HW#1

1. Consider the sample space S consisting of n>1 rolls of an unbiased, six-sided die. Consider the event Aj⊆S consisting of n rolls where the jth roll is 1. Show Aj and Ai are pairwise independent assuming i≠j. Consider the event E that the sum of the n rolls is 0 mod 6. Show for each j, Aj and E are pairwise independents. Show that the events E,A1,...,An are not mutually independent.

Solution:

1. Experiment: Rolling an unbiased six-side die ($n > 1$) times where sides of the die are numbered from 1 - 6 (inclusive)
2. let n rolls of the die be represented by a n-sized tuple $<B_1, B_2, \dots B_n>$ where $1 \le B_i \le 6$
3. The sample space (S) consists of all possible values of the tuple. Each instance of the tuple is an elementary event.
   $|S| = 6^n$
4. $A_i \subseteq S$ represents an event where j^{th} value of the tuple is 1.
   For $n = 2$, $A_2 = \{\{1,1\}, \{2,1\}\}$
5. To prove: A_i and A_j are pairwise independent for $i \neq j$
   i.e $P_r(A_i \cap A_j) = P_r(A_i) \cdot P_r(A_j)$ for $i \neq j$. — ①

   a. $P_r(A_i) = P_r(A_j) = \frac{6^{n-1}}{6^n} = \frac{1}{6}$ —② → Since there are 6^{n-1} permutations where j^{th} roll is 1 & $|S| = 6^n$

   b. $P_r(A_i \cap A_j) = \frac{6^{n-2}}{|S|} = \frac{6^{n-2}}{6^n} = \frac{1}{6^2}$ → There are 6^{n-2} elementary events where at least 2 rolls are 1 (i.e event $A_i \cap A_j$ occurs)

   c. Thus, LHS of ① = $P_r(A_i \cap A_j) = 1/36$
      RHS of ① = $P_r(A_i) \cdot P_r(A_j) = 1/6 \cdot 1/6 = \frac{1}{36}$
      LHS = RHS
      Hence proved that $P_r(A_i \cap A_j) = P_r(A_i) \cdot P_r(A_j)$ for $i \neq j$
      implying that A_i and A_j are pairwise independent for $i \neq j$

6. <u>To prove:</u> For each j, A_j and E are pairwise independent.

a. Let us consider a random variable X which represents the modular sum of the n dice rolls.
For the permutation $\{2, 1\}$ of die values, $X = (2+1) \bmod 6 = 3$

b. We know that

$$P_r(E) = P_r(X=0) = \sum_{s \in S \text{ and } X(s)=0} P_r(s)$$

We therefore need to calculate the number of events $s \in S$ such that $X(s) = 0$

c. Let $MSUM(a, b)$ be the modular sum of die values for rolls numbered a to b (a and b inclusive)

d. We know that

$$X = MSUM(1, n)$$
$$= (MSUM(1, n-1) + MSUM(n, n)) \bmod 6$$

$\rightarrow$ Since $(A+B) \bmod C = (A \bmod C + B \bmod C) \bmod C$

Now,

$$0 \le MSUM(1, n-1),\ MSUM(n, n) \le 5$$

For all possible values of $MSUM(1, n-1)$, there is exactly 1 value of the n^{th} die roll (since $MSUM(n, n)$ is the value of the n^{th} roll modulo 6)

e. This implies that for all possible values of $MSUM(1, n-1)$, we can find exactly 1 value of the n^{th} die roll.

