CS 758/858: Algorithms

Radix Sort

Analysis

More Sorts

http://www.cs.unh.edu/~ruml/cs758

1 handout: slides

D .		
Previous	lv ()n	
I ICVIOUS		

■ O()

■ O() Example

■ Counting Sort

Radix Sort

Analysis

More Sorts

Previously On CS 758...

Wheeler Ruml (UNH)

□ O()

■ O() Example

■ Counting Sort

Radix Sort

Analysis

More Sorts

 $O(g(n)) = \{f(n) : \text{there exist positive constants } c, n_0$ such that $f(n) \le cg(n)$ for all $n \ge n_o\}$

ignore constant factors ignore 'start-up costs' upper bound

We can upper-bound f(except perhaps at start) by scaling g by a constant.

eg, running time of $10n^2 - 5n = O(n^2)$



Wheeler Ruml (UNH)

■ O()

■ O() Example

■ Counting Sort

Radix Sort

Analysis

More Sorts

$$10n^2 + 5n = \Theta(n^2)$$

$$10n \lg \frac{n}{e} = O(n \lg n)$$

Wheeler Ruml (UNH)

Class 2, CS 758 – 4 / 21

■ O()

■ O() Example

Counting Sort

Radix Sort

Analysis

More Sorts

For n numbers in the range 0 to k:

1. for x from 0 to k

2. $\operatorname{count}[x] \leftarrow 0$

3. for each input number x

4. increment count[x]

5. for x from 0 to k

6. do count[x] times

7. emit x

Previously On... O() O() Example Counting Sort Radix Sort Analysis

More Sorts

For n numbers in the range 0 to k:

1. for x from 0 to kO(k)2. count[x] $\leftarrow 0$ O(n)3. for each input number xO(n)4. increment count[x]O(n)5. for x from 0 to kO(k)6. do count[x] timesiterat7. emit xO(1)

O(k) times around loop iterates O(n) times total O(1) each time

 $O(k+n+k+n) = O(2n+2k) = O(n+k) \neq O(n \lg n)$

Radix Sort

■ Stable Counting

■ Radix Sort

Analysis

More Sorts

Radix Sort

Wheeler Ruml (UNH)

Stable Counting Sort

Previously On	Input array contains n records with keys in the range 0 to $k-1$
Radix Sort	
■ Stable Counting	
Radix Sort	
Analysis	
More Sorts	

Radix Sort

Stable Counting

Radix Sort

Analysis

More Sorts

Input array contains n records with keys in the range 0 to k-1

- 1. set count[x] to number of items with key = x
- 2. set pos[x] to total number of keys < x
- 3. for each input record r (in order)
- 4. write r in output array at position pos[key of r]
- 5. increment pos[key of r]

Complexity? Invariants?

Radix Sort

■ Stable Counting

Radix Sort

Analysis

More Sorts

How to sort one million records?

Previously On...

Radix Sort

Stable Counting

Radix Sort

Analysis

More Sorts

How to sort one million records?

How to sort one trillion 4-bit integers?

Previously On...

Radix Sort

Stable Counting

Radix Sort

Analysis

More Sorts

How to sort one million records?

How to sort one trillion 4-bit integers?

How to sort one billion 16-bit integers?

Previously On...

Radix Sort

Stable Counting

Radix Sort

Analysis

More Sorts

How to sort one million records?

How to sort one trillion 4-bit integers?

How to sort one billion 16-bit integers?

How to sort one billion 64-bit integers?

Previously On...

Radix Sort

Stable Counting

Radix Sort

Analysis

More Sorts

How to sort one million records?

How to sort one trillion 4-bit integers?

How to sort one billion 16-bit integers?

How to sort one billion 64-bit integers?

For n numbers with d digits (each digit has k values):

Previously On...

Radix Sort

Stable Counting

Radix Sort

Analysis

More Sorts

How to sort one million records?

How to sort one trillion 4-bit integers?

How to sort one billion 16-bit integers?

How to sort one billion 64-bit integers?

For n numbers with d digits (each digit has k values):

1. for i from 0 to d

2. **stable** sort on digit in place i from right

Previously O	n
--------------	---

Analysis

- Correctness
- Complexity
- Limitations
- Break
- More Sorts

Analysis

Wheeler Ruml (UNH)

Class 2, CS 758 – 10 / 21

Previously On
Radix Sort
Analysis
Correctness
■ Complexity
Limitations
Break
More Sorts

What's the invariant in radix sort?

Complexity

Previously On...

Radix Sort

Analysis

- Correctness
- Complexity
- Limitations
- Break

More Sorts

What's the space complexity? What's the time complexity?

Limitations

Previously On...

Radix Sort

Analysis

Correctness

Complexity

Limitations

Break

More Sorts

Why not implemented more?

