6.006 Introduction to Algorithms
Spring 2008

Lecture 11: Sorting IV: Stable Sorting, Radix Sort

Lecture Overview

- Stable Sorting
- Radix Sort
- Quick Sort ← not officially a part of 6.006
- Sorting Races

Stable Sorting

Preserves input order among equal elements

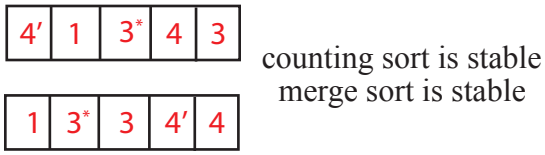

Figure 1: Stability

Selection Sort and Heap: Find maximum element and put it at end of array (swap with element at end of array) NOT STABLE!

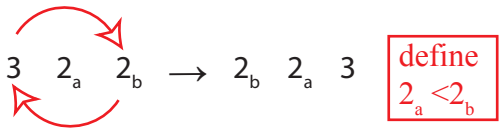

Figure 2: Selection Sort Instability

Radix Sort

- Herman Hollerith card-sorting machine for 1890 census.
- Digit by Digit sort by mechanical machine
  1. Examine given column of each card in a deck
  2. Distribute the card into one of 10 bins

3. Gather cards bin by bin, so cards with first place punched are on top of cards with second place punched, etc.

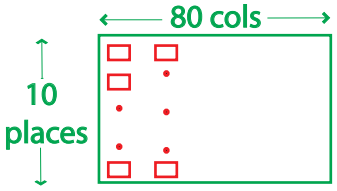

Figure 3: Punch Card

MSB vs. LSB?

Sort on most significant digit first or least significant digit first?

MSB strategy: Cards in 9 of 10 bins must be put aside, leading to a large number of intermediate piles

LSB strategy: Can gather sorted cards in bins appropriately to create a deck!

Example

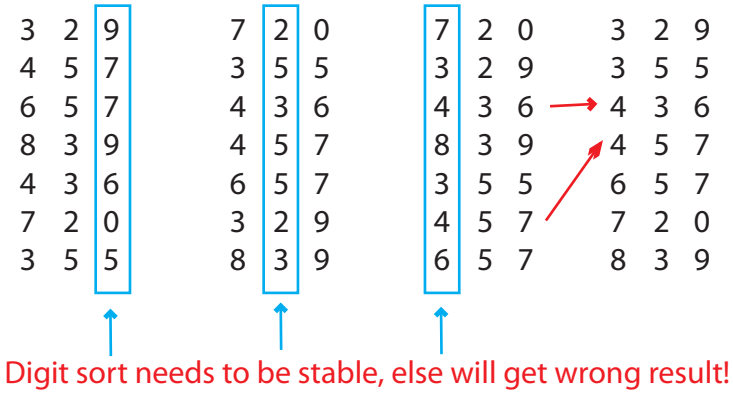

Figure 4: Example of Radix Sort

Analysis

Assume counting sort is auxiliary stable sort. $\Theta(n+k)$ complexity.

Suppose we have n words of b bits each.

One pass of counting sort	$\Theta(n+2^b)$		
b passes of counting sort	$\Theta(b(n+2)) = \Theta(nb)$		
$\frac{b}{r}$ passes	$\Theta(\frac{b}{r}(n+2^r))$	minimized when $r = \lg n$	$\Theta(\frac{bn}{\lg n})$

Quick Sort

This section is for "enrichment" only.

Divide: Partition the array into two. Sub-arrays around a pivot x such that elements in lower sub array $\leq x \leq$ elements in upper sub array. ← Linear Time

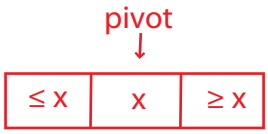

Figure 5: Pivot Definition

Conquer: Recursively sort the two sub arrays

Combine: Trivial

If we choose a pivot such that two sub arrays are roughly equal:

$$T(n) = 2T(n/2) + \Theta(n) \implies T(n) = \Theta(n \lg n)$$

If one array is much bigger:

$$T(n) = T(n-1) + \Theta(n) \implies T(n) = \Theta(n^2)$$

Average case $\Theta(n \lg n)$ assuming input array is randomized!

Sorting Races

Click here for a reference on this.

Bubble Sort: Repeatedly step through the list to be sorted. Compare 2 items, swap if they are in the wrong order. Continue through list, until no swaps. Repeat pass through list until no swaps. $\Theta(n^2)$

Shell Sort: Improves insertion sort by comparing elements separated by gaps $\Theta(n\lg^2 n)$