Examining the suitability of random-number generators in Monte Carlo simulations of Bridge deals

R Geng, T McManus, M Magdon-IsmailRensselaer Polytechnic Institute

***Abstract*— This paper reviews methods of determining whether several random number generators are suitable for Monte Carlo simulations. The simulations are feature transforms of high-dimensional spaces specifically in the context of simulating Bridge deals. The authors identify two methods for comparing experimental results from a Monte Carlo simulation to either known theoretical probabilities or to results from other simulations. In this paper, we apply these methods to several deals from the game Bridge and compare certain statistics from the bridge deals to known theoretical results.**

I. Introduction

Random numbers are ubiquitous in daily life, from science and cryptography to art, gambling, and gaming. Jurors are randomly selected, scientific trials are randomized, and cards are dealt. However, true random numbers are very difficult to come by, especially when a large quantity of numbers is necessary. Therefore, people use mathematical formulas to create large amounts of random-looking (pseudo-random) numbers very quickly. With all uses of generated "random" numbers, it is important to be sure that the generated pseudo-random numbers are as close to random as possible - even indistinguishable from a random series of numbers.

There exist many tests for the randomness of a series of bits or numbers, but frequently we encounter places where random numbers have already been used (generated passwords, decks of cards, or really anything that was randomly generated without giving away the algorithm) and these earlier tests of randomness aren't useful.

For this paper, we've decided to use derived features of Bridge deals as our black-box. The Bridge deals will be randomly generated by several methods including using bits from random number generator and open source software. We have designed two experiments:
A zeroth-order probability test that simply compares the means of a series of deals to the known population mean will check for large-scale periodicity and statistical anomalies that could occur because of poor generation of individual decks.
A test that compares short-term discrepancies between sample values and cumulative values in a timeseries of decks will check for any sort of auto-correcting or short-term periodic behavior in a generator. This is more valuable for examining bridge deals used in online games or tournaments, to make the games more 'fair' or more 'exciting' than a truly random deck would.

II. Related Work

In order to generate large amount of relative random, independent Bridge deals, we can rely on exist software. The Big Deal [1] is one of them. It can accept the random keyboard inputs and desired number of Bridge deals from user as initial seed and output these deals. It is open-source software so the design-of-art and implementation are all presented in the website.

The Big Deal implemented its own pseudo random number generator(PRNG). Normally the PRNG generates binary bits and these binary bits are converted to Bridge deals. There also exists other widely-using PRNG and true RNG. The Mersenne Twister(MT) [2] is a very common and well-tested PRNG. The Python `random` library is implemented by using this PRNG. The Halton sequences [3] are low discrepancy, multiple dimensions sequences for generating random number. They generalise the one-dimensional van der Corput sequences [4]. Except the pseudo random number generator, we also have Intel random number generator [5] by using thermal noise within CPU.

III. Model

Feature	Probability	Description
ls4	3.5080E-01	Length of longest suit = 4
ls5	4.4340E-01	Length of longest suit = 5
ls6	1.6548E-01	Length of longest suit = 6
ls7	3.5266E-02	Length of longest suit = 7
ls8	4.6680E-03	Length of longest suit = 8
ls9	3.7000E-04	Length of longest suit = 9
ls10	1.6464E-05	Length of longest suit = 10
ls11	3.6407E-07	Length of longest suit = 11
ls12	3.0000E-09	Length of longest suit = 12
ls13	6.0000E-12	Length of longest suit = 13
ss0	5.1066E-02	Length of shortest suit = 0 (aka void)
ss1	3.0550E-01	Length of shortest suit = 1 (aka singleton)
ss2	5.3800E-01	Length of shortest suit = 2
ss3	1.0540E-01	Length of shortest suit = 3
6-5	1.3564E-02	Distribution of suits in hand is 6-5-X-X
6-6	7.2300E-04	Distribution of suits in hand is 6-6-1-0
7-5	1.0850E-03	Distribution of suits in hand is 7-5-1-0
7-6	5.5646E-05	Distribution of suits in hand is 7-6-0-0
p<8	2.8585E-01	Total points in hand < 8
p8-13	5.1540E-01	Total points in hand is in 8-13 inclusive
p14-18	1.7394E-01	Total points in hand is in 14-18 inclusive
p19+	2.4813E-02	Total points in hand > 19

Table (I) Features

We use a series of features of Bridge hands that are relevant to Bridge players as a way of reducing the dimensionality of a deck of cards (52) to a reasonable number (20) of features. All these features are obtained by shuffling a deck using the Fisher-Yates shuffle and then splitting it into 4 hands (done by partitioning the deck). Note that each deck has 4 nonindependent hands for the purpose of increasing sample sizes; however, this isn't important in the large scheme of things when working with millions of decks with the specific tests here.

