Studying the relationship between linguistic variables and the
degrees of primitive polynomials used in pseudo-random
number generator based on fuzzy logic

I V Anikin¹ and K Alnajjar¹

¹Kazan National Research Technical University, Kazan, Russia, 420111
e-mail: anikinigor777@mail.ru, khkazan@mail.ru

Abstract. In this paper we study the relation between linguistic variables and the degree of characteristic primitive polynomial of used LFSR in previously suggested fuzzy pseudorandom number generator FRNG, keeping the output pseudorandom series secure. This means that we should take into account two important properties: the randomness of the output and the security against correlation attacks. The first property mainly depends on selection of primitive polynomials used in constructing FRNG. The second property can be realized by making the output series balanced in sense that the probability of appearing the bits of each used LFSR approximately equal to others in the output stream.

1. Introduction

As described in [1] the structure of the suggested FRNG consists of number of LFSRs (to simplify the study we will use only two). The outputs of LFSRs go through 32 bits sized buffers where an estimation of two fuzzy linguistic variables is done - the first involves in evaluating number of ones \(f_0\) in the buffer, and the second \(|f_1-f_2|\) in estimating the difference between number of blocks (0110) of two consecutive ones \(f_1\) and the number of gaps (1001) consist of two consecutive zeros \(f_2\) in the considered buffer for every bit. Then a group of fuzzy If-Then rules plays main role in deciding which one of the used LFSRs is best at every moment and pass it's out bit to be a bit of the pseudorandom series generated by FRNG then a new estimation of the linguistic variables associated with every buffer begins (after shifting the continent of the buffers one bit to the right and inserting a new bit from the related LFSR) to select the next bit and pass it to the output of the system and so on figure 1 illustrates the general structure of the proposed FRNG.

It's very important here to describe the linguistic variables \(f_0, |f_1-f_2|\) in details, because they are the main keys of this study. As seen in ‘figure 1’ these two linguistic variables play essential role with the If-Then rules in deciding which LFSR's bit will be passed to the output of the FRNG depending on the results of comparing the output of combining these two fuzzy variables [4] for each LFSR' buffer and then selecting the best one and passing it's LSB to the output of the generator at every bit repeatedly.

As a result of studying the parameters of FRNG in [2], we found that every linguistic variable has three membership functions (MFs); (Low, Medium, High) for the first variable \(f_0\) and (Excellent, Good, Bad) for the second \(|f_1-f_2|\).

Then a group of fuzzy rules shown in ‘table 1’ helps in estimating the statistical situation of the considered buffer at the moment through combining the two variables, then in the last step we
compare the obtained results for all used LFSRs and select the best one of them and pass it's bit to be
the output of the FRNG, then the process continues in this way repeatedly for every bit.

![General structure of proposed FRNG.](image)

**Figure 1.** General structure of proposed FRNG.

| $f_0$       | $|f_1-f_2|$ | Fuzzy out |
|------------|-------------|-----------|
| Low        | Excellent   | Bad       |
| Low        | Good        | Bad       |
| Low        | Bad         | Bad       |
| Medium     | Excellent   | Best      |
| Medium     | Good        | Good      |
| Medium     | Bad         | Bad       |
| High       | Excellent   | Bad       |
| High       | Good        | Bad       |
| High       | Bad         | Bad       |

Table 1. Fuzzy If-Then rules of FRNG.

Settings of the fuzzy groups for every MF sufficiently affect the output of the pseudorandom
generator. In [2] we studied their influence on the security of the proposed generator FRNG against
correlation attacks [5], we concluded that the output of FRNG should be balanced to have a secure
generator.

Balancing the output series could be achieved via tuning the MFs of the fuzzy linguistic variables $f_0$
and $|f_1-f_2|$ for each LFSR separately, and here lies the main idea of this paper.

2. Tuning the membership functions of the fuzzy linguistic variables $f_0$ and $|f_1-f_2|$
Tuning the MFs of the linguistic variables $f_0$, $|f_1-f_2|$ will directly affect the probability of appearing
the bits of the related LFSR in the generated pseudorandom series. So any tuning or change in the
configurations of the MFs will increase or decrease the probability of appearing the related LFSR’s
bits in the output stream of the FRNG. At the same time these changes shouldn’t disturb the balance of
the FRNG.

