Selection Sort

Introduction

Selection sort is an in-place sorting algorithm with complexity O(n^2). Selection sort generally performs worse than insertion sort which is also a O(n^2) in-place algorithm.

Selection sort outperforms divide-and-conquer algorithms on small arrays and is easy to implement. This makes it a good choice alongside insertion sort in hybrid approaches to sorting.

Correctness Proof

I will prove the correctness of my implementation by a loop invariant proof using the following invariant:

Loop Invariant: At the start of the i^{th} iteration of the loop, the array A[0:i-1], which does not include A[i], contains the smallest i-1 numbers in final sorted order.

Initialization: At the start of the first loop the loop invariant states: "At the start of the first iteration of the loop, the array A[0:0], contains the smallest 0 numbers in final sorted order." Since this array is empty and an empty array is sorted by definition this step is proven.

Maintenance: Assume that the loop invariant holds at the start of the i^{th} iteration. Then the array A[0:i-1] contains the smallest i-1 numbers in sorted order. In the body of the loop we add the next lowest number to the end of the sorted array resulting in the array A[0:i] containing the lowest i numbers in sorted order which is what we needed to show.

Termination: When the for loop terminates i = (n-1)+1 this means that A[0:n]=A is sorted. Since A is now sorted the algorithm is correct.

Run-time Analysis

CODE	COST	TIME
`for(int i=0; i<n-1; i++)`	c_1	$\sum_{i=0}^{n-1} 1 = n$
`    min = i`	c_2	$\sum_{i=0}^{n-2} 1 = n-1$
`    for(int j=i+1; j<n; j++)`	c_3	$\sum_{i=0}^{n-2} \sum_{j=i+1}^{n} 1 = \frac{n^2+n-2}{2}$
`        if(a[j]<a[min])`	c_4	$\sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} 1 = \frac{n^2-n}{2}$
`            min = j`	c_5	$\sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} 1 = \frac{n^2-n}{2}$
`    if(min != i)`	c_6	$\sum_{i=0}^{n-2} 1 = n-1$
`        temp = a[i]`	c_7	$\sum_{i=0}^{n-2} 1 = n-1$
`        a[i] = a[temp]`	c_8	$\sum_{i=0}^{n-2} 1 = n-1$
`        a[temp] = a[i]`	c_9	$\sum_{i=0}^{n-2} 1 = n-1$

Multiplying COST x TIME and adding everything up we obtain the following:

$c_1(n) + c_2(n-1) + c_3(\frac{n^2+n-2}{2}) + c_4(\frac{n^2-n}{2}) + c_5(\frac{n^2-n}{2}) + c_6(n-1) + c_7(n-1) + c_8(n-1) + c_9(n-1)$

$=n^2(c_3 + c_4 + c_5) + n(c_1 + c_2 + \frac{c_3}{2} + \frac{c_4}{2} + \frac{c_5}{2} + c_6 + c_7 + c_8 + c_9) - (c_2 + c_3 + c_6 + c_7 + c_8 + c_9)$

This polynomial can be represented by $an^2 + bn - c$ where a, b and c are defined in terms of the cost c_i. This means that our implementation of Selection Sort runs in O(n^2).