f. Therefore, for any value of X,
number of possible events = number of permutations of $n-1$ die rolls
$= 6^{n-1}$

g. We can thus conclude two things:
(I) For n die rolls, for $X=a$, number of permutations $= 6^{n-1}$ $\forall\ 0 \le a \le 5$
(II) $P_r(X=a) = \frac{6^{n-1}}{6^n} = \frac{1}{6}$ for n die rolls $\forall\ 0 \le a \le 5$

h. To show that A_j and E are pairwise independent, we must show:

$$Pr(E \cap A_j) = Pr(E) \cdot Pr(A_j) \quad — ③$$

i. Calculating LHS of ③

$E \cap A_j$ implies that the event should have a modular sum of 5 for the $n-1$ die rolls excluding j. since 5 is the additive inverse of 1 mod 6

$$\therefore Pr(E \cap A_j) = \frac{\left(\text{number of permutations of } n-1 \text{ die rolls such that } X=5\right)}{|S|}$$

$$= \frac{6^{n-2}}{6^n} \longrightarrow \text{from } 6.g.\,Ⓘ$$

$$\boxed{Pr(E \cap A_j) = \frac{1}{36}} \quad — ④$$

j. Calculating RHS of ③

$Pr(E) = Pr(X=0) = 1/6$ — ⑤ → from ~~6.g.②~~ 6.g. II

$Pr(A_j) = 1/6$ → from ②

$$\therefore \boxed{Pr(E) \cdot Pr(A_j) = \frac{1}{36}} \quad — ⑤$$

h. From ④ & ⑤, we have proved ③

LHS = RHS.

i.e $Pr(E \cap A_j) = Pr(E) \cdot Pr(A_j)$

OR

A_j and E are pairwise independent for all j.

7. <u>To prove:</u> Events $E, A_1, \ldots A_n$ are not mutually independent.

a. We must show that

$$Pr(E \cap A_1 \cap A_2 \ldots \cap A_n) \neq Pr(E) \cdot Pr(A_1) \cdot Pr(A_2) \ldots Pr(A_n) \rightarrow ⑥$$

b. Calculating LHS of ⑥:

$E \cap A_1 \cap A_2 \ldots \cap A_n$ is the event where all n rolls have a value 1. and $X = MSUM(1, n) = 0$

i.e $X = n \bmod 6 = 0$

$\therefore Pr(E \cap A_1 \ldots A_n) = Pr(X = 0)$

$= Pr(n \bmod 6 = 0)$

$\boxed{Pr(E \cap A_1 \ldots A_n) = 0 \text{ or } \frac{1}{6^n}} \rightarrow ⑦$ depending on the value of n

If $n \bmod 6 = 0$ i.e n is a multiple of $6 \rightarrow Pr(X=0) = \frac{1}{6^n}$

else if n is not a multiple of 6 or $n < 6 \rightarrow Pr(X = 0) = 0$

c. Calculating RHS of ⑥

$Pr(E) = 1/6$ — from ⑤

$Pr(A_1) = Pr(A_2) = \ldots = Pr(A_n) = 1/6$ from ②

$\therefore$ RHS $= Pr(E) \cdot Pr(A_1) \cdot Pr(A_2) \ldots Pr(A_n)$

$$= \frac{1}{6} \cdot \frac{1}{6} \cdot \ldots \frac{1}{6} = \prod_{i=1}^{n+1} \frac{1}{6} = \frac{1}{6^{n+1}} \rightarrow ⑧$$

d. From ⑦, ⑧, LHS $\neq$ RHS

$$Pr(E \cap A_1 \cap A_2 \ldots A_n) \neq Pr(E) \cdot Pr(A_1) \ldots Pr(A_n)$$

OR Events $E, A_1, \ldots A_n$ are not mutually independent.

2. Our analysis of the hiring problem shows that it can be useful to be able to pick a function uniformly at random from some space of functions. For the Hiring Problem, the space of functions were permutations of 1,...,n. Call a function of n inputs f symmetric if for any permutation π of 1,...,n, $f(x_1,\ldots,x_n)=f(\pi(x_1),\ldots\pi(x_n))$. For example, the threshold function $T_{nk}(x_1,\ldots x_n)$ which is equal 1 if $\sum_i x_i \geq k$ and 0 otherwise is a symmetric function. Let $\mathcal{F}_{nk}$ be the space of all symmetric functions from $(\mathbb{Z}_k)^n \to \mathbb{Z}_k$. Give pseudo-code for an algorithm to generate an element of $\mathcal{F}_{nk}$ uniformly at random.

