Assignment 3 Report

Running the code:

We have implemented this assignment in python3 and used matplotlib library to plot the graph.

We have used the following code files:

* assignment3.py – to run the randomness test for individual N.
* assignment3\_all.py – to run the randomness test on all N in one go.

To run the code, use the following command

**python3 assignment3.py <number of output bytes> <Number of runs> <name of plot>**

**python3 assignment3\_all.py <Number of runs> <name of plot>**

Examples:

**python3 assignment3.py 2 60 plot\_2\_60.png**

**python3 assignment3\_all.py 60 plot\_2\_20.png**

Plots and Observation:

Chart, histogram

Description automatically generated

Fig 1: Plot for randomness using N=2bytes over 60 runs.

Chart, histogram

Description automatically generated

Fig 2: Plot for randomness using N=4bytes over 60 runs.

Chart, histogram

Description automatically generated

Fig 3: Plot for randomness using N=8bytes over 60 runs.

Chart

Description automatically generated

Fig 4: Plot for randomness using N=32bytes over 60 runs.

Chart, histogram

Description automatically generated

Fig 5: Plot for randomness using N=128bytes over 60 runs.

Chart, histogram

Description automatically generated

Fig 6: Plot for randomness using N=1024bytes over 60 runs.

Chart, line chart

Description automatically generated

Fig 7: Plot for randomness using all the values of N over 60 runs.

Que: If the two original bitstreams were very similar, you would expect the counters for lots of zeros to have higher values than the counters for lots of ones.

Ans: After comparing the outputs for every iterations we found out that we are getting 5010 % of bits, where the bits are different in two equal length output and this is due to the reason that 50% of the bits should be different for random value in an ideal case.

Text

Description automatically generated

Fig7 : Showing the number of different bits between two outputs of 4bytes (32 bits)

Que: Numerical measure of the randomness.

Ans: We can see the following observations from the above plots.

1. As we can see from the above plots the randomness on outputs ranging from short through long (i.e., 2 bytes, 4 bytes, 8 bytes, 32 bytes, 128 bytes, 1024 bytes) is increasing as the value of R is decreasing. This is due to the fact that in the **pseudoRandomGeneration()** function we are shuffling the bits with more number of time for 1024bytes as compared to 2bytes.
2. Another observation we can see from the above plots is that if we increase the number of flip bits then the randomness is increasing but after a certain number of flips it will not vary much and seems stagnant.
3. From fig7 we can conclude that as the values of N is increasing the plot is more closer to each other as compared to the lower values of N. This can be justified by the reason that as the value of N increases the randomness increase and after a certain value of N there will not be very drastic change in randomness thus the values lies close to each other.
4. Lastly we observed that if we decrease the number of runs (say 10) then the plot we are getting was very jumbled and if we increase the number of runs (say 60+) then the plot we are getting is quiet intuitive and values are not any zig-zag pattern.

Que: How many bits need to be flipped (on average) before the differential bitstream looks random?

Ans: For fig1, fig 2, and fig 3 we can conclude that we are getting good amount of randomness after flipping at least 3 random bits. For fig4, fig5 and fig6 we can conclude that we are getting good amount of randomness after flipping at least 2 random bits. We assume this trend is correct as for fig4, fig5 and fig6 we are swapping the outputs by 32, 128 and 1024 times respectively which means we will achieve randomness quiet early and vice versa for fig1, firg2, and fig3.