Assignment 1

COMP 4804 - Winter 2015

March 26, 2015

1 Instructions

This is due at the start of the class on Thursday, January 29, 2015. Please write clearly and answer questions precisely. As a thumb rule, the answer should be limited to 1 written page/per Question, with ample spacing between lines and in margins. Always start a new question on a new page, starting with Question 1, followed by Question 2, ..., Question n. Please cite all the references (including web-sites, names of friends, etc.) which you have used/consulted as the source of information for each of the questions. BTW, when a question asks you to design an algorithm - it requires you to (1) Clearly spell out the steps of your algorithm in pseudocode (2) Prove that your algorithm is correct and (3) Analyze the running time.

2 Problems

1. (a) We toss a fair coin 8 times. Compute the probability of the following events and give some reasoning for your answers.
   i. Obtaining at least four heads.
   ii. Obtaining same number of heads and tails.
   iii. Obtaining at least four consecutive tails.

   (b) Suppose we shuffle a standard deck of 52 cards. Let us assume that the shuffled deck is a random permutation, among all permutations, of these cards. What is the probability of the following events and give some reasoning for your answers.
   i. First two cards include at least one of the Queens.
   ii. First 4 cards are from Spades.
   iii. First four cards do not include an Ace.

2. (a) Construct a sample space and give examples of events where \( \Pr(A|B) = \Pr(A) \), \( \Pr(A|B) < \Pr(A) \) and \( \Pr(A|B) > \Pr(A) \).

   (b) Construct a sample space and three events such that any pair of them are independent, but all three of them are not mutually independent.

3. Each of the two persons toss three fair coins. What is the probability that they obtain the same number of Heads?
4. Suppose two players play a game till one player wins a total of \( n \) rounds. In each round, the outcome is either a win or a loss, and each player is equally capable of winning a round. What is the probability that at the conclusion of the game, when a player has won \( n \) rounds, the loser has won \( k \) (< \( n \)) rounds? Give some reasoning for your answer.

5. Suppose I have two coins in my pocket - a fair coin and a two headed coin (i.e. both sides are Heads). I pull one of the coins (randomly) from my pocket and toss it and obtain a Head. What is the probability that the coin which I tossed is the fair coin? What is the probability that it is the 2-headed coin? Provide some reasoning for your answer.

6. A beer distillery has a tester (hopefully not a person) which can very quickly test the quality of each bottled beer on its assembly line and accept or reject them based on whether they pass or fail its test. If a bottle of the beer doesn’t meet the standards then with 95% certainty the tester will report that the bottled beer is unacceptable. If a bottled beer is perfect, than there is still a 5% chance that the tester may say that the beer doesn’t meet the standards. Suppose on the average this distillery produces 10% of the bottled beers which do not meet the standards on any given day. Let us do the following experiment. Choose a bottle of beer uniformly at random before it reaches the tester, and let us assume that the tester reports that this particular bottled beer doesn’t meet the standards. What is the probability that this bottle actually doesn’t meet the standards? Provide some reasoning for your answer.

7. Suppose you have a biased coin \( C \) (i.e. \( \text{Pr}(\text{H}) \neq \text{Pr}(\text{T}) \neq \frac{1}{2} \)). Von Neumann suggested the following method to use the biased coin \( C \) to simulate an unbiased coin. Strategy is to toss \( C \) twice and note down the outcomes. If the first toss of \( C \) results in Head and the second one to Tail, then we say that the outcome is \( \text{Head} \). If the first toss of \( C \) results in Tail and second one to Head then the outcome is \( \text{Tail} \). Otherwise (that is both the tosses of \( C \) are either Heads or Tails) we repeat the above process (i.e., we will again toss twice ... ). Show that the above method simulates an unbiased coin, i.e. probability that we output \( \text{Head} \) is same as the probability that we output \( \text{Tail} \).

8. There is a collection of \( n \)-labelled balls in a non-transparent bag. The label is an integer in the range \( 1 \ldots n \), where no two balls have the same label. Your task is to draw the balls one after another till all balls are drawn. When a ball is drawn, you do one of the following: If the drawn ball has the highest label among the balls which have been drawn so far, then you keep it. Otherwise, you discard that ball. (This is a game where there is no replacement.) Note that you can have anywhere from 1 to \( n \) balls by the end of the game. Estimate, on the average, how many balls you will have at the end of the game?

9. Running time of many divide-and-conquer algorithms are expressed by a recurrence relation. Consider the following recurrence relation

\[
T(n) = T(n/3) + T(2n/3) + n,
\]

where \( n \) is the size of the problem, typically very large. You can assume that \( T(1) = 1 \). (BTW, \( T(n) \) represents the time to solve the problem of size \( n \). It is solved recursively by partitioning it in two subproblems, one of size \( n/3 \) and other of size \( 2n/3 \), and the time required to partition the problem and merging the solution of the subproblems is \( n \).) Show, using the substitution method, that \( T(n) = O(n \log n) \). (Note: You won’t be able to apply
Master’s Theorem directly as the recursion tree is not that ‘nice’. In the substitution method, you guess a solution for the recurrence, and then verify it, typically using induction.)

10. This is one of the ways to analyze quicksort. First recall quicksort. Suppose that we want to sort an array $A$, consisting of $n$ distinct real numbers. Quicksort works as follows:

- If the array has $> 1$ elements,
  - Pick a pivot element $q$, uniformly at random, among the elements of the array.
  - Split the array into three parts:
    (a) $A_1$ consists of all elements of the array that are $< q$;
    (b) $A_2$ consists of all elements of the array that are $> q$;
    (c) $q$.
  - Sort $A_1$ and $A_2$ recursively, and then return the sequence comprising of sorted elements of $A_1$, $q$, and sorted elements of $A_2$.
- Otherwise, return the array.

Note that arrays $A_1$ and $A_2$ are formed by comparing each element of the array with $q$. Let $T(n)$ denote the expected number of comparisons which are made during this algorithm on an array consisting of $n$ elements. Observe that $T(n)$ can be expressed by $T(0) = 0$, $T(1) = 0$, and for $n \geq 2$,

$$T(n) = (n - 1) + \frac{1}{n} \sum_{k=0}^{n-1} (T(k) + T(n - 1 - k)).$$

The reasoning for this is as follows: Note that $(n - 1)$ accounts for number of comparisons made to form $A_1$ and $A_2$. As the pivoting element can partition the array, so that $A_1$ (and $A_2$) can have any of the possible sizes from 0 to $n - 1$, we take the average of the times for the recursive subproblems. Show, using the substitution method, that $T(n) = O(n \log n)$. 

3