Feature	$\tilde{P}$	**badLCG**	goodLCG	Halton	MT	**RANDU**	**vdc19**	BCryptGen
ls4	3.5080E-01	**3.2239E-01**	3.5082E-01	3.5081E-01	3.5081E-01	**3.5035E-01**	**2.5506E-01**	3.5081E-01
ls5	4.4340E-01	**4.4977E-01**	4.4340E-01	4.4339E-01	4.4338E-01	**4.4319E-01**	**5.4558E-01**	4.4338E-01
ls6	1.6548E-01	**1.7746E-01**	1.6546E-01	1.6548E-01	1.6549E-01	**1.6580E-01**	**1.7176E-01**	1.6549E-01
ls7	3.5266E-02	**4.3091E-02**	3.5266E-02	3.5264E-02	3.5265E-02	**3.5512E-02**	**2.6427E-02**	3.5257E-02
ls8	4.6680E-03	**6.4697E-03**	4.6677E-03	4.6654E-03	4.6688E-03	**4.7449E-03**	**1.1685E-03**	4.6705E-03
ls9	3.7000E-04	**8.2397E-04**	3.7096E-04	3.7014E-04	3.7067E-04	**3.8193E-04**	**0.0000E+00**	3.6888E-04
ls10	1.6464E-05	**0.0000E+00**	1.6455E-05	1.6421E-05	1.6444E-05	**1.7426E-05**	**0.0000E+00**	1.6160E-05
ls11	3.6407E-07	**0.0000E+00**	3.5400E-07	3.4550E-07	3.7075E-07	4.0100E-07	**0.0000E+00**	3.6500E-07
ls12	3.0000E-09	0.0000E+00	3.0000E-09	3.0000E-09	3.7500E-09	5.7500E-09	0.0000E+00	5.0000E-09
ls13	6.0000E-12	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
ss0	5.1066E-02	**6.3141E-02**	5.1061E-02	5.1065E-02	5.1067E-02	5.1057E-02	**1.3961E-01**	5.1070E-02
ss1	3.0550E-01	**3.2712E-01**	3.0555E-01	3.0554E-01	3.0553E-01	**3.0578E-01**	**3.6810E-01**	3.0554E-01
ss2	5.3800E-01	**5.1846E-01**	5.3803E-01	5.3803E-01	5.3805E-01	5.3796E-01	**4.3946E-01**	5.3803E-01
ss3	1.0540E-01	**9.1278E-02**	1.0536E-01	1.0536E-01	1.0535E-01	**1.0521E-01**	**5.2827E-02**	1.0537E-01
6-5	1.3564E-02	**1.5015E-02**	1.3561E-02	1.3566E-02	1.3567E-02	**1.3585E-02**	**1.3196E-02**	1.3569E-02
6-6	7.2300E-04	**1.0681E-03**	7.2343E-04	7.2328E-04	7.2344E-04	7.2295E-04	**8.0895E-04**	7.2231E-04
7-5	1.0850E-03	**1.8921E-03**	1.0840E-03	1.0850E-03	1.0860E-03	1.0900E-03	**7.9988E-04**	1.0827E-03
7-6	5.5646E-05	**9.1553E-05**	5.5375E-05	5.5661E-05	5.5647E-05	5.5392E-05	**0.0000E+00**	5.5827E-05
p<8	2.8585E-01	**2.8421E-01**	2.8585E-01	2.8585E-01	2.8584E-01	**2.8572E-01**	**2.6511E-01**	2.8585E-01
p8-13	5.1540E-01	**5.1785E-01**	5.1539E-01	5.1540E-01	5.1540E-01	**5.1564E-01**	**5.4393E-01**	5.1538E-01
p14-18	1.7394E-01	**1.7343E-01**	1.7395E-01	1.7394E-01	1.7394E-01	1.7389E-01	**1.7866E-01**	1.7396E-01
p19+	2.4813E-02	**2.4506E-02**	2.4814E-02	2.4813E-02	2.4813E-02	**2.4759E-02**	**1.2298E-02**	2.4812E-02

Table (II) Theoretical means and experimental results

We use a series of features of Bridge hands that are relevant to Bridge players as a way of reducing the dimensionality of a deck of cards (52) to a reasonable number (20) of features. All these features are obtained by shuffling a deck using the Fisher-Yates shuffle and then splitting it into 4 hands (done by partitioning the deck). Note that each deck has 4 nonindependent hands for the purpose of increasing sample sizes; however, this isn't important in the large scheme of things when working with millions of decks with the specific tests here.

