CS6140: Asmt1 - Hash Functions and PAC Algorithms

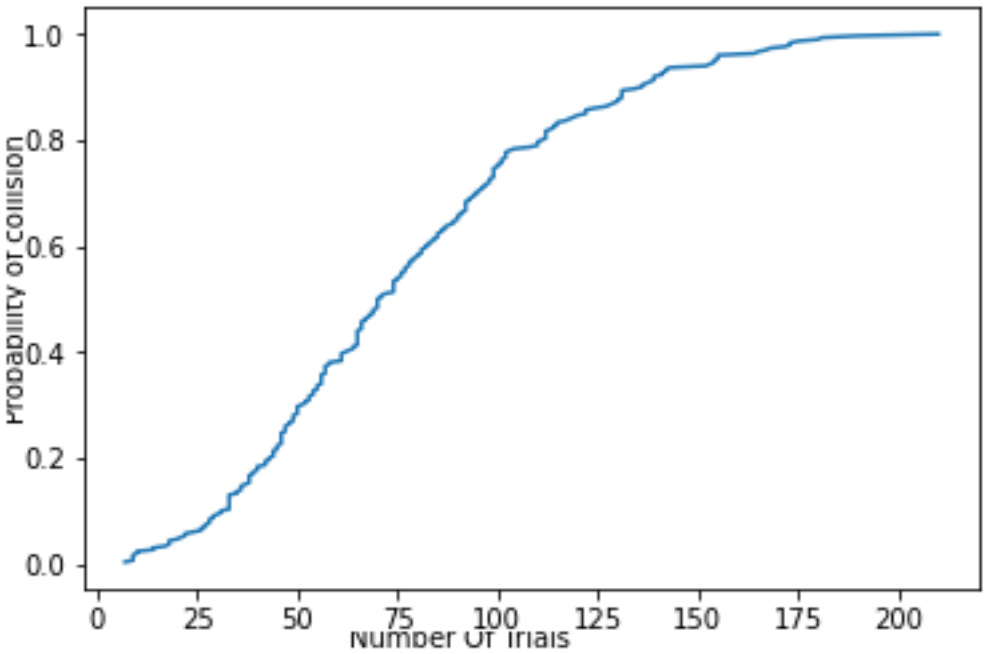
Shuvrajit Mukherjee

January 2018

1 Birthday Paradox

Problem 1A:
Solution: the number of random trials taken while generating numbers randomly in the domain [4000] until two has the same value is **27**

Problem 1B:
Solution: The CDF for the random trial when the experiment is repeated m = 300 times is plotted below:

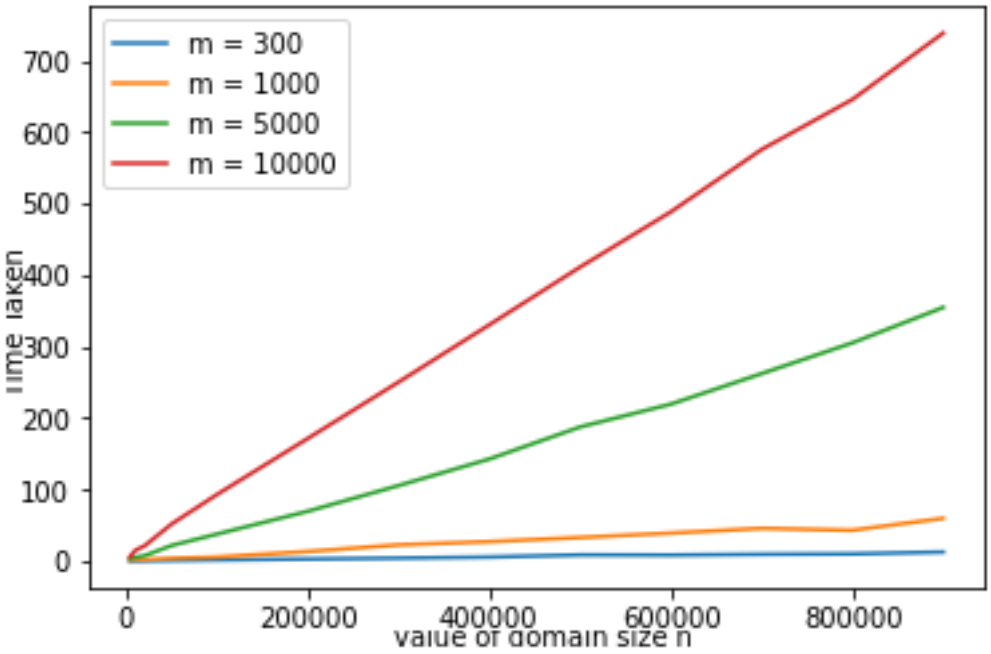


Problem 1C:
Solution: The empirical estimate of the expected value of k is **77.65**