   Solution:

```
inputs = []                              // list of tuples <A1, A2, A3 ... An> representing
                                         // list of all possible inputs to the function where 0 <= Ai < k
// consider this function as the entry point.
function pickRandomSymmetricFunction(n, k):
    domainSize = choose(n+k-1, n)        // choose(n+k-1, n) calculates (n+k-1)Cn. this is the
                                         // total number of combinations of input tuples possible.


    outputs = []                         // list of outputs of chosen symmetric function on each tuple in input.
                                         // outputs[i] = f(inputs[i]) where f is the chosen symmetric function
    for counter = 1 to domainSize:       // note both 1 and domainSize are inclusive in the for loop range.
        outputs[counter] = RANDOM(0, k-1)   // this is the random function introduced in class.

    generateAllInputs(n, k)              // pseudo-code below. This call populates the inputs list.

    f = []                               // list of tuples <A1, A2, A3 ... An, OP> of size n+1
                                         // where the n+1th value is OP and it represents output of the chosen
                                         // symmetric function f on the input tuple <A1, A2, ... An>
                                         // Essentially, f stores all input-output mappings

    for counter = 1 to domainSize:
        f[counter] = inputs[counter]
                    .append(outputs[counter]) // appending output integer OP to input tuple <A1, A2, ... An>

    return f

// This function generates all possible inputs given n and k.
function generateAllInputs(n, k):
    array anyGivenInput[n]
    generateAllInputs(n, k, 0, 0, anyGivenInput)

// This function generates all possible inputs given n and k using backtracking.
function generateAllInputs(n, k, index, lower, anyGivenInput):
    if index == n:
        array temp[n] = copy(anyGivenInput)
        inputs.append(temp)
        return

    for counter = lower to k-1:
        anyGivenInput[counter] = counter
        generateAllInputs(index+1, counter, anyGivenInput)

    return
```

3. Consider the Hiring Problem from class. Candidate i had a rank(i) which was a number from 1 to n. Each candidate had a distinct rank. Rather than using a rank suppose the candidates had a fitness number fit(i) saying how good the candidate was. Suppose fit(i) is equally likely to be any integer number in the range 1 to $\lceil f(n) \rceil$. We only hire a candidate if their fitness number is better than the current best candidate. Calculate how this would affect the analysis of the expected number of candidates we hire for $f(n)=n^2$ and for $f(n)=\log^2 n$.

PAGE No.
DATE / /

HW1 3

Similar to the original hiring problem, let X_i be the indicator random variable which is 1 if i^{th} candidate is hired

$X = \sum_{i=1}^{n} X_i$; $E[X_i] = P_r(\text{candidate } i \text{ is hired})$

$$P_r(i^{th} \text{ candidate is hired}) = P_r(\text{fitness of } j^{th} < \text{fitness of } i^{th})^{i-1}$$

The above is based on the fact that for a given i & a j where the j^{th} comes before i^{th} candidate, we need to have fit(j) < fit(i) for the i^{th} candidate to get hired. j can be anything from 1^{st} candidate to $(i-1)^{th}$ candidate. These are independent hence Pr(fit(j) < fit(i)) is powered (i-1) times

$$P_r(fit(j) < fit(i)) = \frac{[f(n)-1] + [f(n)-2] + \cdots [f(n)-f(n)]}{f(n)\cdot f(n)}$$

$\lceil f(n) \rceil$ is used as $f(n)$

The denominator is all possible cases where each i & j have f(n) options. When fit(i) = f(n), there are [f(n) - 1] ways to choose fit(j) and so on.

$$P_r = \frac{f(n)\cdot f(n) - \left\{\sum_{k=1}^{f(n)} k\right\}}{f(n)\cdot f(n)} = \frac{f(n)^2 - \frac{f(n)(f(n)+1)}{2}}{f(n)^2}$$

$$P_r = \frac{f(n)^2 - f(n)}{2f(n)^2} = \frac{1}{2} - \frac{1}{2f(n)} = \frac{1}{2}\left(1 - \frac{1}{f(n)}\right)$$

