CS 187: Programming with Data Structures (Spring 2010)

Lecture 18: Radix Sort

Rui Wang
Comparison Based Sorting

• How fast can you sort?
• Quadratic $O(N^2)$:
  – Bubble, insertion, selection
• Log-Linear $O(N\times\log N)$:
  – Merge sort, quick sort
• Can you do any better?
Comparison Based Sorting

- How fast can you sort?
- Quadratic $O(N^2)$:
  - Bubble, insertion, selection
- Log-Linear $O(N \log N)$:
  - Merge sort, quick sort
- Can you do any better?
- It can be proved that $O(N \log N)$ is the lower bound for any comparison based sorting!
Non-comparison-based Sorting

• There are also sorting algorithms *not* based on comparison:
  – Counting sort
  – Bucket sort
  – Radix sort

• These algorithms have a lower cost, but they make specific assumptions.
Counting Sort

• **Assumption**: the range of the elements is known and is bounded.

• Example: 1, 3, 3, 5, 4, 2, 1, 4, 5
Each element is taken from \{1,2,3,4,5\}

• How do we sort this?
• What’s the cost?
Counting Sort

• **Assumption**: the range of the elements is known and is bounded.

• Example: 1, 3, 3, 5, 4, 2, 1, 4, 5
  Each element is taken from \{1,2,3,4,5\}

• How do we sort this?
• What’s the cost? \(\rightarrow O(N+k)\)
  (k is the range of the element values).
Counting Sort

• **Assumption:** the range of the elements is known and is bounded.

• Why can this be faster than $O(N \log N)$?
  
  It is not comparison based: it relies on the fact that each element is taken from a bounded range of discrete values. Thus for each possible value, you only need to count how many there are.
Radix Sort

• Similarly, non-comparison-based.
• What does radix mean?
Radix Sort

• Similarly, non-comparison-based.
• What does **radix** mean?

The base of a system of numbers. For example: 10 is the radix of decimal numbers, and 2 is the radix of binary numbers.
Counting Sort

• **Stable sorting**: preserves the relative order of equal elements.

  • Example: 1, 3, 3, 5, 4, 2, 1, 4, 5
    \[\rightarrow 1, 1, 2, 3, 3, 4, 4, 5, 5\]

• Counting sort is a stable sorting algorithm.

• What other sorting algorithms are stable?
  – Bubble? Merge? Quick?
Radix Sort

- Similarly, non-comparison-based.
- Radix sort involves checking the digit of each element, starting from the rightmost (least significant) digit.

Digit: 4 1 3 5 4 5 3 1 4 7

Most significant digit  Least significant digit
Radix Sort

• Start by ordering the elements using their rightmost digit.
• Then sort the next digit on the left, and so on.
• Example:
  sort the following decimal numbers
  329, 457, 657, 65, 839, 436, 720, 355
Radix Sort

• To sort each digit, we can use counting sort, because each digit must be between $[0,9]$.  

• Why does radix sort work?
Radix Sort

• To sort each digit, we can use counting sort, because each digit must be between [0,9].

• **Why does radix sort work?**
  Because sorting of each digit is **stable**: preserves the relative order of equal elements. This is crucial.
Radix Sort

• Working in radix 10 is not very efficient (each digit has 10 possibilities.)

• Working in radix 2 is much better, because each digit can only be 0 or 1.
  – This is easy since we already represent numbers as binaries on a computer.
  – How does this simplify the sorting of each digit?
Radix Sort

• What’s the cost of radix sort?
Radix Sort

• What’s the cost of radix sort?
• Sorting each digit takes $O(N+k)$ time using counting sort. If we work with binary numbers, $k=2$, and we can ignore it.
• If each number has $d$ digits, then we need $d$ passes. So the overall cost is $O(d*N)$.
• As long as $d$ is a small number, this cost is essentially linear to $N$. 
Radix Sort

• What’s the cost of radix sort?
  \[O(d \times N)\]

• However, this is not generally true. Let’s say, you have \(N\) elements, and the range of these elements can go all the way from 1 to \(N\). How many digits are there?
Radix Sort

• What’s the cost of radix sort?
  \( O(d*N) \)

• However, this is not generally true. Let’s say, you have \( N \) elements, and the range of these elements can go all the way from 1 to \( N \). How many digits are there? \( \rightarrow d=\log N \)

• Hence the overall cost in this case is still \( O(N*\log N) \).
Radix Sort

• Still, radix sort is a very attractive algorithm, because it’s very fast, and its performance is predictable.
  – Integers are typically represented as 32-bit binary.

• Does it work on floating point elements?
Summary

• We learned three advanced sorting algorithms:
  1. Mergesort: $O(N \times \log N)$
  2. Quicksort: $O(N \times \log N)$
  3. Radix Sort: $O(d \times N)$