



## Implementing the Classical Poker Approach for Testing Randomness

**Wael M. F. Abdel-Rehim\***  
Department of Mathematics and  
Computer Science, Faculty of Science,  
Suez University, Suez, Egypt

**Ismail A. Ismail**  
Department of Computer Science,  
Faculty of Computer Science and Information  
System, 6 October University, Cairo, Egypt

**Ehab Morsy**  
Department of mathematics,  
Suez Canal University,  
Ismailia 22541, Egypt

---

**Abstract**— *In this paper we discuss the problem of testing randomness motivated by the need to evaluate of the quality of different random number generators which may not generate a true random numbers. Such number generators are used by many practical applications including computer simulations, cryptography, and communications industry, where the quality of the randomness of the generated numbers affects the quality of these applications. In this paper we discuss one of the most popular approaches for testing randomness, Poker test. In particular, two versions of Poker test are presented in the literature, the classical Poker test and the modified Poker test. The modified Poker test (based on Stirling's numbers) has been motivated by the difficulties involved in implementing the classical Poker test at that time. In this paper, we show that the classical Poker test can be implemented with no significant extra running time compared with the modified Poker test.*

**Keywords**— *Poker test, Randomness, Random numbers tests, Cryptography, secret keys*

---

### I. INTRODUCTION

Random numbers are very useful in simulation, game theory, information theory, pattern recognition, probability theory, and statistical. The random numbers are especially helpful in cryptography. Therefore, measuring the quality of randomness of a given sequence is a crucial problem that significantly affects the quality of many practical applications such as distributed algorithms, cryptography (see [1]), statistical sampling, and computer simulation. In other words, the quality of such applications depends on generating unpredictable (random) sequence of quantities. From the practical point of view, such sequence must be of sufficiently large size in the sense that the probability of any particular value being selected must be sufficiently small in order to prevent an adversary from optimizing a search scheme based on such probability.

There are many techniques discussed in the literature for generating random and pseudorandom bits and numbers. A random bit generator is a device or an algorithm which outputs a sequence of independent and unbiased binary digits. A random bit generator can be used to generate uniformly distributed random numbers. However, generating of random bits is an inefficient procedure in most practical environments (storing and transmitting a large number of random bits are impractical if these are required in applications). We can overcome this difficulty by substituting a random bit generator with a pseudorandom bit generator (PRBG);

In order to make sure that such generators are secure enough, they should be subjected to a variety of statistical tests designed to detect the specific characteristics expected of random sequences. We now review a number of empirical tests described in the literatures; for further details see [2, 3, 4, 5].

**Serial test** develops frequency distribution of pairs of samples. Then we compare the actual distribution against this expected distribution, using the chi-square test.

**Runs test** tests the runs up and down or the runs above and below the mean by comparing the actual values to expected values. The statistic for comparison is the chi-square.

**Poker test** (to be explained in details in the next section) treats numbers grouped together as a poker's hand. Then the hands obtained are compared to what is expected using the chi-square test (see [6, 7, 8]).

### POKER TEST

In this section we present in details Poker approach for testing randomness. In particular, we describe the classical Poker test and the Poker's test approximate approach based on Stirling's numbers.

#### Original Poker Test

The classical poker test consists of using all possible categories obtained from poker, i.e., AAAAA (five of a kind), AAAAB (four of a kind), AAABB (full house), AAABC (three of a kind), AABBC (two pairs), AABCD (one pair), and ABCDE (bust). In general, the poker test considers  $n$  groups of five successive integers denoted by  $(X_{5i}, X_{5i+1}, \dots, X_{5i+4})$ ,  $0 \leq i \leq n$ , and observes which of the following seven patterns is matched by each quintuple. The following table 1 summarizes such patterns.

Table 1. Different patterns of the classical Poker test.

Name	Pattern
All different	ABCDE
One Pair	AABCD
Two pairs	AABBC
Three of a kind	AAABC
Full house	AAABB
Four of a kind	AAAAB
Five of a kind	AAAAA

A chi-square test is based on the number of quintuple in each category. We count the number of occurrences in each k-tuples, and then use a chi-square analysis against the theoretical probabilities to determine whether the stack represents a fair poker deck. For the sake of completeness, we compute theoretical probabilities of all such categories. Clearly, the probability of choosing any number =  $\frac{1}{40} \times 4 = \frac{1}{10}$ .