These features and their probabilities given in table I.

Feature	$\tilde{P}$	3 Months	1 Year	BigDeal
ls4	3.5080E-01	3.5012E-01	3.5044E-01	3.4957E-01
ls5	4.4340E-01	4.4482E-01	4.4532E-01	4.4582E-01
ls6	1.6550E-01	1.6554E-01	1.6511E-01	1.6395E-01
ls7	3.5300E-02	3.4580E-02	3.4285E-02	3.5662E-02
ls8	4.7000E-03	4.5188E-03	4.4521E-03	4.6006E-03
ls9	3.7000E-04	3.9758E-04	3.6149E-04	3.7773E-04
ls10	1.7000E-05	1.9394E-05	3.8052E-05	1.9371E-05
ls11	3.0000E-07	0.0000E+00	0.0000E+00	0.0000E+00
ls12	3.0000E-09	0.0000E+00	0.0000E+00	0.0000E+00
ls13	6.0000E-12	0.0000E+00	0.0000E+00	0.0000E+00
ss0	5.1200E-02	5.1986E-02	5.3082E-02	5.2020E-02
ss1	3.0550E-01	3.0596E-01	3.0394E-01	3.0784E-01
ss2	5.3800E-01	5.3713E-01	5.3573E-01	5.3507E-01
ss3	1.0540E-01	1.0492E-01	1.0725E-01	1.0507E-01
6-5	1.3560E-02	1.3556E-02	1.3680E-02	1.3734E-02
6-6	7.2000E-04	7.5637E-04	9.3227E-04	7.3609E-04
7-5	1.0900E-03	1.2897E-03	1.2177E-03	1.2591E-03
7-6	5.6000E-05	4.8485E-05	0.0000E+00	2.9056E-05
p<8	2.8585E-01	2.8496E-01	2.8476E-01	2.8479E-01
p8-13	5.1540E-01	5.1442E-01	5.1596E-01	5.1543E-01
p14-18	1.7394E-01	1.7616E-01	1.7416E-01	1.7556E-01
p19+	2.4813E-02	2.4466E-02	2.5114E-02	2.4223E-02

Table (III) "Black box" data

IV. Zeroth order probability test

We used five different random number generators of varying quality and frequency of usage to simulate, in sequence, 1 billion deals of bridge for a given seed. Table II shows the mean value of features across many deals. Any bolded numbers are significant at 5σ to be different from the true mean values. Notice that the intentionally bad PRNGs - badLCG and vdc19 - fail greatly, while all of the 'good' generators - goodLCG, Halton, MT, and BCryptGen - all succeed. The interesting result is the RANDU generator, which failed on several features. This suggests that there is a quality in poorly made PRNGs that will cause this sort of test to fail even if they don't have a short period. It would be interesting to see if other generators that are well-distributed but fail the spectral test show similar behaviors.

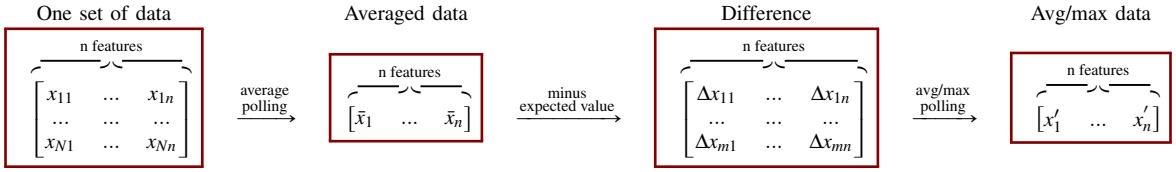
Table III shows that all of the 'black box' data that we've received lies within acceptable bounds at a 5σ level.

V. Discrepancy test

A. Data Generation

We will generate totally $n \times N \times m \times t$, n is feature number, N is row number for first polling, m is row number for second polling and t is number of data points after two polling for each feature. Then we can plot the cumulative mass function for each feature by using these t data points. In the simulation, I choose $N = 20, m = 32, t = 640000$ and $n = 22$ is feature number. Here, we give m another name–window size because we take every m points in a time series and do polling. These m points are within the "window".

