1 Probabilistic analysis

1. You toss a coin, which gives head with probability \( p \) and tail with probability \( 1 - p \). How many times do you expect to toss until you get the first head? Note: use the formula of expected value.

2. DNA sequences consist four nucleotides: A, T, C and G. We generate two random DNA sequences \( S_1 \) and \( S_2 \), by picking A, T, C, G with the same chance. The length of the two DNA sequences are both \( n \). What is the expected number of positions where \( S_1 \) and \( S_2 \) match (i.e. \( S_1[i] = S_2[i] \))?

2 Balls and Bins, again

Suppose we throw \( n \) balls (note we will throw exactly \( n \) balls) into \( n \) bins with the probability of a ball landing in each of the \( n \) bins being equal. We assume each throwing is independent of other throwing. You can assume \( n \) is large. You may need the following mathematical fact: when \( n \to \infty \), \((1 - \frac{1}{n})^n \to e^{-1}\), where \( e \) is the well-known mathematical constant.

1. What is the probability of a particular box (say the first box) end up being empty after the \( n \) throwing?

2. What is the expected number of empty bins?

3 QuickSort

We consider a variation of the QuickSort, which uses the following partition algorithm. Note that the rest of the QuickSort algorithm remains intact: will recursively sort the \( A^- \) and \( A^+ \) partitions. For analysis, we assume the elements in \( A \) are all distinct.

\[ \text{Partition}(A, l, r) \]

1. \textbf{while} true \textbf{do}
2. \hspace{1em} Choose an element \( A[i] \) from \( A[l..r] \) uniformly at random
3. \hspace{1em} \( A^- \leftarrow \) elements in \( A[l..r] \) that are smaller than \( A[i] \).
4. \hspace{1em} \( A^+ \leftarrow \) elements in \( A[l..r] \) that are larger than \( A[i] \).
5. \hspace{1em} if Size of \( A^- \) and size of \( A^+ \) are both no smaller than a quarter of size of \( A[l..r] \) (i.e. \((r - l + 1)/4\)) \textbf{then}
6. \hspace{2em} Return \( A[i] \) (and \( A^- \) and \( A^+ \)) as the result of the partition.
7. \hspace{1em} \textbf{end if}
8. \textbf{end while}

Now answer the following questions.

1. What is the worst-case running time of Partition of list of \( n \) elements? For a single iteration, what is the probability of statement 6 (the Return) will be called? What is the expected running time for a list with \( n \) elements?

The QuickSort algorithm spends most of the time to run Partition subroutine for \( A[l, r] \) of different sizes. We now group the array portions as follows: we say a subproblem is of type \( i \) if its size is between \( n(3/4)^{i+1} \) and \( n(3/4)^i \).

2. How many type \( i \) problems are there? Why? What is the expected running time for Partition to run on a type \( i \) subproblem?

3. What is the total expected running time of QuickSort?