1) The probability of choosing five of a kind =  $(\frac{10}{10} \times \frac{1}{10} \times \frac{1}{10} \times \frac{1}{10} \times \frac{1}{10} \times \frac{5!}{5!0!}) = 0.0001$

2) The probability of choosing four of a kind =  $(\frac{10}{10} \times \frac{1}{10} \times \frac{1}{10} \times \frac{1}{10} \times \frac{9}{10} \times \frac{5!}{4!1!}) = 0.0045$

3) The probability of choosing full house =  $(\frac{10}{10} \times \frac{1}{10} \times \frac{1}{10} \times \frac{9}{10} \times \frac{1}{10} \times \frac{5!}{3!2!}) = 0.009$

4) The probability of choosing three of a kind =  $(\frac{10}{10} \times \frac{1}{10} \times \frac{1}{10} \times \frac{9}{10} \times \frac{8}{10} \times \frac{1}{2} \times \frac{5!}{3!1!1!}) = 0.0720$

5) The probability of choosing two pairs =  $(\frac{10}{10} \times \frac{1}{10} \times \frac{9}{10} \times \frac{1}{10} \times \frac{8}{10} \times \frac{1}{2} \times \frac{5!}{2!2!1!}) = 0.108$

6) The probability of choosing one pair =  $(\frac{10}{10} \times \frac{1}{10} \times \frac{9}{10} \times \frac{8}{10} \times \frac{7}{10} \times \frac{5!}{2!3!}) = 0.5040$

7) The probability of choosing a bust =  $(\frac{10}{10} \times \frac{9}{10} \times \frac{8}{10} \times \frac{7}{10} \times \frac{6}{10} \times \frac{5!}{5!}) = 0.3024$

The following table 2 summarizes such probabilities.

Table 2. Different patterns of the classical Poker test and their probabilities

Pattern	Probability
AAAAA	0.001
AAAAB	0.0045
AAABB	0.009
AAABC	0.072
AABBC	0.108
AABCD	0.504
ABCDE	0.3024

It is well known that Poker test applied to certain applications such as simulation [9] and cryptography [10, 1] in which we need to generate random integers or a random sequence of bits. For example, in cryptography, secret keys (used for encryption of messages or other purposes) are generated using Random Number Generators (RNGs) [10].

### Approximate Poker Test

At the time the classical Poker test is designed, checking the occurrences of these subsequences of length five using a computer program creates difficulties for the programmers as they have no one systematic similarity. This motivates constructing a simpler version of the classical test to overcome the programming difficulties involved.

A good compromise would simply be to count the number of distinct values in the set of five (see [4, 9]). Namely, we get five categories, 1 different, 2 different, 3 different, 4 different and 5 different. Thus, a finite time algorithms have been designed to implement such modified Poker test; see [3, 11].

5 different	all different
4 different	one pair
3 different	two pairs, or three of a kind
2 different	full house, or four of a kind
1 different	five of a kind

This breakdown is easier to determine systematically, and the test is nearly as good. In general, we consider n groups of k successive numbers, and then count the number of k-tuples with r different values. A chi-square test is then made using the following probability of the existence of r different.

$$Pr = \frac{d(d-1)\dots(d-r+1)}{d^k} \left\{ \begin{matrix} k \\ r \end{matrix} \right\} \quad (1)$$

Where  $\left\{ \begin{matrix} k \\ r \end{matrix} \right\}$  denote the Stirling number of the second kind (the number of ways to partition a set of k elements into exactly r parts). The Stirling number can be computed using a well known formula. The following table summarizes the values for the Stirling numbers for k=5 and r=1, 2, 3, 4, 5.

r	1	2	3	4	5
$\left\{ \begin{matrix} 5 \\ r \end{matrix} \right\}$	1	15	25	10	1

It is now possible to make a table with number of special quintuples and the measured number. To calculate the expected values we use equation (1) with d= 10. Now, we determine theoretical probabilities of such categories.

$$Pr(1 \text{ different}) = \frac{10}{10^5} \left\{ \begin{matrix} 5 \\ 1 \end{matrix} \right\} = 0.0001$$

$$Pr(2 \text{ different}) = \frac{10(10-1)}{10^5} \left\{ \begin{matrix} 5 \\ 2 \end{matrix} \right\} = 0.135$$

$$Pr(3 \text{ different}) = \frac{10(10-1)(10-2)}{10^5} \left\{ \begin{matrix} 5 \\ 3 \end{matrix} \right\} = 0.180$$

$$Pr(4 \text{ different}) = \frac{10(10-1)(10-2)(10-3)}{10^5} \left\{ \begin{matrix} 5 \\ 4 \end{matrix} \right\} = 0.504$$

$$Pr(5 \text{ different}) = \frac{10(10-1)(10-2)(10-3)(10-4)}{10^5} \left\{ \begin{matrix} 5 \\ 5 \end{matrix} \right\} = 0.3024$$

Then different hands obtained can be compared to what is expected using the chi-square test to see how far the data has strayed from the theoretical distribution.

