Comp 160: Algorithms **Recitation 4**

1. Let's do an exercise to understand the differences between the ways we measure runtimes:

Algorithm 1 StrangeSort (Array A of n numbers)

$i \leftarrow$ RandomIntegerFrom1To(n)
if $i = 1$ **then**
    InsertionSort(A)
if $i = 2$ **then**
    BubbleSort(A)
if $i \geq 3$ **then**
    MergeSort(A)

- First consider an ideal scenario: you can choose both the array A and the value of i. What would you choose so that runtime is minimized? What would be the resulting runtime (as a function of n)?
- What is the worst-case runtime of this algorithm? (make a bad choice for A and value of i)
- As a third (and more reasonable) middle ground we define $S(n)$ as the worst case runtime of StrangeSort. Express that runtime as a combination of IRVs and runtime of other sorting algorithms as necessary. Justify your expression with 1-2 sentences. Give a bound on $E[S(n)]$ that only depends on n.
- What does this imply for worst case? and expected runtime? Do the previous answers match your intuition?

2. Let's give a different argument to show that randomized selection will run in expected linear time. Let's start by recalling what the algorithm does.

   (a) (warm-up) give the pseudocode of randomized selection

   (b) What is the worst-case runtime of one level of RandomizedSelection? (do not count time spent in recursion)

   (c) As you know, RandomizedSelection does one recursive call with a smaller array. Let X be an IRV that is 1 if the array in the recursive call has size $3n/4$ or less. Compute an upper bound $P(X = 1)$ (and thus $E[X]$)

   Hint: how large can the array we generate be? Express it as a function of the rank of $A[i]$ (the randomly selected pivot)

   (d) Let S_i be the size of A at the i-th level of recursion. By definition, $S_0 = n$. Using the previous answer, compute an upper bound for the expectancy of S_1, S_2, and in general S_i.

   Hint: the definition of expectancy of a random variable Y is $E[Y] = \sum_{y \in Values(Y)} y \cdot P(Y = y)$. Upper bound that sum using X

(e) Using the previous answers, give an upper bound for the expectancy of randomized selection.

Hint: Compute the expected runtime for each level of recursion and add all of them up. You should get a nice geometric series

3. You may have heard that if you have a monkey typing keys at random, it will eventually type your favorite book. Give a reasonable estimate of the time it will take until this happens.

   Note: you can use the following assumptions:

   - The monkey types one character per second
   - The keyboard only has the 26 letters of the English alphabet and the space bar
   - Your favorite book has 100000 characters

   Hint: the big difficulty of this exercise is how to formalize this properly. Questions like this may come in the exam. Make sure to clearly state how you formalize the problem!

Additional practice questions:

4. Have you played *Risk*? It is an old boardgame where people try to conquer portions of the board. Let's just look at a small portion of the game: say that player A attacks player B. In an attack both players roll a 6-sided die. The larger number wins (in case of ties, the defender wins). The effect of a loss is that the losing player losses a unit. When a side has no units remaining that player has lost the fight.

   (a) Find a partner and simulate a battle in which the attacking player has 3 units and the defender has 2

   (b) Justify (in 1-2 sentences) why the defender has more chances of winning the battle if both players have the same number of units

   (c) Say that player B has b units defending a terrain. What is the least amount of units that player A should use so that we expect player A to win?

   (d) **Challenge** Look online for the advanced rules where you can attack with up to 3 units and defend with up to 2 units at a time. Answer the previous questions assuming that each player will always commit as many troops as they can.

5. **Challenge**: redo the monkey problem but with general parameters: say that the monkey types s characters per second, that there are k keys in the keyboard and the book has c characters. Can you find the specific formula? Note that we are interested in the assymptotic behavior (so a good answer would be something like $\Theta(\frac{c^4 2^k}{s^7})$).