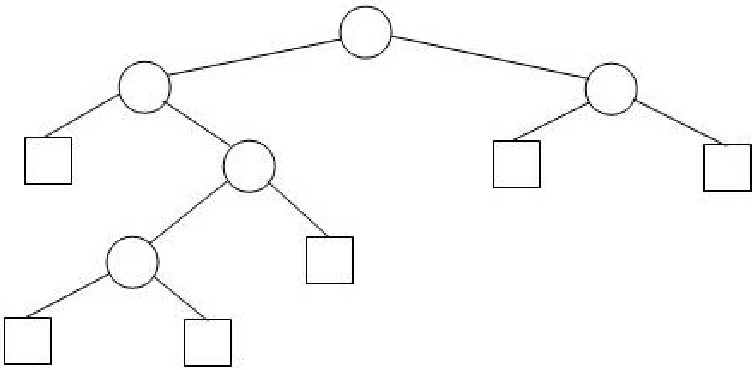
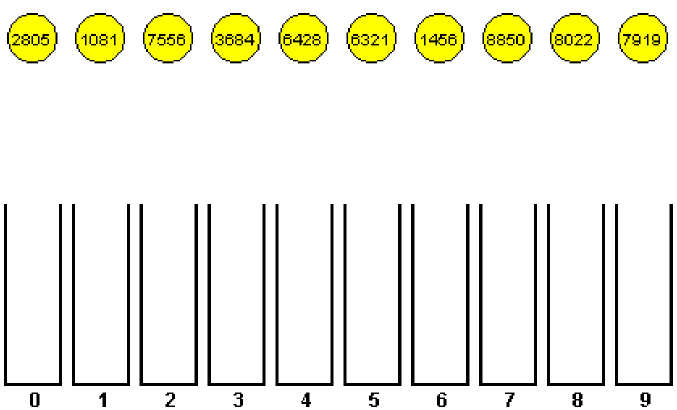


What is the min height of a tree with X external nodes?

CSSE 230 Day 24

Sorting Lower Bound
Radix Sort

Radix sort to the rescue … sort of…



http://www.cs.auckland.ac.nz/software/AlgAnim/radixsort.html

Announcements

- EditorTree evals due last night – late is better than never on these, though!
- Questions on WA8?
- Demo of Doublets
  - Ask questions

A Lower-Bound on Sorting Time

We can't do much better than what we already know how to do.

What's the best best case?

- Lower bound for best case?
- A particular algorithm that achieves this?

What's the best worst case?

- Want a function f(N)
  such that the **worst case running time**
  for all sorting algorithms is $\Omega(f(N))$

- How do we get a handle on
  "all sorting algorithms"?

Tricky!

What are "all sorting algorithms"?

- We can't list all sorting algorithms and analyze all of them
  - Why not?

- But we can find a uniform representation of any sorting algorithm that is based on **comparing** elements of the array to each other

First of all...

- The problem of sorting N elements is at least as hard as determining their ordering
  - e.g., determining that $a_3 < a_4 < a_1 < a_5 < a_2$
  - sorting = determining order, then movement
- So any lower bound on all "order-determination" algorithms is also a lower bound on "all sorting algorithms"

Q1

Sort Decision Trees

- Let A be any **comparison-based algorithm** for sorting an array of distinct elements
- Note: sorting is asymptotically equivalent to determining the correct order of the originals
- We can draw an EBT that corresponds to the comparisons that will be used by A to sort an array of N elements
  - This is called a **sort decision tree**
  - Just a pen-and-paper concept, not actually a data structure
  - Different algorithms will have different trees

Q2-4

So what?

- Minimum number of external nodes in a sort decision tree? (As a function of N)
- Is this number dependent on the algorithm?
- What's the height of the shortest EBT with that many external nodes?

$$\lceil \log N! \rceil \approx N \log N - 1.44N = \Omega(N \log N)$$

No comparison-based sorting algorithm, known or not yet discovered, can **ever** do better than this!

Can we do better than *N* log *N*?

- $\Omega(N \log N)$ is the best we can do if we compare items

- Can we sort without comparing items?

Yes, we can! We can avoid comparing items and still sort. This is fast if the range of data is small.

Q5

- O(N) sort: Bucket sort
  - Works if possible values come from limited range
  - Example: Exam grades histogram
- A variation: Radix sort

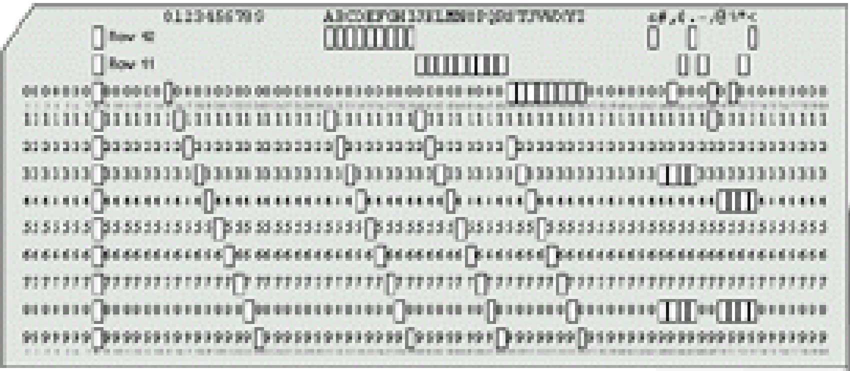
Radix sort

- A picture is worth 10^3 words, but an animation is worth 2^{10} pictures, so we will look at one.
- http://www.cs.auckland.ac.nz/software/AlgAnim/radixsort.html

RadixSort is almost O(n)

- It is O(kn)
  - Looking back at the radix sort algorithm, what is k?
- Look at some extreme cases:
  - If all integers in range 0-100 (so many duplicates if N is large),m then k = _____
  - If all N integers are distinct, k = ____

Radix sort example: card sorter



Used an appropriate combo of mechanical, digital, and human effort to get the job done.

Type 82 Electric Punched Card Sorting Machine

http://en.wikipedia.org/wiki/IBM_card_sorter