Problem 1D:
Solution: The steps of the experiment is described below:
1. Create a function to get the value of k:
   a. Initialize an array of length and define each element to be False.
   b. Generate a random integer (r_1) in the range [0,n]
   c. Mark the r_1^{th} element of the array to be True
   d. Keep on repeating step b and c and keep an counter for k, until we find such an r_1 for which the array element was already True. At this point we can say that we have found two random numbers with same values.
2. For Problem 1B we call the function described in step.1 for m = 300 times and maintain a dataframe with columns like "serial number","value of k", "cumulative fraction". the dataframe is sorted based on the values of k.
3. Repeated steps.1 - step.2 for different values of m (300, 1000, 5000, 10000) and plotted them in a single frame.

The time taken for m = 300 trials in the domain [n = 4000] is **0.11850261688232422 seconds**

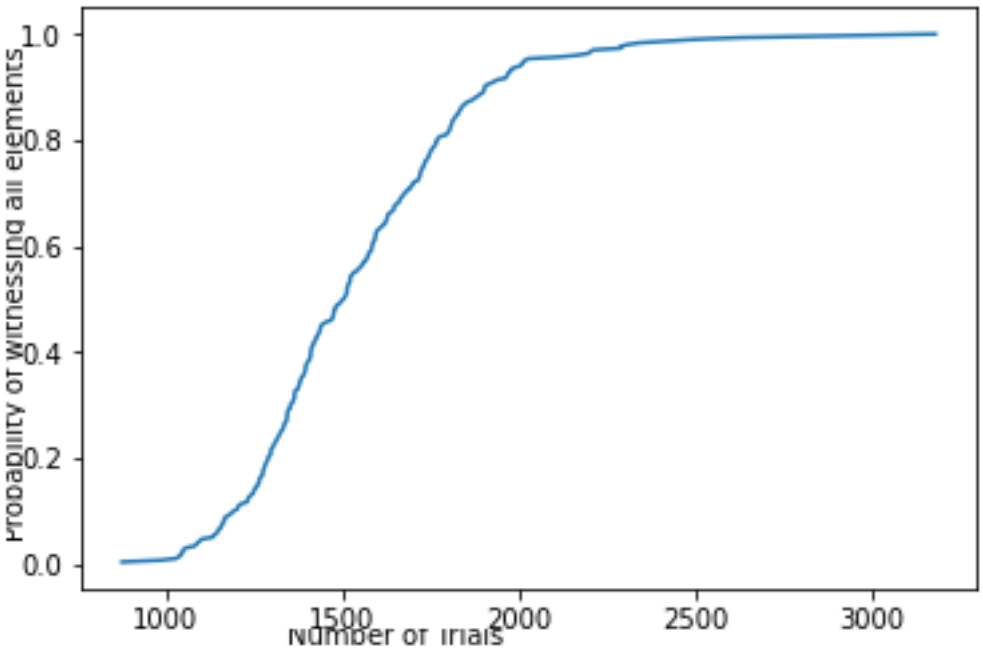
Plot of Runtime(seconds) Vs [n] for different values of m:



2 Coupon Collectors

Problem 2A:
Solution: The number of random trials taken while generating numbers randomly in the domain [250] until each values in the domain has generated at least once is **1789**

Problem 2B:
Solution: The CDF for the random trial when the experiment is repeated m = 300 times is plotted below:

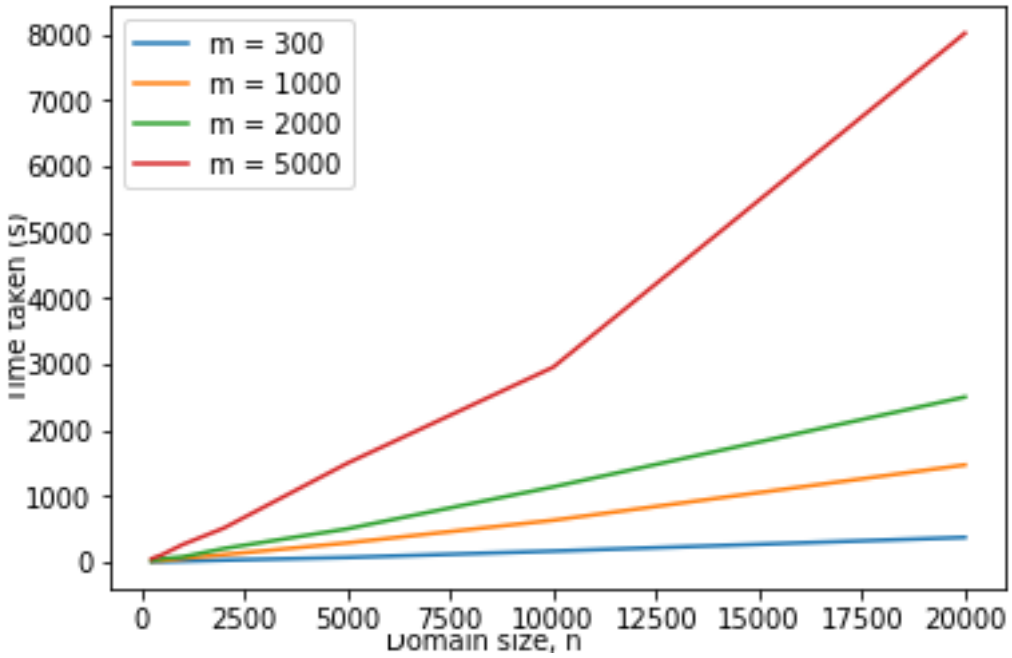


Problem 2C:
Solution: The empirical estimate of the expected value of k is **1470.35**

Problem 2D:
Solution: The steps of the experiment is described below:
1. Create a function to get the value of k:
   a. Initialize an array of length and define each element to be 0.
   b. Generate a random integer (r_1) in the range [0,n]
   c. Define a counter variable to 0. This counter is incremented when the we generate a random number which has not been generated before. We check this by checking the value of the array at the index of the random number. If the value at that index is 0, this means that this particular number has not discovered before and we set the value to 1 and increment the counter.
   d. If the value of the counter is equal to the length of the array we can say all the numbers has been discovered at least once, and we terminate the process.
   e. At every iteration of generating a random number we increment the value of k.
2. For Problem 1B we call the function described in step.1 for m = 300 times and maintain a dataframe with columns like "serial number","value of k", "cumulative fraction". the dataframe is sorted based on the values of k.
3. Repeated steps.1 - step.2 for different values of m (300, 1000, 2000, 5000) and plotted them in a single frame.

The time taken for m = 300 trials in the domain [n = 250] is **1.4114 seconds**

Plot of Runtime Vs [n] for different values of m:


m = 300
m = 1000
m = 2000
m = 5000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
2500
5000
7500
10000
12500
15000
17500
20000

3 Comparing Experiments to Analysis

Problem 3A

Solution: The probability that the two randomly number drawn from n distinct numbers are same is $\frac{1}{n}$.
The probability that the two numbers are different is $(1 - \frac{1}{n})$.
Let us assume that k such trials are performed.
Then the probability that in none of the k trials no two numbers are same is: $(1 - \frac{1}{n})^{\binom{k}{2}}$.
Then the probability that we got two same numbers in k trials is $(1 - (1 - \frac{1}{n})^{\binom{k}{2}})$.
We want this probability to be greater than $\frac{1}{2}$

$$(1 - (1 - \frac{1}{n})^{\binom{k}{2}}) > \frac{1}{2}$$
$$(1 - \frac{1}{n})^{\binom{k}{2}} < \frac{1}{2}$$

In our case $n = 4000$, we will try different values of k to make the probability higher than $\frac{1}{2}$

$$(1 - \frac{1}{4000})^{\frac{k(k-1)}{2}} < \frac{1}{2}$$
$$(0.99975)^{\frac{k(k-1)}{2}} < \frac{1}{2}$$

For $k = 74$, $(0.99975)^{\frac{74*73}{2}} \approx 0.508986$
For $k = 75$, $(0.99975)^{\frac{75*74}{2}} \approx 0.499655$

So we need at least $k = 75$ trials to have a probability of success greater than $\frac{1}{2}$

The empirical estimate for k from Q1c is 77.65 which is very close to the calculated estimation for k.

