$n \log n$ Lower

Radix Sort

COMP2521 24T3

Sorting Algorithms (IV) Non-Comparison-Based Sorting Algorithms

> Sushmita Ruj cs2521@cse.unsw.edu.au

 $n \log n$ lower bound radix sort

Radix Sort

All of the sorting algorithms so far have been comparison-based sorts.

It can be shown that these algorithms require $\Omega(n \log n)$ comparisons. That is, they require at least $kn \log n$ comparisons for some constant k.

Why?

Radix Sort

Suppose we need to sort 3 items.



Obviously, one comparison is not sufficient to sort them.

Radix Sort

Suppose we need to sort 3 items.

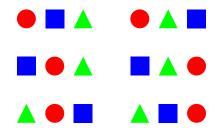


Even two comparisons are not sufficient to sort them. Why?

 $n \log n$ Lower

Radix Sort

If we have 3 items, there are 3! = 6 ways to order them:



Assuming items are unique, one of these permutations is in sorted order.

Suppose we performed the following comparisons:



Four combinations of results are possible: (true, true), (true, false), (false, true), (false, false)

Radix Sort

The two comparisons create four groups, and each permutation of items belongs to one of these groups

 	true	true	false	false
< 🛕	true	false	true	false
		A • •		A •

Radix Sort

Mathematically,

If we have 3 items, then there are 3!=6 ways to order them. In other words, 6 possible permutations.

But if we only perform 2 comparisons, then there are only $2^2=4$ groups, so at least one group will contain more than one permutation.

We need at least 3 comparisons, because this creates $2^3=8$ groups, so each permutation can belong in its own group.

Radix Sort

If we have n items, then there are n! permutations.

If we perform k comparisons, that creates up to 2^k groups.

So given n items, we must perform enough comparisons k such that $2^k \ge n!$

Radix Sort

So given n items, we must perform enough comparisons k such that $2^k > n!$

Taking the
$$\log_2$$
 of both sides gives

$$\log_2 2^k \ge \log_2 n!$$

Since
$$\log_2 2^k = k$$
, we get $k \ge \log_2 n!$

Using Stirling's approximation, we get

$$k \ge n \log_2 n - n \log_2 e + O(\log_2 n)$$

Removing lower-order terms gives

$$k = \Omega(n \log_2 n)$$

Radix Sort

Therefore:

The theoretical lower bound on worst-case execution time for comparison-based sorts is $\Omega(n\log n)$.

Non-Comparison-Based Sorting

 $n \log n$ Lower

Radix Sort

If we aren't limited to just comparing keys, we can achieve better than $O(n \log n)$ worst-case time.

Non-comparison-based sorting algorithms exploit specific properties of the data to sort it.

Radix Sort

Example Analysis Properties

Radix sort is a non-comparison-based sorting algorithm.

It requires us to be able to decompose our keys into individual symbols (digits, characters, bits, etc.), for example:

- The key 372 is decomposed into (3, 7, 2)
- The key "sydney" is decomposed into ('s', 'y', 'd', 'n', 'e', 'y')

Formally, each key k is decomposed into a tuple $(k_1, k_2, k_3, ..., k_m)$.

Radix Sort
Pseudocode
Example
Analysis

Ideally, the range of possible symbols is reasonably small, for example:

• Numeric: 0-9

• Alphabetic: a-z

The number of possible symbols is known as the radix, and is denoted by R.

• Numeric: R = 10 (for base 10)

• Alphabetic: R = 26

If the keys have different lengths, pad them with a suitable symbol, for example:

• Numeric: 123, 015, 007

• Alphabetic: "abc", "zz_", "t___"

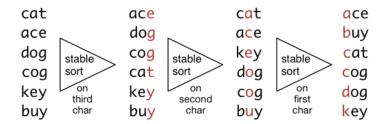
Radix Sort

Example Analysis Properties

Method:

- Perform stable sort on k_m
- Perform stable sort on k_{m-1}
- ...
- Perform stable sort on k_1

Example:



Pseudocode

```
n \log n Lower Bound
```

```
Radix Sort
Pseudocode
Example
```

```
radixSort(A):
    Input: array A of keys where
           each key consists of m symbols from an "alphabet"
    initialise R buckets // one for each symbol
    for i from m down to 1:
        empty all buckets
        for each key in A:
            append key to bucket key[i]
        clear A
        for each bucket (in order):
            for each key in bucket:
                append key to A
```

Example

Radix Sort

Assume alphabet is {'a', 'b', 'c'}, so R = 3.

