Aditya Shirode, Rhythm Shah, Shrenuj Gandhi

Definition. An *inversion* in a permutation π over $\{1, \ldots, n\}$ is a pair of integers i and j such that $i < j$, but $\pi(i) > \pi(j)$. More concretely: in an array of n elements, it's a pair i, j with $i < j$ and $A[i] > A[j]$.

6. [4 points] *Purpose: understanding the concept of an inversion and applying it to detailed analysis of a sorting algorithm*

   (a) Prove that the number of key comparisons during insertion sort is $\geq$ the number of inversions in the original array. Under what condition is the number of key comparisons equal to the number of inversions?

   (b) What is the worst-case number of key comparisons as a function of the number of inversions. Your answer should be in the form $I(A) + f(n)$, where $I(A)$ is the number of inversions in A and $f(n)$ is a function of n. Prove your answer.

   3 points for (a), 1 point for (b)

Answer:

(a) To prove that number of key comparisons during insertion sort is $\geq$ the number of inversions in the original array, we consider the edge cases.

Case 1 : When array is sorted (best case for insertion sort)
In this case, the key value (from j=2 to n) is compared to previous element during the sort. Hence there are (n-1) comparisons. However, since the array is sorted, the number of inversions is 0. This satisfies the condition that number of comparisons during sort $\geq$ number of inversions in original array.

Case 2 : When array is in decreasing order (worst case for insertion sort)
In this case, every new key value (from j=2 to n) is compared with every element in the sub-sorted array i.e on sorted array on the left. Hence there are $\frac{n(n-1)}{2}$ comparisons. In the original array, every element is greater than the element on its right. Hence the number of inversions are $\frac{n(n-1)}{2}$. This also satisfies the condition that number of comparisons during sort $\geq$ number of inversions in original array.

Since the condition is satisfied by the corner cases, we can say that that number of comparisons during sort $\geq$ number of inversions in original array.

As we have seen in *Case* 2, number of comparisons during sort = number of inversions in original array when the array is in decreasing order (worst case scenario for insertion sort).

(b) Let C(A) be the worst-case number of key comparisons would be defined in the form

$$C(A) = I(A) + f(n)$$

Our task is to determine f(n). We have seen in 6a (case 1 and case 2) that $C(A) \geq I(A)$. More precisely, when $I(A) = 0$, $C(A) = n - 1$. And when $I(A) = \frac{n(n-1)}{2}$, $C(A) = I(A)$. From this observation we can say that for all the cases in between, $C(A) - I(A)$ lies between 0 and n-1. In other words, this difference grows slowly than n (size of the array). Hence we can say that $f(n) = O(n)$.

Hence the equation becomes

$$C(A) = I(A) + O(n)$$