Problem 3B

Solution: Let r_i be the expected number of trials we need to take before witnessing exactly i distinct coupons. Let $r_0 = 0$, and $t_i = r_i - r_{i-1}$ to measure the expected number of trials between witnessing $i - 1$ distinct numbers and i distinct numbers.

Clearly, $r_1 = t_1 = 1$, because the first trial always yields a new coupon.
Then the expected number of trials to witness all the numbers in the given domain is $T = \sum_{i=1}^{n} t_i$
To measure t_i we will define p_i as the probability that we witness a new number after already witnessing $i - 1$ distinct numbers. Thus $t_i = \frac{1}{p_i}$. And $p_i = \frac{n-(i-1)}{n}$

$$T = \sum_{i=1}^{n} t_i = \sum_{i=1}^{n} \frac{n}{n - i + 1} = n \sum_{i=1}^{n} \frac{1}{i}$$

$n \sum_{i=1}^{n} \frac{1}{i}$ is known as the n^{th} Harmonic Number H_n.
It is known that, $H_n = \gamma + \ln n + o(\frac{1}{n})$, where $\gamma = 0.577$ is the $Euler - Masheroni\ constant$. Thus,

$$k = T = nH_n \approx n(\gamma + \ln n)$$

Replacing γ with 0.577 and n with 250, we get the expected number of trials as:

$$k = 250(0.577 + \ln 250)$$
$$k = 1524$$

So the expected number of trials to witness all the numbers in range 0 to 250 is 1524.

The empirical average as calculated in experiment Q2.C is 1470, which is close to the logically calculated expectation.

4 Random Numbers

Problem 4A
Solution: Any number between 1 and 1024 can be represented using 10 binary bits. So we can call rand-bit() 10 times to generate a random number in that range. The least significant bits to the most significant bits, each of them have equal probability ($=\frac{1}{2}$) of being 0 or 1. Now we have 10 random bits, which we can now convert to decimal number system to get a random integer between 0 and 1024.

Problem 4B
Solution: In order to generate a number between 1 and 1000 we will follow the algorithm described in Q4.A, i.e call the rand-bit() 10 times. Now check if the result is less than 1000, if so we get our result, if not repeat the process untill we get a number below 1000. This process may run forever, because we may get a number which is greater than 1000 everytime. But we can calculate the expected number of trials required and we also know when it's finished and it's correct.
Let the expected number of trials be E.
The probability that the number will be less than 1000 is $\frac{1000}{1024}$. So with this probability the expected number of trial is 1. but if this fails, then the expected number of trails will be (E+1) with probability $\frac{24}{1024}$

$$E = \frac{1000}{1024} + (E+1)\frac{24}{1024}$$
$$E = 1.024$$

And in each trial we cann rand-bit() 10 times, so the expected number of rand-bit() call is 10.24

Problem 4C
Solution: In our calculation from Q4.B, where we have assumed n to be 1000. Now we will calculate the expectation for any positive integer n.

Number of bits required to represent a decimal number n in binary is $\lceil \log_2 n \rceil$. So there would be $2^{\lceil \log_2 n \rceil}$ possible numbers using $\lceil \log_2 n \rceil$ bits. So the probability that a number is less than n is $n/2^{\lceil \log_2 n \rceil}$.

let's assume the probability of success, p = $n/2^{\lceil \log_2 n \rceil}$.
Now we can calculate the expected number of trials using the similar approach described in Q4.B.

