Worst – case Optimality of InsertionSort and Lower Bounds (worst – case and average) on Sorting that removes at most one inversion after each comparison

There are n ! permutations of n distinct elements.

An inversion in permutation π is a pair (j, k) such that j appears before k in π but k < j.

For example, permutation (3, 4, 1, 5, 2) has 5 inversions :
(3, 1), (3, 2), (4, 1), (4, 2) and (5, 2).

Definition

C - a class of sorting algorithms that sort by comparisons of keys and remove at most one inversion after each comparison.

Insertion sort is in class C.

Exercise : Prove it !

Let

T (n)

be the minimum number of comparisons that any algorithm in class C must perform on any input of size n in the worst case.

Let

T_{avg} (n)

be the minimum number of comparisons that any algorithm in class C must perform on any input of size n in the average case, assuming that all arrangements (permutations) of input elements are equally likely $\left(\text{have the same probability of } \frac{1}{n!}\right)$.

We will establish lower bounds on T (n) and T_{avg} (n).

Let ' s count inversions in permutations.

"Ordered" permutation (1, 2, 3, ..., n) has 0 inversions.

This is the best - case scenario for a sorting program in class C.

All pairs (j, k), where $1 \le k < j \le n$,
are inversions in "anti-ordered" permutation $(n, \ldots, 3, 2, 1)$.

This is the worst - case scenario for a sorting program in class C.

How many pairs of that kind are there?

n^2 pairs (j, k)

n are of the form (j, j)

So, $n^2 - n$ are of the form (j, k) where $j \neq k$.

Half of them are of the form (j, k), where $1 \le k < j \le n$

So, there are $\frac{n^2 - n}{2} = \frac{n(n-1)}{2}$ inversions in "anti-ordered" permutation $(n, \ldots, 3, 2, 1)$.

Therorem 1. Every algorithm in class C
must perform at least $\frac{n(n-1)}{2}$ comparisons in the worst case.

Proof. There are $\frac{n(n-1)}{2}$ inversions in a decreasingly ordered input array of n elements
for a sorting program P in class C. So, P must perform at least that many comparisons.

Corollary. Insertion Sort is worst - case optimal in class C.
(Make sure you know why.)

Theorem 2. Every algorithm in class C must
perform at least $\frac{n(n-1)}{4}$ comparisons in the average case.

Proof. Let $n \ge 2$
(otherwise, no comparisons are made).

Given permutation π, let reverse (π) be the result of reversing the order of π.

For example, reverse $(4, 2, 1, 3) = (3, 1, 2, 4)$.

Of course, reverse (π) is a permutation and is unique for each π,

reverse $(\pi) \neq \pi$
(since the only palindrome permutation is for n = 1)

and reverse (reverse (π)) = π.

Thus we can partition the set Perm_n of all n! permutations of (1, 2, 3, ..., n) onto

$\frac{n!}{2}$ sets $\{\pi, \text{reverse}(\pi)\}$,

where π is any permutations of (1, 2, 3, ..., n). (Make sure you know why.)

The set

$P = \{\{\pi, \text{reverse}(\pi)\} \mid \pi$ is a permutation of (1, 2, 3, ..., n)$\}$

is a partition because :

each element of P is non - empty,

$\bigcup P = \text{Perm}_n$, and

for any R, S $\in$ P, either R = S or $R \cap S = 0$.
(Make sure you know why.)

Since a pair (i, j), where $1 \leq j < i \leq n$,
is an inversion either in a permutation π or in reverse (π),
there is total $\frac{n(n-1)}{2}$ of inversions in each element $\{\pi, \text{reverse}(\pi)\}$ of P.
(Make sure you know why.)

So, the total number of inversions in Perm_n is egual to the total number of permutations in P,
which is equal to $\frac{n(n-1)}{2}$ times the number of elements of P, or

$$\frac{n(n-1)}{2} \times \frac{n!}{2}$$

Hence, since each permutation have the same probability $\frac{1}{n!}$,
the expected number of inversions per permutation is

$$\frac{1}{n!} \times \frac{n(n-1)}{2} \times \frac{n!}{2} =$$

$$= \frac{n(n-1)}{4}$$

which is the same as the number of inversions per permutation on average.

So, each algorithm in C must perform at least $\frac{n(n-1)}{4}$ comparisons on average

(Make sure you know why.) while sorting a permutation of n distinct elements.