

Elementary Sorting Algorithms

n We are given n records to sort.
n There are a number of simple sorting algorithms whose worst and average case performance is quadratic O(n²):
- Insertion sort
- Selection sort
- Bubble sort

The Insertion Sort Algorithm

n Given an array of integers
n The Insertion Sort algorithm views the array as having a sorted side and an unsorted side.
n The sorted side starts with just the first element, which is not necessarily the smallest element.
n The sorted side grows by taking the front element from the unsorted side.

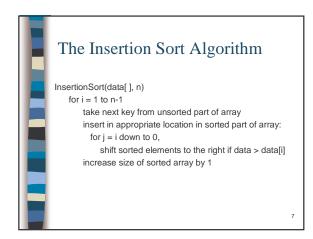
The Insertion Sort Algorithm
(cont.)

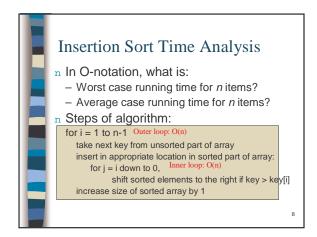
n The sorted side grows by taking the front element from the unsorted side.
n Inserting it in the place that keeps the sorted side arranged from small to large.
n In some cases there is no need to move the new inserted item.

How to Insert an Element

n Copy the new element to a separate location.
n Shift elements in the sorted side, creating an open space for the new element.
n Continue shifting elements until you reach the location for the new element.
n Copy the new element back into the array, at the correct location.

1





The Selection Sort Algorithm

n Start by finding the smallest element.
n Swap the smallest entry with the first element.
n Part of the array is sorted.
n Find the smallest element in the unsorted side.
n Swap with the front of the unsorted side.
n The size of the sorted side is increased by one element.
n Continue until the unsorted side has just one number. Why?

The Selection Sort Algorithm
(cont.)

Basic Idea: Repeatedly select the smallest element, and move this element to the front of the unsorted side.

Selectsort(data[], n) for i = 1 to n-1 find smallest key in unsorted part of array; swap smallest item to front of unsorted array; decrease size of unsorted array by 1;

Selection Time Sort Analysis

n In O-notation, what is:

- Worst case running time for n items?

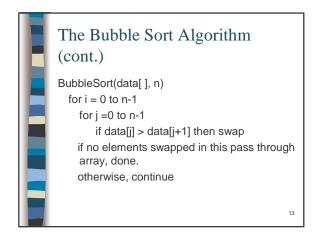
- Average case running time for n items?

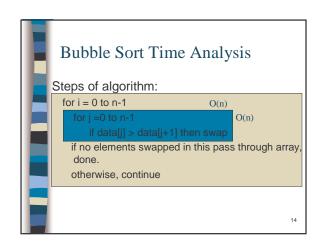
n Steps of algorithm:
for i = 1 to n-1 O(n)
find smallest key in unsorted part of array O(n)
swap smallest item to front of unsorted array
decrease size of unsorted array by 1

n Selection sort analysis: O(n²)

The Bubble Sort Algorithm

n Scan the array from right to left.
n Look at pairs of elements (adjacent elements) in the array, and swaps their order if needed.
n Repeatedly scan the array from right to left elements until you reach the location for the new element.
n Continue scanning until done





Conclusions n Selection Sort, Insertion Sort, and Bubble Sort all have a worst-case time of O(n²), making them impractical for large arrays. n But they are easy to program, easy to debug. n Insertion Sort also has good performance when the array is nearly sorted to begin with. n But more sophisticated sorting algorithms are needed when good performance is needed in all cases for large arrays. Next time: Quick Sort, Merge Sort, and Radix Sort.