting

CSCI 3110

- Algorithm Efficiency
 - Measuring the Efficiency of Algorithms
 - Algorithm Growth Rates
 - Keeping Your Perspective
- 2 Recursion
 - Examples
 - Iteration vs. Recursion
 - One Real Example: The Towers of Hand
- 3 Summary
- 4 Sorting Algorithms
 - Selection Sort
 - Bubble So
 - Insertion Sort
 - Merg
 - Quick Sort
 - Radix Sor
 - Summary
 - STL Sorting Algorithms

Algorithm Efficiency

Measuring the Efficiency of

Algorithm Growth Rates

Keeping Your Perspective

Examples

Recursion
Towers of Ha



Algorithn Selection Sc











Quick Sort

CSCI 3110

Algorithm

Algorithm Growth

Quick Sort

Summary

- Choose a pivot
- Partition the array about the pivot
 - left section: items < pivot</p>
 - right section: items > pivot
 - Pivot is now in correct sorted position
- Sort the left and the right section recursively



Quick Sort

CSCI 3110

Algorithm

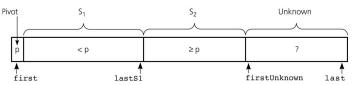
Quick Sort Summarv

Strategy (Divide-and-conquer):

- Choose a pivot
- Partition the array about the pivot
 - left section: items < pivot</p>
 - right section: items > pivot
 - Pivot is now in correct sorted position
- Sort the left and the right section recursively

How to partition an array:

■ Invariant: The items in the region S_1 are all less than the pivot, and those in S_2 are all greater than or equal to the pivot.





CSCI 3110

Algorithm Efficiency

Measuring the Efficiency of Algorithms Algorithm Growth

Keeping Your

Recursion's

teration vs.

Towers of Hanoi

Summary

0 "

Algorithms

Bubble Sort

Insertion Sort

Mergesort

Quick Sort

Summary

STL Sorting





CSCI 3110

Original array:

5 6 7 8 9

Algorithm Efficiency

Efficiency of Algorithms Algorithm Growth

Keeping Your

Recursion

eration vs.

Towers of Hanoi

Summary

Sorting Algorithms

Selection So

Selection So Bubble Sort

Insertion S

Mergesort

Quick Sort

Summar STL Sor

◆□ → ◆□ → ◆ ■ → ● ● 今 へ ○



CSCI 3110

Original array:

Pivot |

9

9

Unknown

Algorithm Growth

Algorithm

Efficiency

Towers of Hanoi

Summary

Sorting Algorithms

Bubble Sort

Mergesort

Quick Sort



CSCI 3110

Original array:

igiriai array.

5 6 7 8 9

Pivot Unknown 5 6 7 8

Pivot S₂ Unknown

9

Perspective

Algorithm Growth

Algorithm

Efficiency

Recursion

Examples teration vs.

_ ...

Towers of Hanoi

Summary

Sorting Algorithms

Selection Sc

Bubble Sort

Insertion S

Mergesort Quick Sort

Radix Sor

Summary STL Sorting S₁ is empty



CSCI 3110

Algorithm

Efficiency

Algorithms
Algorithm Growth

Rates Rates

Keeping You Perspective

Recursion

Examples

Towers of Hanoi

Summary

Sorting Algorithms

Selection Sc

Bubble Sort

Bubble Son

Mergesort

Quick Sort

Radix Sor

Summary STL Sorting

Original array:	5	6	7	8	
	-	_	00 0		•

	Pivot	Unknown			
	5	6	7	8	9
ĺ	Pivot	S ₂	Ur	nknov	vn
	5	6	7	8	9

9

5	6	/	8	9
Pivot	5	S ₂		nowr
5	6	7	Ω	a

S₁ is empty

S₁ is empty



CSCI 3110

Algorithm

Efficiency

Algorithm Growth

Towers of Hanoi

Summary

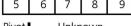
Algorithms

Bubble Sort

Mergesort

Quick Sort

Original array:



DE-BOOKERSON		OTTICE	iown	
5	6	7	8	9

Pivot S ₂		Unknown			
5	6	7	8	9	

Pivot	S ₂		Unknow	
5	6	7	8	9

Pivot		S_2		Unkr	owr
5	6	7	8	9	



CSCI 3110

Algorithm

Algorithm Growth

Towers of Hanoi

Summarv

Algorithms

Bubble Sort

Quick Sort

Summary

Original array: 9 Unknown Pivot I 6 8 9 Pivot Unknown 8 9 6 Pivot Unknown 5 9 6 8 Unknown **Pivot** 6 8 9 Pivot I First partition: 6 8 9

S₁ is empty

S₁ is empty

S₁ is empty

S₁ is empty

4 comparisons, 0 exchanges



Implementation & Analysis

CSCI 3110

Algorithm

Algorithm Growth

Quick Sort

Summary

Analysis:

- Average case: $O(n * \log_2 n)$
- Worst case: $O(n^2)$
 - When the array is already sorted and the smallest item is chosen as the pivot
- Quicksort is usually extremely fast in practice
- Even if the worst case occurs, quicksort's performance is acceptable for moderately large arrays



Algorithm Analysis, Recursion, and Sorting

CSCI 3110

- Sorting Algorithms

 - Radix Sort

Algorithm

Algorithm Growth



Radix Sort

Summary





CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Growth

Keeping Your

Keeping You Perspective

Examples
Iteration vs.

