Concepts of Algorithms
CS-211
Chapter 3
Sorting (Selection Sort)

Dr. Fethi Fkih
College of Computer
Qassim University

Selection Sort

Why do we care so much about sorting?

- sorting is used by many applications
- Sorting is initial step of many algorithms
- many techniques can be illustrated by studying sorting

Selection Sort **(Continued)**

Suppose an array **A** with ***N*** elements A[0], A[1], . . . A[N-1] is in memory. The Selection sort algorithm for sorting ***A*** works as follows**.** First find the smallest element in the list and swap/exchange it with the first position. Then find the second smallest element in the array and swap/exchange it with the second position. Continue in this way until the entire array is sorted.

Selection Sort (Continued)

Pass 1: Find the location LOC of the smallest in the list of N elements A[0], A[1], ... A[N-1], and then interchange A[LOC] and A[0]. Then: A[0] is sorted.

Pass 2: Find the location LOC of the smallest in the sublist of N-1 elements A[1], A[2], . . . A[N-1], and interchange A[LOC] and A[1]. Then: A[0], A[1] are sorted. Since A[0] $\leq$ A[1].

. . . . . .

. . . . . .

Pass N-1: Find the location LOC of the smallest A[N-2] and A[N-1], and then interchanged A[LOC] and A[N-1].

Then: A[0], A[1], A[2], ... A[N-1] are sorted.

Selection Sort (Continued)

Selection sort is:

- The simplest sorting techniques.
- a good algorithm to sort a small number of elements
- an incremental algorithm – induction method

Selection Sort Algorithm

Algorithm: SELECTION SORT (A, K, N, LOC)

[An Array **A** is in memory. This procedure finds the location Loc of the smallest element among A[K], A[K+1], ...A[N-1].]

1. Repeat steps 2 to 5 for K=0,1,2, ... N-2:

2. Set MIN = A[K] and LOC = K. [Initializes pointers.]

3. Repeat for J=K+1, K+2, ... N - 1

4. If MIN > A[J], then: MIN = A[J] and LOC = J. [End of loop.]

5. Set TEMP = A[K], A[K] = A[LOC] and A[LOC] = TEMP
[Interchange A[K] and A[LOC].]
[End of step 1 loop.]

6. End

Selection Sort Example

Pass	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]
K=0, LOC=3	**77**	33	44	**11**	88	22	66	55
K=1, LOC=5	11	**33**	44	77	88	**22**	66	55
K=2, LOC=5	11	22	**44**	77	88	**33**	66	55
K=3, LOC=5	11	22	33	**77**	88	**44**	66	55
K=4, LOC=7	11	22	33	44	**88**	77	66	**55**
K=5, LOC=6	11	22	33	44	55	**77**	**66**	88
K=6, LOC=6	11	22	33	44	55	66	**77**	88
Sorted	11	22	33	44	55	66	77	88

Analysis of Algorithms

- It easy to see that the number of element comparisons performed by algorithm is

$$\sum_{s=1}^{n-1}(n-s) = (n-1)+(n-2)+.....1 = \sum_{s=1}^{n-1} S = n(n-1)/2$$

- Observation : the number of element comparison by algorithm selection sort is **n(n-1)/2 = O(n²)**

Analysis of Algorithms

- Best Case : Array is sorted

➔Complexity = $O(n^2)$

- Worst Case : Array in inverse order

➔ Complexity = $O(n^2)$

END