It is approximately 1/2, irrespective of $f(n) = n/2$ or $f(n) = \log_2 n$

$$E[X] = E\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} P_r(i \text{ is hired}) = \sum_{i=1}^{n}\left(\frac{1}{2} - \frac{1}{2f(n)}\right)^{i-1}$$

$=$

DATE / / /

For $f(n) = n/2$

$$E[x] = \sum_{i=1}^{n} \left(\frac{1}{2} - \frac{1}{n}\right)^{i-1}$$

$$= -\frac{2n \cdot \left(\left(\frac{1}{2} - \frac{1}{n}\right)^{n} - 1\right)}{n+2} = \frac{2n\left(1 - \left(\frac{1}{2} - \frac{1}{n}\right)^{n}\right)}{n+2}$$

For $f(n) = \log_2 n$

$$E[x] = \sum_{i=1}^{n} \left(\frac{1}{2} - \frac{1}{2\log n}\right)^{i-1} = \frac{2\log_2 n\left(1 - \left(\frac{1}{2} - \frac{1}{2\log n}\right)^{n}\right)}{\log_2 n + 1}$$

For $E[x] = \sum_{i=1}^{n}\left(\frac{1}{2} - \frac{1}{2f(n)}\right)^{i-1}$, since $\frac{1}{2} > \frac{1}{2} - \frac{1}{2f(n)}$

we can approximate to the upper bound $E[x] \approx \sum_{i=1}^{n}\left(\frac{1}{2}\right)^{i-1}$ which is sum of a G.P.

This gives $E[x] = \underline{O}\left(2 - 2^{1-n}\right)$; which is the upper bound

$\therefore$ For both values for $f(n)$, the cost of hiring, $c_h \cdot$(avg. no. of hires), is $O\left(ch \cdot (2 - 2^{1-n})\right)$ (ch = cost of hiring)

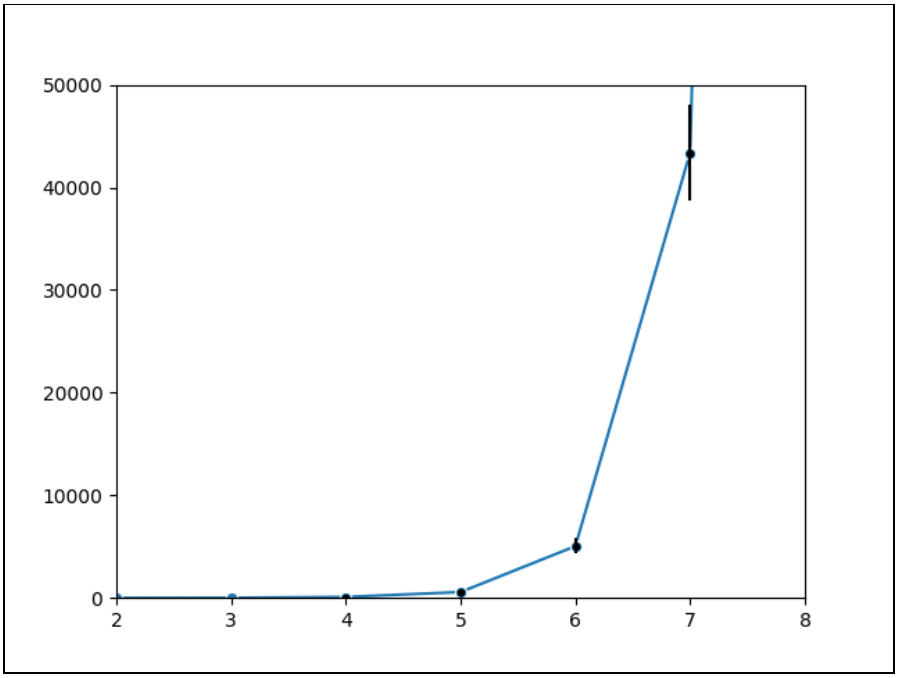
Similar to original problem, worst case is $O(ch \cdot n)$, $O(n)$ when we have strictly ascending fitnesses & we hire each candidate. Similarly, best case is $O(1)$ where we hire only once in the scenario of strictly descending fitnesses.