Towers of Han

_

Sorting Algorithm

Selection So Bubble Sort

Mergeso

Quick Sort

Summary STL Sorting

Strategy (Divide-and-conquer):

- Treats each data element as a character string
- Repeatedly organizes the data into groups according to the i^{th} character in each element

Analysis:

■ Average case: O(n)



CSCI 3110

Algorithm Efficiency

Measuring the Algorithms

Algorithm Growth

Keeping Your

Examples

Towers of Hanoi

Summary

Sorting Algorithms

Bubble Sort

Mergesort

Radix Sort

Summary

< □ > < □ > < Ē > < Ē > **₹** 990



CSCI 3110

Algorithm Efficiency

Measuring the Efficiency of Algorithms

Rates

Perspective

Examples

eration vs. Recursion

Towers of Hanoi

Summary

Sorting Algorithms

Selection Sc

Bubble Sort

Mergesort

Out-to O--

Radix Sort

Summary STI Sorting 0123, 2154, 0222, 0004, 0283, 1560, 1061, 2150

Original integers



CSCI 3110

Algorithm Efficiency

Measuring the Efficiency of Algorithms

Rates

Keeping You Perspective

Examples

ration vs. cursion

Towers of Hanoi

Summary

Sorting Algorithms

Selection Sc

Bubble Sort

Mergesort

Quick Sort

Radix Sort

Summary STL Sorting 0123, 2154, 0222, 0004, 0283, 1560, 1061, 2150 Original integers (1560, 2150) (1061) (0222) (0123, 0283) (2154, 0004) Grouped by fourth digit



CSCI 3110

Algorithm Efficiency

Efficiency of Algorithms Algorithm Growth

Rates Keeping Your

Keeping You Perspective

Examples

eration vs. ecursion

Towers of Hanoi

Summary

Algorithms

Selection Sc

Insertion So Mergesort

Quick Sort

Radix Sort

Summary STL Sorting 0123, 2154, 0222, 0004, 0283, 1560, 1061, 2150 Original integers (156**0**, 215**0**) (106**1**) (022**2**) (012**3**, 028**3**) (215**4**, 000**4**) Grouped by fourth digit 1560, 2150, 1061, 0222, 0123, 0283, 2154, 0004 Combined



(0004)

CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Growth

Keeping Your Perspective

Fxamples

ration vs. ecursion

Towers of Hanoi

Summary

Algorithms

Selection Sc

Insertion So

Mergesort

Radix Sort

Summary

0123, 2154, 0222, 0004, 0283, 1560, 1061, 2150 Original integers (156**0**, 215**0**) (106**1**) (022**2**) (012**3**, 028**3**) (215**4**, 000**4**) Grouped by fourth digit 1560, 2150, 1061, 0222, 0123, 0283, 2154, 0004 Combined

(0222, 0123) (2150, 2154) (1560, 1061)

(0283) Grouped by third digit



CSCI 3110

Algorithm Efficiency

Algorithm Growth

Recursion?

Towers of Hanoi Summarv

Algorithms

Mergesort

Radix Sort

0123, 2154, 0222, 0004, 0283, 1560, 1061, 2150 Original integers (1560, 2150) (1061) (0222) (0123, 0283) (2154, 0004) Grouped by fourth digit 1560, 2150, 1061, 0222, 0123, 0283, 2154, 0004 Combined (0004)(02**2**2, 01**2**3) (21**5**0, 21**5**4) (15**6**0, 10**6**1) (02**8**3) Grouped by third digit 0004, 0222, 0123, 2150, 2154, 1560, 1061, 0283 Combined



CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Growth

Keeping Your Perspective

Examples

Iteration vs. Recursion

Towers of Hanoi
Summary

Sorting Algorithms

Selection So

Bubble Sort

Mergesort

Quick Sort

Summary STI Sorting 0123, 2154, 0222, 0004, 0283, 1560, 1061, 2150
Original integers
(1560, 2150) (1061) (0222) (0123, 0283) (2154, 0004) Grouped by fourth digit
1560, 2150, 1061, 0222, 0123, 0283, 2154, 0004
(0004) (0222, 0123) (2150, 2154) (1560, 1061) (0283) Grouped by third digit
0004, 0222, 0123, 2150, 2154, 1560, 1061, 0283
Combined
(0004, 1061) (0123, 2150, 2154) (0222, 0283) (1560) Grouped by second digit



CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Growth

Keeping Your Perspective

Recursion Examples

Iteration vs.
Recursion

Summary

Sorting

Selection Sor

Bubble Sort

Mergesort

Radix Sort

Summary STL Sorting 0123, 2154, 0222, 0004, 0283, 1560, 1061, 2150 Original integers
(1560, 2150) (1061) (0222) (0123, 0283) (2154, 0004) Grouped by fourth digit
1560, 2150, 1061, 0222, 0123, 0283, 2154, 0004 Combined
(0004) (0222, 0123) (2150, 2154) (1560, 1061) (0283) Grouped by third digit
0004, 0222, 0123, 2150, 2154, 1560, 1061, 0283 Combined
(0004, 1061) (0123, 2150, 2154) (0222, 0283) (1560) Grouped by second digit
0004, 1061, 0123, 2150, 2154, 0222, 0283, 1560 Combined



CSCI 3110

Algorithm Efficiency

Efficiency of Algorithms Algorithm Growth

Keeping Your Perspective

Recursion Examples

Iteration vs. Recursion

Summarv

Sorting

Selection Sort

Insertion :

Quick Sort Radix Sort

Radix Sort Summary STL Sorting 0123, 2154, 0222, 0004, 0283, 1560, 1061, 2150
Original integers
(1560, 2150) (1061) (0222) (0123, 0283) (2154, 0004) Grouped by fourth digit
1560, 2150, 1061, 0222, 0123, 0283, 2154, 0004
(0004) (0222, 0123) (2150, 2154) (1560, 1061) (0283) Grouped by third digit
0004, 0222, 0123, 2150, 2154, 1560, 1061, 0283
Combined
(0004, 1061) (0123, 2150, 2154) (0222, 0283) (1560) Grouped by second digit
0004, 1061, 0123, 2150, 2154, 0222, 0283, 1560
Combined

(**0**004, **0**123, **0**222, **0**283) (**1**061, **1**560) (**2**150, **2**154)

Grouped by first digit



Algorithm Analysis, Recursion, and Sorting

CSCI 3110

- Mossuring the Et
 - Measuring the Efficiency of Algorithms
 - Algorithm Growth Rates
 - Keeping Your Perspective
 - 2 Recursion?
 - Examples
 - Iteration vs. Recursion
 - One Real Example: The Towers of Han
 - 3 Summary
 - 4 Sorting Algorithms
 - Selection Sort
 - Bubble So
 - Insertion Sort
 - Mergesort
 - Quick Sor
 - Quick Soi
 - Summary
 - STL Sorting Algorithms

Algorithm Efficiency

Algorithms
Algorithm Growth

Keeping Your

Perspective

Examples Iteration vs.

Recursion Towers of Ha

Sorting

Bubble Sort

Mergeson

Radix Sort

Summary STL Sorting Algorithms





Sorting Algorithm Comparison

CSCI 3110

Algorithm Efficiency

Algorithms
Algorithm Growth

Keeping Your

Perspective

Examples Iteration vs

Iteration vs.
Recursion
Towers of Hanoi

Cummon

Sorting Algorithm

Bubble Sort

Mergesort

Quick Sort Radix Sort

Summary
STL Sorting

	Worst case	Average case
Selection sort	n^2	n^2
Bubble sort	n^2	n^2
Insertion sort	n^2	n^2
Mergesort	$n * \log n$	n * log n
Quicksort	n^2	n * log n
Radix sort	n	n
Treesort rt	n^2	n * log n
	n * log n	n * log n

<u>Click here</u> to get a straightforward feeling of the difference between $O(n^2)$ and $O(n * log_2 n)$.



Algorithm Analysis, Recursion, and Sorting

CSCI 3110

Algorithm

Efficiency

Algorithms
Algorithm Growth

Rates Keeping Your

Keeping You Perspective

Recursio

Iteration vs.

Recursion

TONGIS OF FIBE

Sorting

Algorithms

Bubble Sort

Mergeso

Quick Sort

Radix Sort

Summary STL Sorting Algorithms Algorithm Efficiency

Measuring the Efficiency of Algorithms

Algorithm Growth Rates

Keeping Your Perspective

2 Recursion?

Examples

Iteration vs. Recursion

One Real Example: The Towers of Hano

3 Summary

4 Sorting Algorithms

Selection Sort

Bubble So

Insertion Sort

Mergesort

Quick Sort

Radix Sort

Summary

STL Sorting Algorithms



STL Sorting Algorithms

CSCI 3110

Algorithm

Measuring the Efficiency of

Algorithm Grow Rates

Keeping Your Perspective

Recursion's

Examples

Recursion Towers of Hand

Summ

Sorting Algorithms

Bubble Sort

Insertion So Mergesort

Mergesort
Quick Sort
Radix Sort

Radix Sort Summary STL Sorting Algorithms

Some sort functions in the STL library header (algorithm):

- sort
 - Sorts a range of elements in ascending order by default
- stable_sort
 - Sorts as above, but preserves original ordering of equivalent elements
- partial_sort
 - Sorts a range of elements and places them at the beginning of the range
- nth_element
 - Partitions the elements of a range about the *n*th element
 - The two subranges are not sorted
- partition
 - Partitions the elements of a range according to a given predicate