HW - Week 13

Name: Sahand Sabour **Student ID:** 2020280401

Page 112:

5.2-3: Since there are 6 sides to a dice, the probability of getting each side would be

$$\Pr\{X = k\} = \frac{1}{6}$$

Hence, the expected value for dice i would be

$$E[X_i] = \sum_{k=1}^{\infty} k \Pr\{X_i = k\} = \frac{\sum k}{6} = \frac{21}{6} = 3.5$$

Accordingly, given n dices, since the dices are similar and have the same expected value, we have that

$$E[X] = E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i] = n \times E[X_i] = 3.5n$$

Page 142:

5.4-2: Initially, given that all the bins are empty, we need to toss at least two balls and at most b+1 balls (pigeonhole principal) to have a bin contain two balls. Accordingly, for the k^{th} toss we know that k-1 balls have been tossed to empty bins since the condition hasn't been met yet. The probability of tossing the ball into one of the k-1 non-empty bins at the k^{th} toss would be $\frac{k-1}{b}$ while the probability of the ball being tossed into an empty been would be calculated via the following multiplication, since after a bin is occupied the numerator decreases by 1 until all bins have been filled, which gives the overall probability as

$$\Pr\{X_b = 2\} = \frac{k-1}{b} \times (\frac{b-k+2}{b}) \times \ldots \times (\frac{b-1}{b})$$

$$\Pr\{X_b = 2\} = \frac{b!\,(k-1)}{b^k (b-k+1)!}$$

Accordingly, the expected number of tosses would be

$$E[X_b] = \sum_{k=2}^{b+1} k \; Pr\{X_b = 2\} = \sum_{k=2}^{b+1} \frac{b!\,(k-1)(k)}{b^k(b-k+1)!}$$

As mentioned previously, considering that the probability of having a bin with two balls after b+1 tosses is 1 due to the pigeonhole principal, the above equation can be rewritten as

$$E[X_b] = 1 + \sum_{k=2}^{b} k \; Pr\{X_b = 2\} = 1 + \sum_{k=2}^{b} \frac{b!\,(k-1)(k)}{b^k(b-k+1)!}$$

Page 144:

5-2.h: In the case that k=0, then the condition A[i]=x is never met, which makes the worst running time n. The expected running time would also be n since all of the elements have to be scanned to ensure that the no element satisfies the condition.

If k=1, the worst running time occurs when x is at the last position of the array, which takes n since all the elements have to be scanned. For the expected running time, since x could be in either of the n positions of A, the probability of x being in a position i would be

$$Pr\{X_i\} = \frac{1}{n}$$

Hence, let m denote the position of X in the array; the expected value of scans to find X would be

$$E[X_i] = \sum_{x=1}^{\infty} x \; Pr\{X_i = x\} = \frac{n(n+1)}{2}\left(\frac{1}{n}\right) = \frac{n+1}{2}$$

5-2.i: I would use DETERMINSTIC-SEARCH. The average time for RANDOMIZED_SEARCH would be n(ln(n)+O(1)) whereas the the average expected running time for the DETERMINSTIC_SEARCH and SCRAMBLE_SEARCH is n, as previously shown. Moreover, generating the permutations in SCRAMBLE_SEARCH has a linear time complexity n, which makes DETERMINSTIC_SEARCH more efficient.

Page 204:

8.4-4: The area of the circle is $A = \pi r^2 = \pi(1-0)^2 = \pi$. We can consider the circle as a set of n rings, each having an area of $\frac{\pi}{n}$. Let us assume that each ring starts at distance r_{i-1} of the center and ends at radius r_i. Hence, it can be derived that

$$\pi r_i^2 - \pi r_{i-1}^2 = \frac{\pi}{n}$$

$$r_i^2 - r_{i-1}^2 = \frac{1}{n}$$

$$r_i = \sqrt{\frac{1}{n} + r_{i-1}^2}$$

which can be considered as a recurrence relation that gives

$$r_0 = \sqrt{\frac{1}{n}} \ \ and\ r_1 = \sqrt{\frac{1}{n} + \frac{1}{n}} = \sqrt{\frac{2}{n}}$$

Accordingly, the general case would be

$$r_i = \sqrt{\frac{i-1}{n}} \ \ and\ r_{i+1} = \sqrt{\frac{1}{n} + \frac{i-1}{n}} = \sqrt{\frac{i}{n}} \ (proof\ by\ induction)$$

Each point could be assigned to a ring if

$$r_{i-1} < \ distance\ to\ center\ (d)\ \leq r_i$$

$$\sqrt{\frac{i-2}{n}} < d \leq \sqrt{\frac{i-1}{n}}$$

Hence, we would have n buckets and each point would be assigned to a bucket if it satisfies the above condition for a ring.