Let's assume the expected number of trials for success be E.

$$E = p + (E+1) * (1-p)$$
$$E = \frac{1}{p}$$
$$E = \frac{2^{\lceil \log_2 n \rceil}}{n}$$

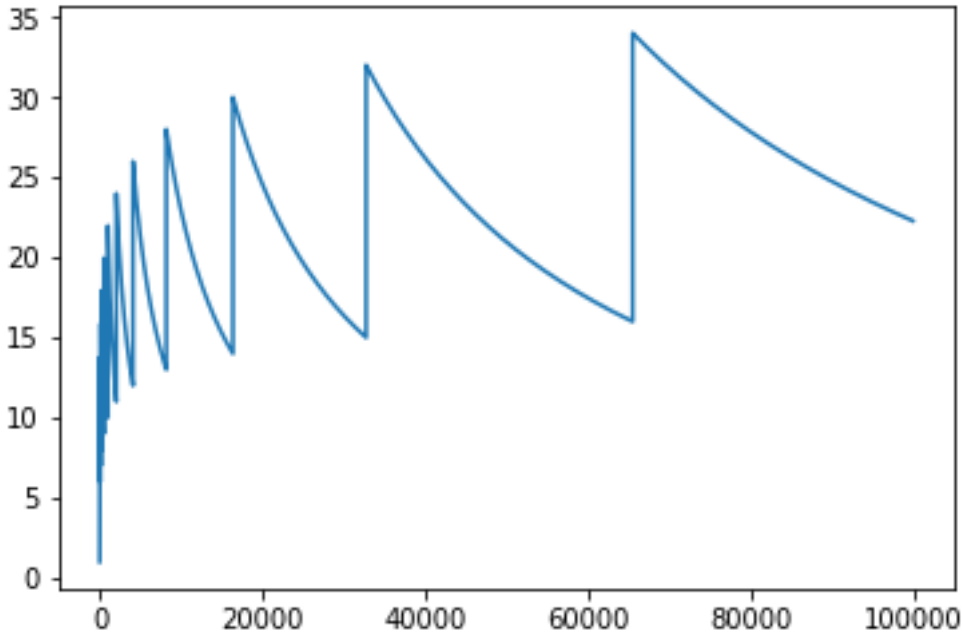
Now each of these trials calls rand-bit() $\lceil \log_2 n \rceil$. So the expected number of rand-bit() call is $\lceil \log_2 n \rceil * \frac{2^{\lceil \log_2 n \rceil}}{n}$.

In order to analyse the number of rand-bit() calls with increasing n we need to consider the following two scenarios:

1. For the range where $\lceil \log_2 n \rceil$ is fixed the rand-bit() call decreases with increasing n.

2. As soon as $\lceil \log_2 n \rceil$ increases to next integer the number of calls suddenly increases and thereafter decrases with increasing n.

So the plot with expected number of rand-bit() calls against n will look like the following:


35
30
25
20
15
10
5
0
0
20000
40000
60000
80000
100000

5 BONUS: PAC Bounds

Solution:
Let X_j be the j^{th} (where $j \in [1, k]$) random trial of picking a random number $i \in [n]$
We are required to calculate the bound on the probability that μ is deviated from the expected value of a number being picked by ϵ

Let $A = \frac{1}{k} \sum_{i=0}^{k} X_i$

Given $\mu = max_{i \in [n]} f_i/k$

According to Chernoff bound:

$$Pr[|A - E[X_i]| > \epsilon] \leq 2exp(\frac{-k\epsilon^2}{2\Delta^2})$$

Domain size = n. Therefore, $\Delta = n$.

Hence,

$$Pr[|A - 1/n| \geq \epsilon] \leq 2 \times exp\Big(\frac{-k\epsilon^2}{2n^2}\Big)$$

According to the question, $Pr[|\mu - 1/n| \geq \epsilon] \leq 0.02$
Therefore,

$$0.02 \approx 2 \times exp\Big(\frac{-k\epsilon^2}{2n^2}\Big)$$

$$k = \frac{9.2(n)^2}{\epsilon^2}$$

For only 0.002 % probability of failure
we will have the value for k to be,

$$ln(0.001) = \frac{-k\epsilon^2}{2n^2}$$

$$-6.9 = \frac{-k\epsilon^2}{2n^2}$$

$$k = \frac{13.8(n)^2}{\epsilon^2}$$

This shows that the number of trials required(k) increases as the value of ϵ decreases.
Also this is to notice that we were required to calculate the probability $Pr[|\mu - 1/n| \geq \epsilon]$. but we have calculated the probability $Pr[|A - 1/n| \geq \epsilon]$. Now since $\mu = max_{i \in [n]} f_i/k$, we can say that $\mu \geq A$. So $Pr[|\mu - 1/n| \geq \epsilon] \leq Pr[|A - 1/n| \geq \epsilon]$. Hence the calculated values of k is a lighter upper bound.