In above plot, I did not include LCG-bad and one-year CMF curve to simplify the plot. There are several distinguishable observations among these plots. Firstly, in plot (a), the cumulative mass function of Halton RNG is flatter than CMF of other RNGs. For the lower part, the halton RNG CMF



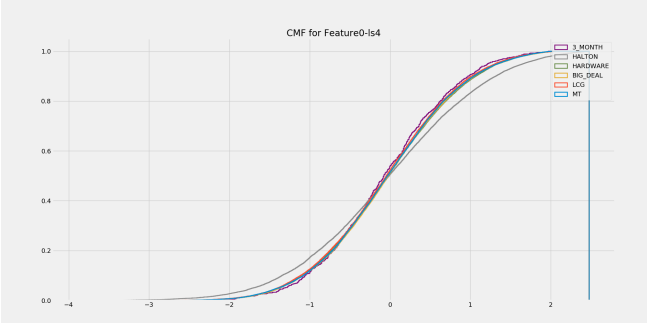


Figure (1) Interpolation for Data 1

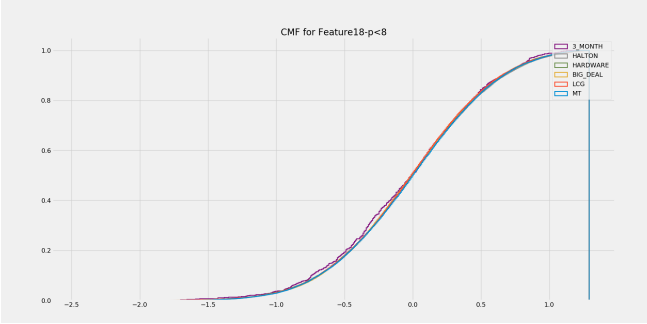


Figure (2) Interpolation for Data 2

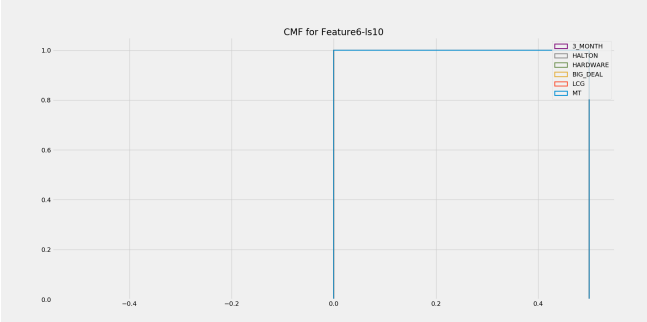


Figure (3) Interpolation for Data 1

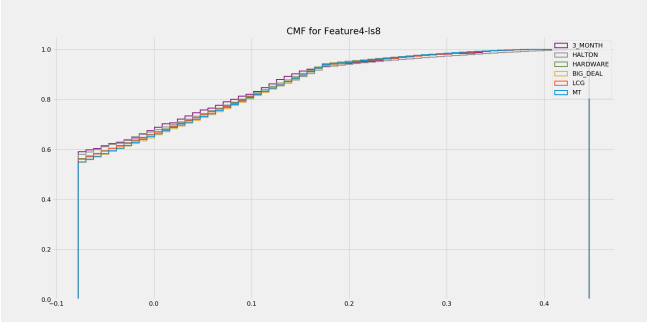


Figure (4) Interpolation for Data 2

grows quicker than other RNGs, and for the upper part, the halton RNG CMF grows slower than other RNGs. Secondly, in both plot(a) and plot(b), the 3-month curve which is purple one has more small jumps than other lines(they almost do not jump). This observation is within expectation since we only have limited real game data, that the increase step size of real game curve will be larger than other RNGs' curve. Although the real game curve has more small jumps, according to the plot(a), the overall shape is almost same as other RNGs' curve except halton RNG's curve. Thirdly, in plot(c), all the CMF jump from 0 to 1 at $x = 0$. This kind of feature such as length of longest suit == 12 is rare in the game. The probability is approximate 10^{-9}. Actually, this is an extreme situation that we do not obverse any occurrence of this feature. For other relative rare feature, we can obverse some occurrences. For example, in plot(d) which is the CMF for feature longest suit == 8, there is a jump for CMF from 0 to 0.6 at $x = -0.0776$. The reason is that many data points do not obverse occurrence of feature 4. $\bar{x}_4 = 0$ in period of m. Therefore

$$
\begin{aligned}
x_4^{'} &= \text{avg}([\bar{x}_{14} - \hat{x}_4, ..., \bar{x}_{m4} - m \times \hat{x}_4]) \\
\text{Here } m &= 32 \\
&= \text{avg}([-0.0047, -0.0096, ..., -0.1504) \\
&= -0.0776
\end{aligned}
$$