The statistical property of the output of LFSR is mainly depend on the selected characteristic
polynomial [6]. In [2] we decided to use a special type of primitive polynomials to construct our
generator, and we found that this type of polynomials sufficiently increases the efficiency of FRNG
and has many practical advantages. The following formula briefly describes this type of polynomials:

$$f(x) = (1 + x^{b_1})(1 + x^{b_2})...(1 + x^{b_m}) + x^n$$

(1)

Parameters of these type of polynomials $(b_1, b_2, ..., b_m, n)$ should satisfy the following conditions (2), in
addition to satisfying the primitivity tests:
For simplicity we will use a simple version of FRNG that contains only two LFSRs, and we will investigate the relationship between the degrees of selected primitive characteristic polynomials and the associated MFs of the fuzzy linguistic variables \( f_0 \) and \( |f_1 - f_2| \) for each of them, regarding the balance of output series of FRNG. The following two characteristic polynomials used in constructing the FRNG:

\[
P_1(x) = (1 + x)(1 + x^2)(1 + x^3)(1 + x^4)(1 + x^5)(1 + x^6) + x^{39} + x^{89}
\]
\[
= 1 + x + x^2 + x^3 + x^4 + x^7 + x^{11} + x^{15} + x^{16} + x^{17} + x^{18} + x^{39} + x^{40}
\]
\[
+ x^{44} + x^{45} + x^{49} + x^{50} + x^{54} + x^{55} + x^{56} + x^{57} + x^{61} + x^{66} + x^{67} + x^{71} + x^{72} + x^{89}
\]
\[
P_2(x) = (1 + x)(1 + x^3)(1 + x^4)(1 + x^{20})(1 + x^{53}) + x^{97}
\]
\[
= 1 + x + x^2 + x^3 + x^4 + x^7 + x^{11} + x^{12} + x^{20} + x^{21}
\]
\[
+ x^{24} + x^{25} + x^{27} + x^{28} + x^{31} + x^{32} + x^{33} + x^{34} + x^{35} + x^{36} + x^{37} + x^{74} + x^{77} + x^{78} + x^{80} + x^{81} + x^{84} + x^{85} + x^{97}
\]

So at the beginning we will briefly explain how to calculate the probability of appearing one of LFSR’s bits in the output of the FRNG. Then we will discuss the process of tuning the MFs regarding the obtained value of P. Then via number of examples we will study how these configurations related to the degrees of used characteristic primitive polynomials of LFSRs.

2.1. Calculating the probability value P

For simplicity only two LFSRs used in constructing the FRNG so we will calculate the probability value \( P(\text{out}_{\text{sys}} = \text{out}_{\text{LFSR1}}) \), which denotes the probability that the output of generator is equal to the output of the LFSR1, and the second value \( P(\text{out}_{\text{sys}} = \text{out}_{\text{LFSR2}}) \), which denotes the probability that the output of generator is equal to the output of the LFSR2. These values should be as close as possible to 0.5 to say that the output stream of FRNG is uncorrelated or balanced. We should repeatedly calculate this values at every modifications (iteration) in the MFs settings of the linguistic variables until the desired values is reached. To calculate the value of probability \( P(\text{out}_{\text{sys}} = \text{out}_{\text{LFSR}}) \) after tuning we should count how many bits of the output series is equal to the output of LFSRx, then divide the resulting value on the total number of generated bits (the length of the series in our case 1024000).

Finally, it’s very important to mention that the tuning process should be done also when setting a new key values to FRNG (when changing the seeds of used LFSRs). So the resulting configurations of the membership functions of linguistic variables \( (f_0, |f_1 - f_2|) \) of every LFSR will definitely depend on the initiate state of used LFSRs (the seeds).

2.2. Tuning the MFs regarding the calculated value P

Firstly we initiate the membership functions of linguistic variables \( (f_0, |f_1 - f_2|) \) of every LFSR then generate a 1024000 bits then calculate the first value of probability P, and here starts the tuning process. The calculated value of P will often be far from 0.5 at the beginning, so the constructed FRNG in this case is not balanced, and here lies the necessity of tuning process to make the generator secure against the correlation attacks.
As previously described every linguistic variable has three MFs (Low, Medium, High) for the first variable \(f_0\) and (Excellent, Good, Bad) for the second \(|f_1-f_2|\). The tuning process starts from the following initial configurations ‘figure 2’ for both of used LFSRs:

- The first fuzzy linguistic variable \(f_0\) has three MFs they mapped as following:
  1. when \(f_0\) belongs to \(\{0,...,8\}\) is mapped to \{Low\},
  2. when \(f_0\) belongs to \(\{9,...,24\}\) is mapped to \{Medium\},
  3. and when \(f_0\) belongs to \(\{25,...,32\}\) is mapped to \{High\}.