We want to sort the array:

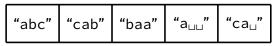
First, pad keys with blank characters:

Each key contains three characters, so m=3.

Pseudocode

Example Analysis

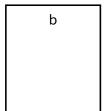
Analysis Properties Array:



Buckets:

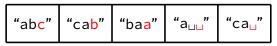
С





Pseudocode Example

Analysis Properties Array:



Buckets:

Ц

а

b

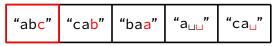
С

Pseudocode

Example Analysis

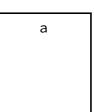
Properties

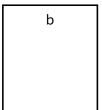
Array:



Buckets:

П





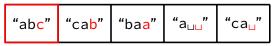
Radix Sort

Pseudocode

Example Analysis

Analysis Properties

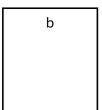
Array:



Buckets:

П



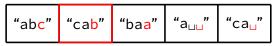


Radix Sort

Pseudocode

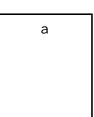
Example Analysis

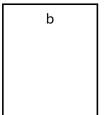
Array:



Buckets:

 \sqcup

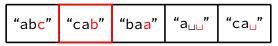




Pseudocode

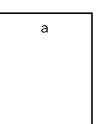
Example Analysis Properties

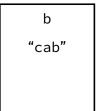
Array:



Buckets:

П

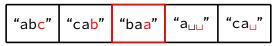




Pseudocode

Example Analysis Properties

Array:



Buckets:

Ц

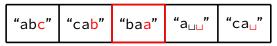




Pseudocode

Example Analysis Properties

Array:



Buckets:

П



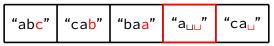


Pseudocode

Example Analysis

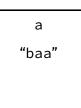
Analysis Properties





Buckets:

Ц



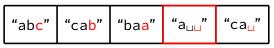


Pseudocode

Analysis

Example





Buckets:

"a_{⊔⊔}"



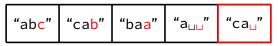


Radix Sort

Pseudocode

Example Analysis

Array:



Buckets:

"a_{⊔⊔}"

а

"baa"

b

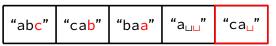
"cab"

Pseudocode

Example

Analysis

Array:



Buckets:

"a_{⊔⊔}"

"ca_□"

а

"baa"

b

"cab"

Example

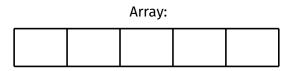
 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis

Properties



Buckets:

L

"a_{⊔⊔}"

"ca⊔

а

"baa"

b

"cab"

C

Radix Sort

Pseudocode

Example Analysis **Properties**

Array:

Buckets:

"a_{⊔⊔}"

а

"baa"

b

"cab"

Example

 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis Properties Array:

"a⊔"

Buckets:

L

"a_{⊔⊔}"

"ca⊔"

а

"baa"

b

"cab"

C

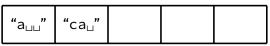
Radix Sort Example

 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis Array:



Buckets:

"a_{⊔⊔}"

"ca_□"

а

"baa"

b

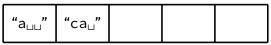
"cab"

Pseudocode

Example Analysis

Radix Sort

Array:



Buckets:

"a_{⊔⊔}"

"ca_□"

а

"baa"

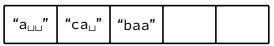
b

"cab"

Radix Sort Pseudocode

Example Analysis

Array:



Buckets:

"a_{⊔⊔}"

"ca_□"

а

"baa"

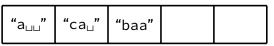
b

"cab"

Radix Sort

Pseudocode

Example Analysis Properties Array:



Buckets:

П

"a_{⊔⊔}"

"ca_□"

а

"baa"

b

"cab"

С

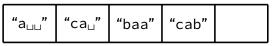
Radix Sort

Pseudocode

Analysis

Example

Array:



Buckets:

"a_{⊔⊔}"

"ca_□"

а

"baa"

b

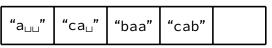
"cab"

Pseudocode

Analysis

Example





Buckets:

"a_{⊔⊔}"

"ca_□"

а

"baa"

b

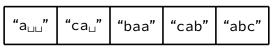
"cab"

Radix Sort

Pseudocode

Example Analysis Properties

Array:



Buckets:

П

"a_{⊔⊔}"

"ca⊔"

а

"baa"

b

"cab"

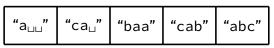
(

Pseudocode

Example

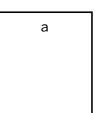
Analysis

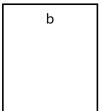
Array:



Buckets:

 \sqcup



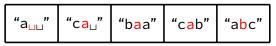


Pseudocode

Analysis

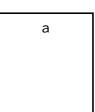
Example

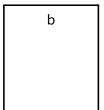
Array:



Buckets:

 \sqcup



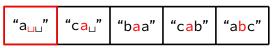


Radix Sort

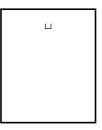
Pseudocode

Example Analysis Properties

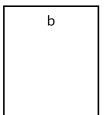


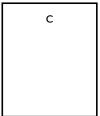


Buckets:









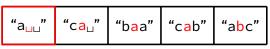
Radix Sort

Pseudocode

Example Analysis

Analysis Properties

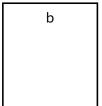




Buckets:

ப "a_{⊔⊔}"

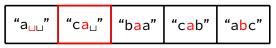




Pseudocode

Example Analysis

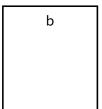
Analysis Properties Array:



Buckets:

⊔ "a_{⊔⊔}"



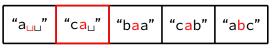


Radix Sort

Pseudocode

Example Analysis

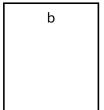




Buckets:

"a⊔⊔



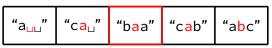


Radix Sort

Pseudocode

Example Analysis





Buckets:

"a⊔⊔

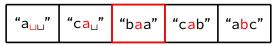




Pseudocode

Example Analysis

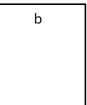
Array:



Buckets:

"a_{⊔⊔}

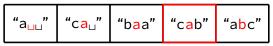
"ca_□" "baa"



Radix Sort

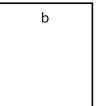
Pseudocode

Example Analysis Properties Array:



Buckets:

⊔ "а_{⊔⊔}" a "ca⊔" "baa"

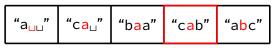


Radix Sort

Pseudocode

Example Analysis





Buckets:

"a⊔⊔

"ca⊔"

"baa"

"cab"

b

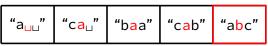
Radix Sort

Pseudocode Example

Analysis Properties

Bound

Array:



Buckets:

Ц

"a_{⊔⊔}"

а

"ca⊔"

"baa"

"cab"

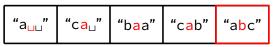
b

Radix Sort

Pseudocode

Example Analysis

Array:



Buckets:

"a⊔⊔

"ca∟"

"baa"

"cab"

b

Radix Sort Example

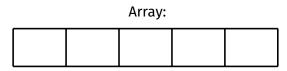
 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis

Properties



Buckets:

Ш

"a_{⊔⊔}"

a "ca⊔" "baa" "cab" b "abc"

