Reminder from Monday

Anything worth finding is worth sorting

“Radix Sort”

Best way to sort physical objects
• “algorithm” used to sort punched cards by hand or by machine since 1890
Basic step – sort a place

Move numbers, item by item, into bins
  • each bin gets a specific value
Assemble one stack from each bin
  • taking bins in order of value

Example

Numbers:
  101, 220, 012, 201, 210, 112, 002, 020, 120, 211
Example

Numbers:
101, 220, 012, 201, 210, 112, 002, 020, 120, 211

Sort by lowest digit first, into 3 bins:
0 220, 210, 020, 120
1 101, 201, 211
2 012, 112, 002

Example

Sort by lowest digit first, into 3 bins:
0 220, 210, 020, 120
1 101, 201, 211
2 012, 112, 002

Assemble overall list:
220, 210, 020, 120, 101, 201, 211, 012, 112, 002
Radix sort algorithm

Apply basic step in turn to each digit place

• In order from smallest to largest
• Use output from last step as input to next step

We’ve done step one – let’s continue.

Next step – place 2

Numbers:

220, 210, 020, 120, 101, 201, 211, 012, 112, 002
Next step – place 2

Numbers:

220, 210, 020, 120, 101, 201, 211, 012, 112, 002

Sort by 2\textsuperscript{nd} digit into 3 bins:

0 101, 201, 002
1 210, 211, 012, 112
2 220, 020, 120

Assemble overall list:

101, 201, 002, 210, 211, 012, 112, 220, 020, 120
Final step – place 3

Numbers:
101, 201, 002, 210, 211, 012, 112, 220, 020, 120

Sort by 3rd digit into 3 bins:
0  002, 012, 020
1  101, 112, 120
2  201, 210, 211, 220
**Final step – place 3**

Sort by 3rd digit into 3 bins:

- 0 002, 012, 020
- 1 101, 112, 120
- 2 201, 210, 211, 220

Assemble final list:

002, 012, 020, 101, 112, 120, 201, 210, 211, 220

---

**Radix sort - properties**

How many operations does each step involve, if there are N cards?

A. 1 – it happens in one step
B. N – one per card
C. N² – go through all the cards per card
Radix sort - properties

How many steps of bucketing and stacking cards will we need, to sort D digit numbers

A. 1 – it happens in one step
B. D – one per digit
C. $D^2$ – go through all the digits per digit

Radix sort - properties

Each step requires us to look at each card once.
One step for each digit.

With N cards and D digits, that's ND operations.

- For “keys”, D is about log N.
  Each number occurs once.
- Otherwise, D could be somewhat more.
Radix sort – example

Sorting numbers in the millions on punched cards.
  • Seven digit numbers.
  • So run the basic step 7 times.
  • 7N operations for N cards.

Thinking some more

Radix sort is good for cards.
  • Sorting cards involves moving them around.
    Easy to move cards around in stacks.

Not so good in computer memory.
  • Sorting involves copying data.
    Each data item must be copied individually.
**Sorting “in place”**

Fastest computer algorithms avoid copying data
- unless it’s specifically out of order

---

**Need one idea first though**

“Recursive” sorting
- Part of the algorithm for sorting a list can be sorting a smaller list
- It works out because eventually you get down to a list of length 1 – which is always sorted
Simplest recursive sort

Mergesort!

- If the list has 1 element you’re done.
- Otherwise cut the list in half.
- Sort each half.
- Scan through the two sorted halves in order to make resulting sorted list

Example

Numbers:

101, 220, 012, 201, 210, 112, 002, 020, 120, 211

Cut in half:

101, 220, 012, 201, 210
112, 002, 020, 120, 211
**Example**

Cut in half:

101, 220, 012, 201, 210
112, 002, 020, 120, 211

Sort each half (we'll get back to this)

012, 101, 201, 210, 220
002, 020, 112, 120, 211

**Example**

Go through halves in order to make sorted list:

A: 012, 101, 201, 210, 220
B: 002, 020, 112, 120, 211

Result:

002, 012, 020, 101, 112, 120, 201, 210, 211, 220
Pseudocode for merge:

set i to 1 and j to 1 and k to 1
repeat until i > length A or j > length B
  • if item i of A > item j of B
    • set item k of R to item j of B; change j by 1
  • else
    • set item k of R to item i of A; change i by 1
    • change k by 1
repeat until i > length A
  • set item k of R to item i of A; change i and k by 1
repeat until j > length B
  • set item k of R to item j of B; change j and k by 1

How long to do?

If there are N items in list A and M items in list B, how long does this take in all?

A. N
B. M
C. N+M
D. NM
E. N/M