CPS130 : Homework 2 Solutions

1. Suppose you repeatedly toss a fair coin until you get two heads (not necessarily in a row). What is the probability that you stop on the 10th toss?

For the trials to stop at the 10th trial, it is necessary to have the second head at the 10th toss. That means, there should be exactly 1 head and 8 tails in the first 9 tosses. The probability of that happening is \( \binom{9}{1} \left( \frac{1}{2} \right)^8 \left( \frac{1}{2} \right) \). Also, the 10th toss should be a head so the total probability becomes:

\[
P = \binom{9}{1} \left( \frac{1}{2} \right)^8 \left( \frac{1}{2} \right) \times \left( \frac{1}{2} \right) = \frac{9}{1024}
\]

2. * Suppose that \( n \) balls are tossed into \( n \) bins, where each toss is independent and the ball is equally likely to end up in any bin. What’s the expected number of empty bins?

Let us define \( n \) random variables \( X(1 \) to \( n) \) for each bin where

\[
X_i = \begin{cases} 
0, & \text{if it has atleast 1 ball} \\
1, & \text{if it has none}
\end{cases}
\]

The probability of the \( i \)th bin not having a single ball is the same as finding the probability that each of the \( n \) balls miss the \( i \)th bin. The probability of any ball not reaching the \( i \)th bin is \( (1 - \frac{1}{n}) \). Thus the probability of none of the \( n \) balls reaching the \( i \)th bin will be \( (1 - \frac{1}{n})^n \).

Hence expected number of empty bins will be:

\[
E[\text{no. of empty bins}] = \sum_{i=1}^{n} 1 \times p_{X_i}(1) \\
= \sum_{i=1}^{n} [1 - \frac{1}{n}]^n \\
= n \times (1 - \frac{1}{n})^n \\
\approx n/e
\]

3. Show how quicksort would operate on the following numbers:
   \(< 23, 17, 14, 6, 13, 10, 1, 5, 7, 12 >\)

Make sure to use the first element as pivot each time and please show the status of the array after every call to PARTITION.

The following shows the array after every call to PARTITION. The
underlined element is the new pivot with respect to which the partition will be carried out while the parenthesised element is the element in its correct position after the being the pivot in one of the earlier steps. I am using the last element as the pivot.

\(< 23 \ 17 \ 14 \ 6 \ 13 \ 10 \ 1 \ 5 \ 7 \ 12 >\)
\(< 6 \ 10 \ 1 \ 5 \ 7 \ 12 \ 17 \ 14 \ 23 \ 13 >\)
\(< 6 \ 1 \ 5 \ (7) \ 10 \ (12) \ (13) \ 14 \ 23 \ 17 >\)
\(< 1 \ (5) \ 6 \ (7) \ (10) \ (12) \ (13) \ 14 \ (17) \ 23 >\)
\(< (1) \ (5) \ (6) \ (7) \ (10) \ (12) \ (13) \ (14) \ (17) \ (23) >\)

4. A sorting algorithm is described as **stable** if equal elements are in the same relative order in the sorted sequence as in the original sequence. Is quicksort stable? Explain.

The original Quicksort is not stable. The reason is due to the fact that the pivot is swapped with the rest of the keys. Hence it is possible that it is swapped with a key that is equal making the sort unstable. However, the routine PARTITION given in the book is stable since it ensures by making the pivot the last element and using a previous pointer \(i\), that it is never gets swapped with an equal element.

5. * 7.2-4 [CLRS pg 153]

The problem states that the array is almost sorted. We can interpret it as there is no element further away than \(c\) positions away from its sorted position (where \(c\) is a constant). In INSERTION-SORT, once an element is placed into the sorted position in a sorted subarray, the algorithm moves onto the next element. Hence it will take at most \(cn\) comparisons to sort all the numbers making the time complexity \(O(n)\).

In QUICKSORT, however, PARTITION will produce one subproblem of size \(n - c\) and the other of \(c - 1\), thus the recurrence equation will be:

\[ T(n) = T(n - c) + T(c - 1) + n - 1 \]

Solving it you get \(T(n) = O(n^2)\) which is the worst case time for QUICKSORT. Thus INSERTION-SORT will be more effective in this case.