## II. EXPERIMENTAL RESULTS

In this section we implement the classical Poker test and the corresponding modified version. We evaluate both versions of the test by implementing programs using C++ code that create random numbers and count the occurrence of these differences or count number of occurrences, then classified each to possible type of poker hand. Finally, it determines the chi-square. We have compared the two algorithms by comparing their running time. The experimental results are reported on PC 2.4 GHz, 1024 MB of RAM, 256 KB of cache.

**Example 1:** We implement the classical Poker test on the ten million digits (2,000,000 Poker hands). The degrees of freedom (df) for chi-square table equals 6.

Table 3. Summary of Chi-Square Analysis for poker Test

Cell/Poker Hand	Observed number of hands (O)	Expected number of hands (E)	(O - E) <sup>2</sup> / E
Busts (All different)	604219	604800	0.5581
One pair	1008237	1008000	0.0557
Two pair	216514	216000	1.2231
Three of a kind	143808	144000	0.2560
Four of a kind	9087	9000	0.8410
Five of a kind	193	200	0.2450
Full house	17942	18000	0.1869
Sums	2000000	2000000	X <sup>2</sup> = 3.37

For df =6, we get X<sup>2</sup>.<sub>.05</sub> = 12.6. Since the obtained value X<sup>2</sup> = 3.37 is less than X<sup>2</sup>.<sub>.05</sub> = 12.6, the null hypothesis is retained. This implies that the underlying data is truly random.

**Example 2:** Now we implement the modified version of the Poker test on the ten million digits (2,000,000 Poker hands)

Table 4. Summary of Chi-Square Analysis for modified approach (Stirling)

Cell/Poker Hand	Observed number of hands (O)	Expected number of hands (E)	(O - E) <sup>2</sup> / E
1 different	193	200	0.2450
2 different	27029	27000	0.0311
3 different	360322	360000	0.2880
4 different	1008237	1008000	0.0557
5 different	604219	604800	0.5581
Sums	2000000	2000000	X <sup>2</sup> = 1.18

For  $df = 4$ , we get  $X^2_{.05} = 9.49$ . Since the obtained value  $X^2 = 1.18$  is less than  $X^2_{.05} = 9.49$ , the null hypothesis is retained. Thus, the data is consistent with the series being random.

Now, we analyze Chi-square for both the classical and the modified Poker test approaches described in Figures 2 and 3. We apply the two methods to ensemble of different size and checked the results to see if they are within a specified confidence level. The results are shown in the following table 5.

Table 5. Summary of Chi-Square Analysis for the original Poker's test and the modified approach

Random No.	No of hands	Original Poker's test Chi-square value	Modified approach Chi-square value
1000	200	7.93	3.28
5000	1000	7.1	6.85
10000	2000	3.42	2.7
50000	10000	3.63	2.85
100000	20000	3.51	3.28
500000	100000	7.64	2.78
1000000	200000	2.63	1.79
5000000	1000000	1.65	1.41
10000000	2000000	3.37	1.18

The degrees of freedom (df) is 6 in the classical poker test and 4 in the modified approach. Our chi-squared values is less than the critical value for the 0.05 significance level (12.9 to be precise in the classical Poker test and 9.49 in the modified approach), we accept the null hypothesis as true and conclude that the two methods seemed to produce acceptable chi-square statistics. The chi-squares were within the 95% confidence interval.

Finally, we analyze the running time the classical and the modified Poker test approaches described in Figures 2 and 3. We determine the running time of executing both algorithms (time is in milliseconds). The resulting running time is shown in the following table 6.

Table 6. Summary of execution time for the original Poker's test and modified approach

Number of Random Numbers	The classical Poker test	The modified Poker test
1000	47	32
5000	78	31
10000	93	78
50000	172	141
100000	219	203
500000	359	343
1000000	1375	1296
5000000	3219	2984
10000000	6047	5906

The results of table 6 (shown in Figure 1) imply the there is no a significant difference in term of the running time between the classical and the modified Poker test.