Other data points obverse occurrence of feature 4 for different times, so the CMF increases step by step after the beginning jump. If we increase the window size m, we have more probability to view occurrence of feature. The jump at the beginning of CMF will decrease. However since we only have limited number of real data game, we want to make sure there are enough data points to plot smooth real game data curve. The window size m can not be large. The $m = 32$ is a good value after trade-off.

B. Short-term timeseries Statistic Analysis

After generating the distributions, we apply several statistic tests to analysis these data and probability distributions. The statistic data provides more quantitative divergence than the plots.

1) Binary Ratio Test: The each data point is the average deviation from expected value among a sequence of bridge deals. We use sign function to threshold the data points and calculate the ratio of the number of $+1$ over the number of total data points.

$$
\text{sign}(x) = \begin{cases} 1 & \text{for } x \geqslant 0 \\ -1 & \text{for } x < 0 \end{cases}
$$

The ratio should be close to $\frac{1}{2}$. The table IV shows the result of the binary-test. For the most features, the results are close to the $\frac{1}{2}$ for each RNG and real games. For more

	MT	LCGbad	LCGgood	BIGDEAL	Hardware	HALTON	3_MON	1_YEAR	diff1	diff2
ls4	0.494941	0.475594	0.489651	0.493626	0.494519	0.505118	0.469565	0.497561	0.035553	0.001314
ls5	0.497693	0.510736	0.502131	0.496925	0.497920	0.494513	0.506832	0.478049	0.032687	0.000995
ls6	0.484704	0.481938	0.488030	0.486901	0.485608	0.471498	0.501863	0.490244	0.030366	0.002197
ls7	0.458683	0.463377	0.460083	0.460516	0.458118	0.437518	0.460870	0.478049	0.040531	0.002398
ls8	0.353054	0.375487	0.351060	0.354280	0.352125	0.338543	0.332919	0.346341	0.042568	0.002155
ls9	0.046414	0.053707	0.045593	0.046893	0.046379	0.049112	0.050932	0.046341	0.008114	0.000514
ls10	0.002106	0.002441	0.001970	0.002116	0.002188	0.002798	0.002484	0.004878	0.002908	0.000081
ls11	0.000048	0.000000	0.000030	0.000046	0.000051	0.000090	0.000000	0.000000	0.000090	0.000005
ls12	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
ls13	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
ss0	0.461220	0.479984	0.463050	0.463006	0.460992	0.437308	0.458385	0.480488	0.043179	0.002014
ss1	0.492542	0.489264	0.497776	0.493995	0.493301	0.482909	0.505590	0.480488	0.025102	0.001453
ss2	0.500032	0.488772	0.498146	0.497700	0.499673	0.511379	0.478261	0.524390	0.046129	0.002333
ss3	0.479087	0.476566	0.475775	0.479292	0.478504	0.480237	0.453416	0.507317	0.053901	0.000787
6-5	0.439693	0.451164	0.439229	0.441487	0.440787	0.424758	0.432298	0.448780	0.026406	0.001794
6-6	0.085492	0.085453	0.084914	0.086049	0.085481	0.082779	0.091925	0.107317	0.024539	0.000568
7-5	0.121620	0.124997	0.124039	0.122353	0.121581	0.118128	0.141615	0.131707	0.023487	0.000772
7-6	0.007034	0.010254	0.006940	0.007331	0.007065	0.006889	0.006211	0.000000	0.010254	0.000296
p<8	0.499744	0.517095	0.499520	0.498191	0.500432	0.500799	0.501863	0.468293	0.048802	0.002241
p8-13	0.499431	0.483393	0.497233	0.501074	0.498277	0.497677	0.491925	0.502439	0.019046	0.002797
p14-18	0.495242	0.477536	0.496377	0.494212	0.495433	0.497140	0.556522	0.529268	0.078985	0.001221
p19+	0.458771	0.486822	0.456570	0.456766	0.458873	0.457638	0.462112	0.463415	0.030252	0.002107