4. Conduct 10 experiments for each positive integer between 2 and 11. I.e., Ten experiments for 2, ten for 3, etc (seed your random number generators differently each time). You can do this programmatically if you want. Determine the average number of iterations for each integer and its standard deviation. Plot the integers and the number of trials with the standard deviations on a clearly labeled graph where you fit the curve that you would expect based on the coupon collector problem to the data using χ2 minimization. How well does the curve fit? Write up your experiments and put the graphs in your Hw1.pdf

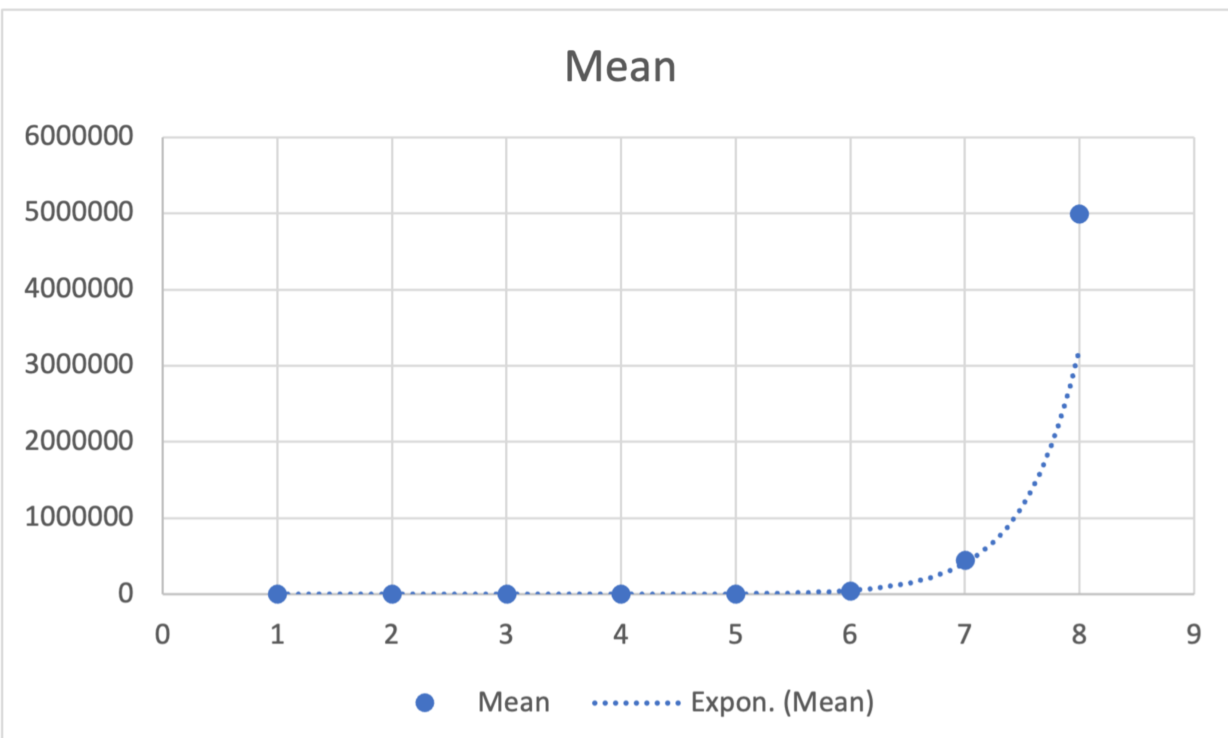
The data generated for integers {2,3,...,10} for 10 iterations each is as follows -
(refer to last 2 rows for Mean and SD calculations)