Break

Previously On
Radix Sort
Analysis
Correctness
Complexity
Limitations
Break
More Sorts

everyone receiving piazza notifications?

book access?

see book for example proofs

- asst 1: agate, valgrind, submit, happy TA
- no hardcopy submission
- probabilistic grading
- schedule: asst 1, 2, 3

Radix Sort

Analysis

More Sorts

■ Insertion Sort

■ Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

More Sorting Algorithms

Wheeler Ruml (UNH)

Class 2, CS 758 – 15 / 21

Insertion Sort

Previously On...

Radix Sort

Analysis

More Sorts

Insertion SortMerge Sort

■ Quicksort

Partition

Lower Bounds

EOLQs

for i from 2 to nmove A[i] earlier until in place

worse case? best case?

Radix Sort

Analysis

More Sorts

■ Insertion Sort

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

'divide and conquer': divide, combine, and conquer

Mergesort(A, i, j)

- 1. if $i \geq j$, done
- 2. k $\leftarrow (i+j)/2$
- 3. Mergesort(A, i, k)
- 4. Mergesort(A, k + 1, j)
- 5. merge A[i..k] and A[k+1..j] into A[i..j]

how does merge work? running time?

Quicksort

Previously On...

Radix Sort

Analysis

More Sorts

■ Insertion Sort

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

divide, conquer, and combine

Quicksort(A, i, j)1. choose pivot key x

1. Choose proof key x2. partition A[i, j] into A[i, n = 1]

- 2. partition A[i..j] into A[i..p-1] and A[p+1..j]
- 3. if p 1 > i then Quicksort(A, i, p 1)
- 4. if j > p + 1 then Quicksort(A, p + 1, j)

Quicksort

Previously On...

Radix Sort

Analysis

More Sorts

Insertion Sort

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

divide, conquer, and combine

Quicksort
$$(A, i, j)$$

1. choose pivot key x
2. partition $A[i..j]$ into $A[i..p-1]$ and $A[p+1..j]$
3. if $p-1 > i$ then Quicksort $(A, i, p-1)$
4. if $j > p+1$ then Quicksort $(A, p+1, j)$

+:

entirely in-place, no allocation often less copying than merge sort

-:

expected $O(n \lg n)$ needs tricks to avoid worst case

Partition

Previously On...

Radix Sort

Analysis

More Sorts

Insertion SortMerge Sort

Quicksort

Partition

Lower Bounds

EOLQs

Partition(A, i, j)1. choose pivot key p and swap into A[j]2. x = i3. for y = i to j - 14. if $A[y] \le p$ 5. swap A[x] and A[y]6. $x \leftarrow x + 1$ 7. swap A[x] and A[j]

A: (i:) less (x:) greater (y:) unknown (j:) pivot

Previously On	
Radix Sort	
Analysis	
More Sorts	
Insertion Sort	
Merge Sort	
Quicksort	
Partition	
Lower Bounds	
EOLQs	

What is the minimum that a sorting algorithm must do?

Radix Sort

Analysis

More Sorts

 $\blacksquare Insertion Sort$

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items?

Previously On...

Radix Sort

Analysis

More Sorts

 $\blacksquare Insertion Sort$

■ Merge Sort

Quicksort

Partition

Lower Bounds

■ EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items? binary tree with n! leaves

Previously On...

Radix Sort

Analysis

More Sorts

■ Insertion Sort

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items? binary tree with n! leaves has height at least lg(n!)

Previously On...

Radix Sort

Analysis

More Sorts

■ Insertion Sort

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items? binary tree with n! leaves has height at least $\lg(n!)$

Stirling:
$$n! = \sqrt{2\pi n} (\frac{n}{e})^n (1 + \Theta(\frac{1}{n}))$$

SO:

Previously On...

Radix Sort

Analysis

More Sorts

Insertion Sort

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items? binary tree with n! leaves has height at least $\lg(n!)$ Stirling: $n! = \sqrt{2\pi n} (\frac{n}{2})^n (1 + \Theta(\frac{1}{2}))$

$$\lg(n!) = \lg(\sqrt{2\pi n}(\frac{n}{e})^n(1+\Theta(\frac{1}{n})))$$

SO:

Previously On...

Radix Sort

Analysis

More Sorts

Insertion Sort

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items? binary tree with n! leaves has height at least $\lg(n!)$ Stirling: $n! = \sqrt{2\pi n} (\frac{n}{e})^n (1 + \Theta(\frac{1}{n}))$

$$\lg(n!) = \lg(\sqrt{2\pi n}(\frac{n}{e})^n(1+\Theta(\frac{1}{n})))$$
$$= \lg\sqrt{2\pi} + \lg\sqrt{n} + \lg(\frac{n}{e})^n + \lg(1+\Theta(\frac{1}{n}))$$

Wheeler Ruml (UNH)

Previously On...

