# COMP 3711

## Tutorial 3b

## Question 1

Consider the HIRE-ASSISTANT algorithm described in the lecture notes.

Assume that the candidates are presented in a random order.

The analysis in the lecture notes calculated the ***Expected*** number of hires.

For this problem calculate:

(a) **the probability** that you hire exactly one person.

(b) **the probability** that you hire exactly $n$ people.

There are $n!$ different interview orderings that can be chosen and each ordering is equally likely to be chosen with probability $1/n!$.

(a) Hiring exactly one time means
that you hired the first person and that
person was the best person.

The probability of the first person being the best person is exactly

$$\frac{1}{n}.$$

(b) Hiring exactly $n$ times
that the candidates are interviewed
in increasing order of their quality.

The probability of this unique random ordering being chosen is exactly

$$\frac{1}{n!}.$$

## Question 2

Use indicator random variables to solve the following, **hat-check problem**.

Each of $n$ customers gives a hat to a hat-check person at a restaurant.

The hat-check person gives the hats back to the customers in a random order.

**What is the expected number of customers who get back their own hat?**

An equivalent question is to suppose that there are $n$ students in a class who have submitted homeworks.

The teacher returns the homeworks to the students in a random order and asks them to mark the homework they have been handed.

What is the expected number of students are asked to mark their own homework?

Set

$$X_i = \begin{cases} 1 & \text{if the i-th customer gets back his own hat} \\ 0 & \text{otherwise} \end{cases}$$

$X$ = # customers who gets back their own hat $= \sum_{i=1}^{n} X_i$

Observe that $E(X_i) = \Pr(X_i = 1) = \frac{1}{n}$.

$$E(X) \underset{\text{Defn of } X}{=} E\left(\sum_{i=1}^{n} X_i\right) \underset{\text{Linearity of Expectation}}{=} \sum_{i=1}^{n} E(X_i) = \sum_{i=1}^{n} \frac{1}{n} = 1$$

*Note: $X_i$'s are not **independent**.*
*E.g., when $X_1 = X_2 = \cdots = X_{n-1} = 1$, then $X_n$ must be 1 as well.*
*But Independence is not needed for linearity of expectation.*

## Question 3

Let $A[1..n]$ be an array of n distinct numbers.
In class we said that if $i < j$ and $A[i] > A[j]$, then the pair $(i, j)$ is called an **inversion** of $A$.

Suppose that the elements of $A$ form a uniform random permutation of $\langle 1,2, \dots, n \rangle$.

Use indicator random variables to compute the expected number of inversions.

For $i < j$ set

$$X_{ij} = \begin{cases} 1 & \text{if } A[i] > A[j] \\ 0 & \text{otherwise} \end{cases}$$

Then $E(X_{ij}) = \Pr(X_{ij} = 1) = \frac{1}{2}$. **WHY?**

$$E(X) = E\left(\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right) \longleftarrow \boxed{\text{Sum over all pairs}}$$

$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E(X_{ij}) \longleftarrow \boxed{\text{Linearity of Expectation}}$$

$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{2} = \frac{1}{2} \cdot \frac{n(n-1)}{2} = \frac{n(n-1)}{4}$$

## Implications of the solution

Recall that we showed that Bubblesort and Insertion Sort both performed exactly the *inversion number* of swaps.

The result of this problem actually showed that

**The expected number of swaps performed by Bubblesort and Insertion Sort on a random input of size $n$ is**

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

## Question 4

1. Illustrate how Mergesort would work on input [1,2,3,4,5,6,7,8,9].

2. Illustrate how Mergesort would work on input [9,8,7,6,5,4,3,2,1].

3. Illustrate how Quicksort would work on input [1,2,3,4,5,6,7,8]. Assume that the last item in the subarray is always chosen as the pivot.

4. Illustrate how Quicksort would work on input [8,7,6,5,4,3,2,1]. Assume that the last item in the subarray is always chosen as the pivot.

## Solution 4

1. Illustrate how Mergesort would work on input $[1,2,3,4,5,6,7,8,9]$.

2. Illustrate how Mergesort would work on input $[9,8,7,6,5,4,3,2,1]$.

3. Illustrate how Quicksort would work on input $[1,2,3,4,5,6,7,8]$.
Assume that the last item in the subarray is always chosen as the pivot.

4. Illustrate how Quicksort would work on input $[8,7,6,5,4,3,2,1]$.
Assume that the last item in the subarray is always chosen as the pivot.

See external slides for worked solutions.
Note how the recursive structure for Mergesort does not change.
It remains the same (for fixed $n$) regardless of the actual input.

Note that for both sorted and reverse sorted input, Qsort runs badly.

It is not difficult to extend the analysis of these examples to show that Quicksort always runs in $\Theta(n^2)$ time, on both sorted and reverse-sorted input of size $n$.