C

Radix Sort Example

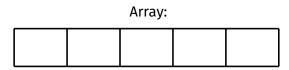
 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example

Analysis **Properties**



Buckets:

"a_{⊔⊔}"

"ca_□" "baa" "cab"

Example

 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis

Array:

"a_{⊔⊔}"

Buckets:

"a_{⊔⊔}"

"ca_□" "baa"

"cab"

b

Radix Sort Example

 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis

Array: "a_{⊔⊔}"

Buckets:

"a_{⊔⊔}"

"ca⊔" "baa"

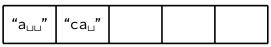
"cab"

b

Radix Sort Pseudocode

Example Analysis

Array:



Buckets:

"a_{⊔⊔}"

"ca⊔" "baa"

"cab"

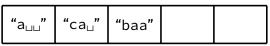
b

Doulla

Radix Sort

Example Analysis

Analysis Properties Array:



Buckets:

П

"a_{⊔⊔}"

а

"ca⊔"

"baa"

"cab"

b

"abc"

C

Radix Sort

Pseudocode

Example Analysis

Array:

"a_{⊔⊔}" "ca_□" "baa" "cab"

Buckets:

"a_{⊔⊔}"

"ca⊔"

"baa"

"cab"

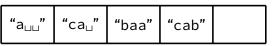
b

Radix Sort

Pseudocode

Example Analysis

Array:



Buckets:

"a_{⊔⊔}"

"ca⊔"

"baa"

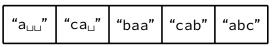
"cab"

b

Pseudocode

Example Analysis

Analysis Properties Array:



Buckets:

П

"a_{⊔⊔}"

а

"ca⊔"

"baa"

"cab"

b

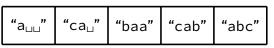
"abc"

Pseudocode

Example

Analysis

Array:



Buckets:

"a_{⊔⊔}

"ca⊔"

"baa"

"cab"

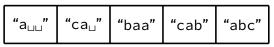
b

"abc"

Pseudocode

Example Analysis

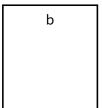
Analysis Properties Array:



Buckets:

П



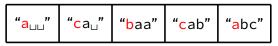


Radix Sort

Pseudocode

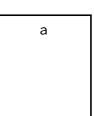
Example Analysis

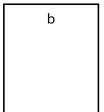
Array:



Buckets:

 \sqcup



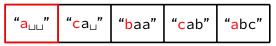




Pseudocode

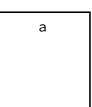
Example Analysis

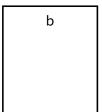
Analysis Properties Array:



Buckets:

П



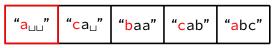


Radix Sort

Pseudocode

Example Analysis

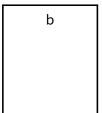


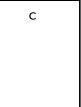


Buckets:

 \sqcup





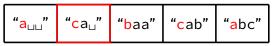


Radix Sort

Pseudocode

Example Analysis

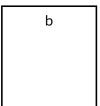




Buckets:

 \sqcup



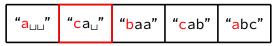


Pseudocode

Example Analysis

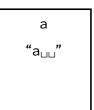
Analysis Properties

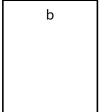
Array:



Buckets:

П





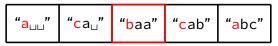
c "ca⊔"

Pseudocode

Example Analysis

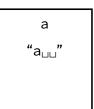
Properties

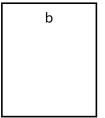
Array:

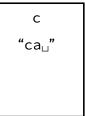


Buckets:





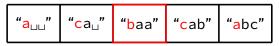




Pseudocode

Example Analysis Properties

Array:



Buckets:

Ц





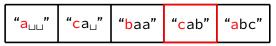
c "ca⊔"

Pseudocode

Example Analysis

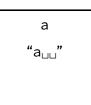
Analysis Properties

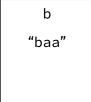
Array:



