ICS 311: Algorithms Spring 2025

Problem Set 5

Kyle Berney *Due: Friday, Feb 14, 2025 at 4pm*

You may discuss the problems with your classmates, however **you must write up the solutions on your own** and **list the names** of every person with whom you discussed each problem.

Start **every** problem on a separate sheet of paper, with the exception of Problems 1 (Peer credit assignment) – Problem 1 can be on the same page as any other problem. Any problem that starts on the same sheet of paper as some other problem will receive 0 points!

1 Peer Credit Assignment (1 point extra credit for replying)

Please list the names of the other members of your peer group for this week and the number of extra credit points you think they deserve for their participation in group work.

- You have a total of 60 points to allocate across all of your peers.
- You can distribute the points equally, give them all to one person, or do something in between.
- You need not allocate all the points available to you.
- ***You cannot allocate any points to yourself!*** Points allocated to yourself will not be recorded.

2 Coin tossing (20 pts)

Consider a random experiment of tossing a biased coin three times. We assume that the tosses are independent of each other and the probability of a head is 0.2. Let X be the random variable whose value represents the number of heads obtained.

(a) (4 pts) List the possible outcomes of three coin tosses. What are the probabilities $Pr[X = 0]$, $Pr[X = 1]$, $Pr[X = 2]$, $Pr[X = 3]$?

(b) (8 pts) Compute $E[X]$ using the definition of expected value (i.e., without the use of indicator random variables). *Show your work.*

(c) (8 pts) Let X_i be an indicator random variable, whose value is 1 if the i-th toss resulted in heads. Define X in terms of X_i's and compute $E[X]$ using this expression. *Show your work.*

3 Efficient Encoding (15 pts)

An information source generates symbols at random from a four-letter alphabet $\{a, b, c, d\}$ with probabilities $Pr[a] = 5/9$, $Pr[b] = 2/9$, $Pr[c] = 1/9$, $Pr[d] = 1/9$. A coding scheme encodes these symbols into binary codes as follows:

a : 0
b : 10
c : 110
d : 111

Compute the expected length of the encoding of n letters generated by the information source. You may use any method you like, but must show your work/justify your answer.

4 3-way Quicksort (45 pts)

The analysis of QUICKSORT assumed that there are no duplicate values. In this problem we will study the case when the sequence of items we want to sort contains duplicate values.

(a) **(5 pts)** What is the expected runtime of RANDOMIZED-QUICKSORT on a sequence of n identical items (i.e. all entries of the input array are the same)? How does it compare to the expected runtime of RANDOMIZED-QUICKSORT on an array of values picked uniformly at random which contains no duplicates. Justify your answer.

To avoid the case above, we are going to design a new algorithm, 3WAYPARTITION, which partitions array into three parts, those that are strictly less than the pivot, equal to the pivot, and strictly greater than the pivot. Then we will use 3WAYPARTITION to sort arrays which contain duplicate values efficiently.

(b) **(10 pts)** Design a new algorithm 3WAYPARTITION(A, p, r), which takes as input an array A and two indices p and r and returns a pair of indices (e, g). 3WAYPARTITION(A, p, r) should partition the array A around the pivot $q = A[r]$ such that every element of $A[p..(e-1)]$ is strictly smaller than q, every element of $A[e..(g-1)]$ is equal to q and every element of $A[g..r]$ is strictly greater than q. Write down the **pseudocode** of your algorithm, and **analyze its runtime**. Show your work. Your algorithm should have the same runtime as the PARTITION(A, p, r) algorithm presented in the lecture notes/book.

Hint: modify PARTITION(A, p, r)*, so that it adds the items that are greater than q from the right end of the array and all items that are equal to q to the right of all items that are smaller than q. You will need to keep additional pointers which will point to the locations in A where the next item should be written.*

(c) **(15 pts)** Using either loop invariants or induction prove that your 3WAYPARTITION(A, p, r) correctly partitions the array A around the pivot $q = A[r]$.

(d) **(10 pts)** Design a new algorithm 3WAYQUICKSORT which uses 3WAYPARTITION to sort an array of n items and avoids the problems of part (a). Write down the pseudocode.

(e) **(5 pts)** What is the runtime of 3WAYQUICKSORT on an array of n items whose values are picked uniformly at random? What is the runtime of 3WAYQUICKSORT on a sequence of n **identical** items? Justify your answers.

5 Streaming Uniform Random Element (20 pts)

A *streaming algorithm* processes data as a stream, i.e., input is presented as a sequence of items that can be examined only once. Hence, a streaming algorithm must make a decision about each input element at the time when it sees that element for the first time and based only on information that it has seen so far, without any knowledge of future inputs. Streaming algorithms are useful when a lot of data is being produced very quickly and the system is unable to store and examine the data later. For example, network routers that implement network security must process lots of internet packets and make a decision on whether to let each packet through or not (e.g., dropping the packets that happen to be malicious) without the luxury of waiting for all the data to arrive first or of storing the data for further inspection later (a malicious attacker might do the damage by then).

In this problem, you will analyze a streaming algorithm for selecting an item uniformly at random from a given stream.

You are given a stream of $n > 1$ elements, where n is unknown in advance (i.e., the value of n is only known after all n elements in the stream have been revealed to the algorithm). We want to select one element from this stream uniformly at random. To accomplish this, our algorithm does the following. When it sees the first element in the stream, it stores it. Next, whenever it sees the i-th element in the stream ($i = 2, \ldots n$), with probability $\frac{1}{i}$ it decides to replace the previously stored element with the current one. After the stream ends, i.e., after all n elements have been examined, the algorithm returns/outputs the element that it currently stores.

Prove that when this algorithm terminates, it is equally likely to output any one of the n elements. That is, prove that the probability that this algorithm outputs the i-th element for $i = 1, \ldots, n$ is equal $\frac{1}{n}$. *Hint: make sure your analysis accounts for the fact that an item seen earlier could be replaced by an item seen later.*

6 Linearity of expectations (OPTIONAL - 0 pts)

Let X and Y be arbitrary non-negative random variables, and let Z be another random variable, such that $Z = \max(X, Y)$. Prove that $E[Z] \leq E[X] + E[Y]$. *Hint: Try to define Z in terms of X and Y.*

7 Curious Permutation (OPTIONAL - 0 pts)

You are tasked with designing an algorithm that takes an array A and randomly permutes its elements in such a way that the resulting array can be any permutation, except the original input. You developed the following algorithm:

RANDOM-PERMUTE(A)

1: $n = A.length$
2: **for** $i = 1$ **to** $n - 1$
3: swap $A[i]$ with $A[\text{RANDOM}(i + 1, n)]$

Does this algorithm work correctly? Justify your answer.