Radix Sort

Analysis

More Sorts

Insertion Sort

Merge Sort

Quicksort

Partition

Lower Bounds

EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting *n* items? binary tree with *n*! leaves has height at least $\lg(n!)$ Stirling: $n! = \sqrt{2\pi n} (\frac{n}{e})^n (1 + \Theta(\frac{1}{n}))$ so: $\lg(n!) = \lg(\sqrt{2\pi n}(\frac{n}{e})^n (1 + \Theta(\frac{1}{n})))$ $= \lg\sqrt{2\pi} + \lg\sqrt{n} + \lg(\frac{n}{e})^n + \lg(1 + \Theta(\frac{1}{n}))$

 $= \Theta(\lg\sqrt{n} + n\lg(\frac{n}{e})) + lg(1 + \Theta(\frac{1}{n}))$

Wheeler Ruml (UNH)

Class 2, CS 758 - 20 / 21

SO:

Previously On...

Radix Sort

Analysis

More Sorts

■ Insertion Sort

■ Merge Sort

■ Quicksort

Partition

Lower Bounds

■ EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items? binary tree with n! leaves has height at least lg(n!)Stirling: $n! = \sqrt{2\pi n} (\frac{n}{a})^n (1 + \Theta(\frac{1}{n}))$ $\lg(n!) = \lg(\sqrt{2\pi n}(\frac{n}{e})^n(1+\Theta(\frac{1}{e})))$ $= \lg \sqrt{2\pi} + \lg \sqrt{n} + \lg \left(\frac{n}{e}\right)^n + \lg \left(1 + \Theta\left(\frac{1}{n}\right)\right)$ $= \Theta(\lg\sqrt{n} + n\lg(\frac{n}{2})) + lg(1 + \Theta(\frac{1}{n}))$ $= \Theta(\lg\sqrt{n} + n\lg n - n\lg e)) + lg(1 + \Theta(\frac{1}{n}))$

Class 2, CS 758 – 20 / 21

Wheeler Ruml (UNH)

SO:

Previously On...

Radix Sort

Analysis

More Sorts

■ Insertion Sort

■ Merge Sort

■ Quicksort

Partition

Lower Bounds

■ EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items? binary tree with n! leaves has height at least lg(n!)Stirling: $n! = \sqrt{2\pi n} (\frac{n}{a})^n (1 + \Theta(\frac{1}{n}))$ $\lg(n!) = \lg(\sqrt{2\pi n}(\frac{n}{c})^n(1+\Theta(\frac{1}{c})))$ $= \lg \sqrt{2\pi} + \lg \sqrt{n} + \lg \left(\frac{n}{2}\right)^n + \lg \left(1 + \Theta\left(\frac{1}{n}\right)\right)$ $= \Theta(\lg\sqrt{n} + n\lg(\frac{n}{2})) + lg(1 + \Theta(\frac{1}{n}))$ $= \Theta(\lg\sqrt{n} + n\lg n - n\lg e)) + lg(1 + \Theta(\frac{1}{n}))$ $\Theta(n \lg n)$

Class 2, CS 758 – 20 / 21

Wheeler Ruml (UNH)

SO:

Previously On...

Radix Sort

Analysis

More Sorts

■ Insertion Sort

■ Merge Sort

■ Quicksort

Partition

Lower Bounds

■ EOLQs

What is the minimum that a sorting algorithm must do? How many possible outputs are there for sorting n items? binary tree with n! leaves has height at least lg(n!)Stirling: $n! = \sqrt{2\pi n (\frac{n}{c})^n (1 + \Theta(\frac{1}{n}))}$ $\lg(n!) = \lg(\sqrt{2\pi n}(\frac{n}{c})^n(1+\Theta(\frac{1}{c})))$ $= \lg \sqrt{2\pi} + \lg \sqrt{n} + \lg \left(\frac{n}{e}\right)^n + \lg \left(1 + \Theta\left(\frac{1}{n}\right)\right)$ $= \Theta(\lg\sqrt{n} + n\lg(\frac{n}{e})) + lg(1 + \Theta(\frac{1}{n}))$ $= \Theta(\lg\sqrt{n} + n\lg n - n\lg e)) + lg(1 + \Theta(\frac{1}{n}))$ $= \Theta(n \lg n)$

so comparison-based sorting takes $\Omega(n \lg n)$ time

Wheeler Ruml (UNH)

Class 2, CS 758 – 20 / 21

EOLQs

Radix Sort

Analysis

More Sorts

- Insertion Sort
- Merge Sort
- Quicksort
- Partition
- Lower Bounds
- EOLQs

- What's still confusing?
- What question didn't you get to ask today?
 - What would you like to hear more about?

Please write down your most pressing question about algorithms and put it in the box on your way out.

Thanks!