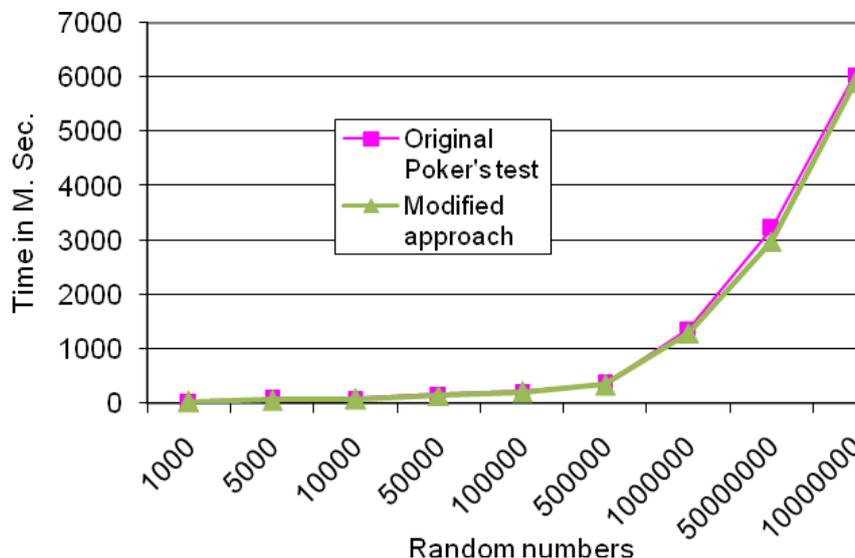


Figure 1. Performance comparison of the classical the modified Poker test in term of the implementing time of both approaches.

1. Read number of hands to deal
2. Open File to read random numbers
3. do
4. Loads the hands into an array // a sequence of 5 numbers
5. Determine which kind of combination this group of 5 contains
6. Count the number of similar values; breaks at 5
7. Increment the appropriate counter //(7 counters: all different, one pair, two pairs, three of a kind, full house, four of a kind and five of a kind)
8. While (loads the hands < number of hands)
9. Calculate the percentage of the n total repetitions corresponding to each counter
10. Computes the expected theoretical values
11. Compute chi square using the expected probabilities
12. Measure execution time in the program
13. Print "The program execution time"
14. Print "Chi square"

Figure 2. The classical Poker test pseudo-code algorithm

1. Read number of hands to deal
2. Open File to read random numbers
3. do
4. Loads the hands into an array // a sequence of 5 numbers
5. Determine which kind of combination this group of 5 contains
6. Count the number of distinct values; breaks at 5
7. Increment the appropriate counter //(4 counters: 5 different, 4 different, 3 different and 2 different 1 different)
8. While (loads the hands < number of hands)
9. Calculate the percentage of the n total repetitions corresponding to each counter
10. Computes the expected values using Stirling numbers
11. Compute chi square using the expected probabilities
12. Measure execution time in the program
13. Print "The program execution time"
14. Print "Chi square"

Figure 3. The modified Poker test pseudo-code algorithm

### III. CONCLUSIONS

We have been studied Poker test, one of the most popular approaches for testing randomness. In particular, we have been compared the performance of implementing the well known two versions of Poker test, the classical and the modified Poker test, where the modified Poker test has been

motivated by the difficulties involved in implementing the classical Poker test at the time the classical version is designed. We have been shown that the classical Poker test can be implemented with no significant extra running time compared with the modified Poker test. These results encourage the using of the classical test over the modified one.

**REFERENCES**

- [1] Menezes, A. J., Oorschot, P. C. V., Vanstone, S. A., 1997. Handbook of applied cryptography. 1st Edn., CRC Press, Boca Raton, ISBN-10: 0849385237.
- [2] Kendall, M G, Smith, B. B., 1938. Randomness and random sampling numbers. Journal of the Royal Statistical Society 101: 147–166.
- [3] Hamilton, J. A., Nash, D. A. and Pooch, U.W., 1997. Distributed Simulation. 1st Edn., CRC Press, Boca Raton, ISBN-10: 0849325900.
- [4] Knuth, D. E., 1997. The Art of Computer Programming: Seminumerical Algorithms. Vol. 2 (3rd Edn.), Addison-Wesley, Mass, ISBN-10: 0201896834.
- [5] Sheskin, D. J., 1997. Handbook of Parametric and Nonparametric Statistical Procedures. 2nd Edn., CRC Press, Boca Raton, ISBN-10: 0849331196.
- [6] Talamba, S., 2001. A Theoretical and Empirical Study of Uniform Pseudo-Random Number Generators. Senior Thesis, Department of Computer Science, Middlebury College.
- [7] Rutti, M., 2004. A Random Number Generator Test Suite for the C++ Standard. Diploma Thesis, Institute for Theoretical Physics, ETH Zurich.
- [8] Stewart, W. J., 2009. Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling. 1st Edn., Princeton University Press, Princeton, ISBN-10: 0691140626.
- [9] Karian, Z. A., Dudewicz, E. J., 1999. Modern statistical systems and GPSS simulation. 2nd Edn., CRC Press, Boca Raton, ISBN-10: 0849339227.
- [10] Brands, S. and Gill, R., 1995. Cryptography, statistics and pseudo-randomness. I. Probability Mathematical Statistics, 15: 101-114.
- [11] Karl, A., 2008. Pseudorandom Numbers: Generation, Statistical Measures, Monte Carlo Methods, and Implementation in C++. Senior Thesis, Department of Mathematics, University of Notre Dame.