Table (IV) Ratio of binary output for different features

	MT	LCG-bad	LCG-good	BIG_DEAL	HARDWARE	HALTON	3_MONTH	1_YEAR
MT	0.000000	0.014270	0.004535	0.006230	0.005257	0.019311	0.023328	0.026211
LCG-bad	0.014270	0.000000	0.010375	0.014180	0.014982	0.027093	0.026853	0.031649
LCG-good	0.004535	0.10375	0.000000	0.005895	0.005255	0.019785	0.013528	0.024609
BIG_DEAL	0.006230	0.014180	0.005895	0.000000	0.005345	0.019689	0.023258	0.025709
HARDWARE	0.005257	0.014982	0.005255	0.005345	0.000000	0.018295	0.022318	0.024857
HALTON	0.019311	0.027093	0.019785	0.019689	0.018295	0.000000	0.031933	0.032759
3_MONTH	0.023328	0.026853	0.013528	0.023258	0.022318	0.031933	0.000000	0.035090
1_YEAR	0.026211	0.031649	0.024609	0.025709	0.024857	0.032759	0.035090	0.000000

Table (V) KL distance for each model with respect to other models

detailed observation, the similarity among MT, BigDeal and Hardware are more precise than other RNG. We calculated the difference(diff1 in the table IV) among the values of the all RNGs for each features. And the difference(diff2 in the table IV) for MT, BigDeal and Hardware. The diff2$\leqslant 0.0027$ is much smaller than diff1$\leqslant 0.078$

$$\text{diff1} \gg \text{diff2}$$

From the binary ratio test, we got the conclusion that the MT, hardware and BigDeal generators agree with each other. They might be good generators for bridge game.

2) Kullback–Leibler Divergence Test: The Kullback-Leibler divergence [6] (also called relative entropy) is a measure of how the given probability distribution is different from the reference probability distribution. The following is the equation of Kullback-Leibler divergence.

$$K_{KL}(P||Q) = -\sum_{x\in\chi} P(x)\log\left(\frac{Q(x)}{P(x)}\right)$$

The following table V shows the divergence between two distributions. The bigger the value, the more divergence for these CMF.

The KL distance agrees the observation and zeroth test from previous section. In discrepancy test, we can not see any difference among MT, LCG, big-deal and hardware curves. Their KL distance is approximate 5×10^{-3}. The 3-month curve and 1-year curve have average 2×10^{-2} distance from above four curves. The Halton curve has average 1.36×10^{-2} distance from above four curves and has average 3.23×10^{-2}. From the Figure , the halton curve is above MT, LCG, big-deal and hardware curves and 3-month curve is below four of curves at first stage. And at second the halton curve and 3-month reverse their position. Therefore, the distance between halton curve and 4 curves is smaller than the distance between halton curve and 3-month.

Among the KL distance, timeseries discrepancy test and zero order test, there is one disagreement that the halton RNG passes the zero order test but it does not pass all timesreies discrepancy test.

C. Long-term timeseries Statistic Analysis

For short term timeseries, we only consider each data point as a small period of time. For lone-term timeseries, we will consider the time as a factor among the data points. After including time stamp, we can farther evaluate the time dependency of adjacent deals generated by different random number generators.