n	2	3	4	5	6	7	8	9	10
Iterations	2	17	106	583	4259	46877	466934	4776822	57796847
	3	9	110	580	4889	47063	428980	5352107	54312263
	3	17	134	446	6276	48777	404519	4569954	56926892
	3	25	84	855	4309	39759	387009	5980844	57119699
	2	12	92	603	6034	38207	478346	4503388	50874130
	4	26	64	524	4525	42017	386805	4957277	52689406
	2	8	127	545	4697	37693	459119	5057454	53351737
	2	14	93	605	5845	44018	427716	4924279	53843218
	2	25	56	481	4256	51040	451136	4724106	55351226
	2	14	73	492	5343	38315	582278	5065922	56214070
Mean	2.5	16.7	93.9	571.4	5043.3	43376.6	447284.2	4991215.3	54847948.8
SD	0.70710678	6.63408706	25.774449	113.485388	776.037664	4873.15969	57316.6922	429205.615	2218209.12

Plot the integers and the number of trials with the standard deviations on a clearly labeled graph



Now we need to fit the curve -

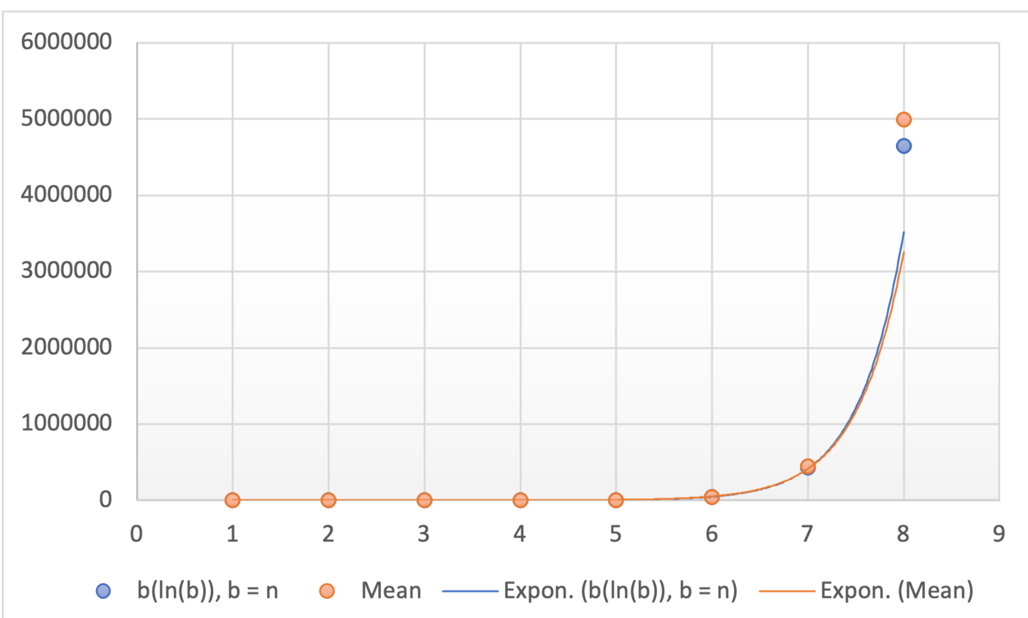


As per the coupon collector problem the value of distinct coupons is b(lnb) here b is no of distinct permutation of a value n

We plotted a graph to show a correlation between the Coupon Collector Value and Mean of iterations for each integer call Y-axis: Coupon collector value and Mean of iterations call X-axis: n

n	n!	Value using Coupon Collectors Formula b(ln(b)), b = n	Mean
2	2	1.386294361	2.5
3	6	10.75055682	16.7
4	24	76.27329193	93.9
5	120	574.4990091	571.4
6	720	4737.060873	5043.3
7	5040	42966.81326	43376.6
8	40320	427577.589	447284
9	362880	4645527.156	4991215

Plotting these values against n as (x-axis) gives us the following graph



We can see that The average value of random permutations calls is close to the Coupon collector value.