Buckets:

П





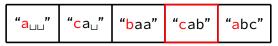
c "ca⊔"

Pseudocode

Example

Analysis

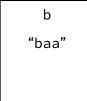
Array:



Buckets:

Ш





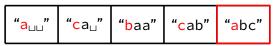
"ca_□" "cab"

Pseudocode

Example Analysis

Analysis Properties

Array:



Buckets:

Ц



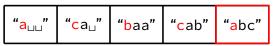


c "ca⊔" "cab"

Radix Sort Pseudocode

Example Analysis

Array:



Buckets:

Ш





Radix Sort Example

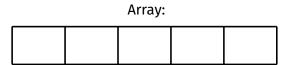
 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis

Analysis Properties



Buckets:

Ш



b "baa"

"ca⊔" "cab"

Radix Sort Example

 $n \log n$ Lower Bound

Radix Sort

Analysis **Properties**

Pseudocode Example

Array:

Buckets:

 \sqcup

"a_{□□}" "abc"

b

"baa"

Example

 $n \log n$ Lower Bound

Radix Sort

Pseudocode Example

Analysis **Properties** Array:



Buckets:

 \sqcup

"a_{□□}" "abc"

b

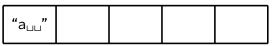
"baa"

Radix Sort

Pseudocode

Example Analysis





Buckets:

 \sqcup

"a_{□□}" "abc"

b "baa"

Radix Sort Example

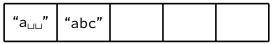
 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis

Array:



Buckets:

Ш

"a_{□□}" "abc"

b

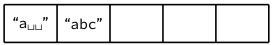
"baa"

Pseudocode

Analysis

Example

Array:



Buckets:

Ш

"a_{□□}" "abc"

b "baa"

Example

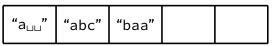
 $n \log n$ Lower Bound

Radix Sort

Pseudocode

Example Analysis

Array:



Buckets:

 \sqcup

"a_{□□}" "abc"

"baa"

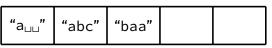
b

Pseudocode

Example Analysis

Analysis Properties

Array:



Buckets:

П

a "a_{⊔⊔}" "abc"

b "baa"

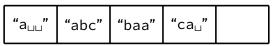
Radix Sort

Pseudocode

Example

Analysis

Array:



Buckets:

 \sqcup

"a_{□□}" "abc"

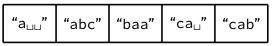
b "baa"

"ca⊔" "cab"

Pseudocode

Example Analysis

Analysis Properties Array:



Buckets:

Ш

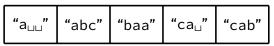
a "a_{⊔⊔}" "abc"

b "baa"

Pseudocode

Example Analysis

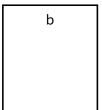
Analysis Properties Array:



Buckets:

П





Analysis:

- Array contains n keys
- Each key contains *m* symbols
- Radix sort uses R buckets
- A single stable sort runs in time O(n+R)
- Radix sort uses stable sort m times

Hence, time complexity for radix sort is O(m(n+R)).

• $\approx O(mn)$, assuming R is small

Therefore, radix sort performs better than comparison-based sorting algorithms:

• When keys are short (i.e., m is small) and arrays are large (i.e., n is large)

Radix Sort Properties

 $n \log n$ Lower Bound

Radix Sort

Example Analysis

Properties

Stable

All sub-sorts performed are stable

Non-adaptive

Same steps performed, regardless of sortedness

Not in-place

Uses O(R + n) additional space for buckets and storing keys in buckets

Other Non-Comparison-Based Sorts

 $n \log n$ Lower Bound

Radix Sort

Pseudocode Example

Properties

- Bucket sort
- MSD Radix Sort
 - The version shown was LSD
- Key-indexed counting sort
- · ...and others

Radix Sort

Pseudocode Example

Analysis Properties https://forms.office.com/r/zEqxUXvmLR