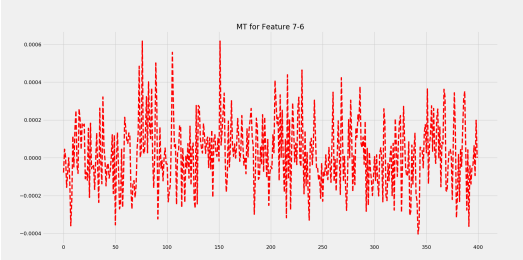


Figure (5) Long-term timeseries for MT

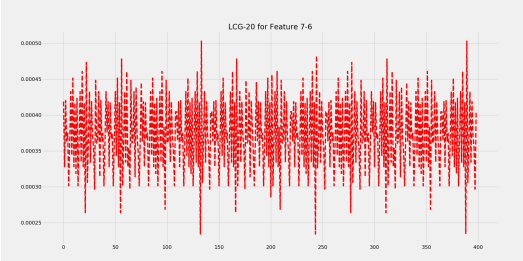


Figure (6) Long-term timeseries for LCG-20

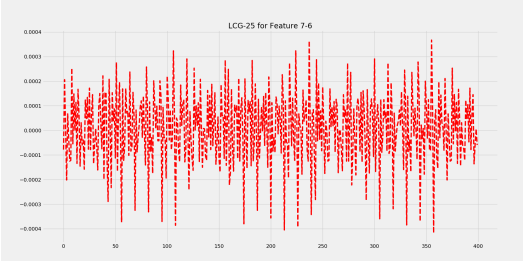


Figure (7) Long-term timeseries for LCG-25

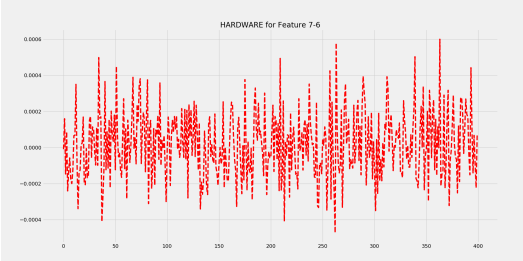


Figure (8) Long-term timeseries for hardware RNG

I generated 64000000 deals of bridge. After the first polling in short-term timeseries test, there remains 2,000,000 time period. It is hard to show 2,000,000 data points, so we did another average polling. Average polling will calculate every contributes from each time period. It contains more information than max polling. The following plots are time series deviations from theoretical value for different RNGs'.

From figure6 and figure7, the LCG20 and LCG25 show obvious periodic repetitions patterns. In long-term, the same set of bridge deals will appear again and again. In real game, we are not expected to see repeated set of deals. So we get the same result that the LCG random number generator family can not be used in bridge deal generation. For other random number generator, they do not have the obvious

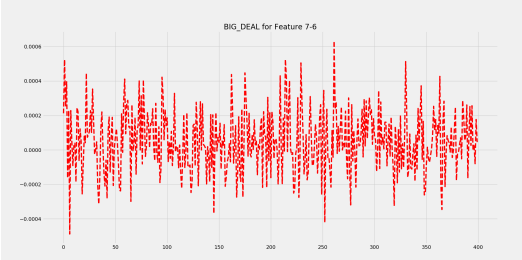


Figure (9) Long-term timeseries for BigDeal

pattern. Although LCG25 preforms good in short run, the long-term test demonstrates the weakness of LCG25 RNG. For other RNGs', we can not observe clear pattern in the graph. For further research, we can increase the bridge deals. That more deals means we have more chance to go through the scope of PRNG. After reaching the scope of PRNG, the PRNG will start a new circulation. The last circulation should be a period of pattern. What's more, we can develop a model to evaluate the randomness of deals respect to time. If we can do some predictions to next few deals, we can conclude the randomness is bad for this RNG. For now, I can conclude the MT, BigDeal, halton PRNG and hardware real RNG are good enough to generate bridge deals for the game.

VI. Conclusion

All of zero test, short-term/long-term timeseries discrepancy test show a good PRNG can provide enough randomness for Bridge deals generator. The Mersenne Twister and Bigdeal has same performance as hardware RNG. If users want to let the game more interesting, they can try using 52-dimension Halton sequence as Bridge deal generator. It is still random but there will be low discrepancy of one feature in a time period of several deals. In other words, the specific feature will distribute more evenly than other Bridge deal generator. The LCG with higher exponent(25) has good performance in short-term discrepancy test but it shows obvious pattern in long-term discrepancy test. Three tests check different features for each PRNG and real RNG.

References

[1] H. van Staveren, "Big deal–a new program for dealing bridge hands." `https://sater.home.xs4all.nl/doc.html`, September 2000. Accessed on 2019-09-17.

[2] T. N. Makoto Matsumoto, "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 1998.

[3] F. X.Wang, "Randomized halton sequences," *Mathematical and Computer Modelling*, 2000. Pages 887-899.

[4] G. F. R.HOFER, P.KRITZER, "Distribution properties of generalized van der corput–halton sequences and their subsequences," *International Journal of Number Theory*, 2009. Pages 719-746.

[5] P. B.Jun, "The intel random number generator," *CRYPTOGRAPHY RESEARCH, INC. WHITE PAPER PREPARED FOR INTEL CORPORATION*, 1999.

[6] J. S.Eguchi, "Interpreting kullback–leibler divergence with the neyman–pearson lemma," *Journal of Multivariate Analysis*, 2006. Pages 2034-2040.