- The second fuzzy linguistic variable \(f_1\) has also three MFs they mapped as following:
  1. when \(|f_1-f_2|\) belongs to \(\{0,1,2\}\) is mapped to \{Excellent\},
  2. when \(|f_1-f_2|\) belongs to \(\{3,...,6\}\) is mapped to \{Good\},
  3. and when \(|f_1-f_2|\) belongs to \(\{7,...,10\}\) is mapped to \{Bad\}.

![Figure 2. Initial configurations of linguistic variables (\(f_0, |f_1-f_2|\)).](image)

Taking Table 1 into account we see that the most important MF of the first linguistic variable \(f_0\) is (Medium) and when it's width is greater (increasing number of elements in its fuzzy group), then the probability of appearing of the related LFSR in the output sequence will sufficiently increase and vice versa. Also the same thing for the second linguistic variables \(|f_1-f_2|\) but with the first MF (Excellent) which is the most important one for the second variable. It's worth mentioning that the other MFs of both linguistic variables affect the calculated value of probability but their effect is not so sufficient and tuning them will be very useful when resulting value of probability \(P\) is not too far from 0.5.

So if we want to increase the probability of appearing the output bits of the LFSR1 in the output of the generator we should increase the width of (Medium) MF of the first linguistic variable \(f_0\) or increase the width of (Excellent) MF of the second variable \(|f_1-f_2|\) that are associated with LFSR1 or increase them both, or we have another option in tuning them by decreasing these parameters for LFSR2 to make it's probability of appearing less. Then we should evaluate the balance of the output sequence by computing the new values of probability, then according to this value the tuning process will continue repeatedly until the desired values are reached \(P=\text{P(out}_{sys} = \text{out}_{LFSR1})\geq0.5\) and \(P(\text{out}_{sys} = \text{out}_{LFSR2})\geq0.5\) within the acceptance value of difference \(\epsilon \geq |P-0.5|\). ‘Figure 3’ illustrates the iterative process of tuning the MFs of the linguistic variables \((f_0, |f_1-f_2|)\).

Finally, it's very important to mention that the tuning process should be done also when setting a new key values to FRNG (when changing the seeds of used LFSRs). So the resulting configurations of the membership functions of linguistic variables \((f_0, |f_1-f_2|)\) of every LFSR will definitely depend on the initiate state of used LFSRs (the seeds).
3. Studying the relationship between the configurations of linguistic variables and the degrees of used primitive polynomials of LFSRs

As mentioned previously, the primitive characteristic polynomials were deeply investigated and accurately selected to have a defined type by (1) that makes them fit all the requirements of high performance and randomness. The selected primitive polynomials has excellent statistical properties by virtue of high Hamming weights [7]. So here we will pay more attention on their degrees and how they related to the associated linguistic variables \( (f_0, |f_1 - f_2|) \). To study the relationship between the linguistic variables and the degrees of used primitive characteristic polynomials of LFSRs, we have made a number of numerical experiments using MATLAB environment (version 7.14.0.739 (R2012a)) [8,9], in each of them we used different primitive polynomials of type (1) with different degrees then we have tuned their MFs using the previously described process ‘figure 3’ to reach the balanced version of the constructed FRNG, with the acceptance value of difference \( \varepsilon = 0.05 \). Then we recorded the obtained results in table 2.

![Figure 3. Tuning process of MFs of linguistic variables related to LFSRx.](image)

As seen in the table 2, there is a strong relation between the degrees of used primitive polynomials and the configurations of membership functions (setting of the fuzzy groups of the linguistic variables \( f_0, |f_1 - f_2| \)). There is an inversely proportional relationship between the degree of the polynomial and the width of Medium and Excellent membership functions of the linguistic variables \( f_0, |f_1 - f_2| \) respectively.

It’s clear from table 2, as selecting a polynomial with a big degree will lead to make the width of Medium and/or Excellent membership functions of the associated linguistic variables \( f_0, |f_1 - f_2| \) more narrow relatively depending on the resulting value of the calculated probability \( P \) in order to have a balanced output sequence of FRNG.

Also from the executed experiments we can conclude that this mutual relationship mainly depends on the difference between the degrees of used polynomials. So using a polynomials with a big differ in degrees will lead to a big differ in the setting of membership functions of the linguistic variables (one of them will have a narrow width in settings of MFs and the other will have a wide width). This appears very clear in the last experiment (№ 5 in table 2), when the degrees were chosen to be \( (n_1=61) \) considering the first linguistic variable \( f_0 \), for the first polynomial where the width of “Medium”
The IV International Conference on Information Technology and Nanotechnology  
IOP Conf. Series: Journal of Physics: Conf. Series 1096 (2019) 012178  
doi:10.1088/1742-6596/1096/1/012178

membership function is relatively big (it contains more elements belong to its fuzzy group from 9 till 23) and \((n_2=147)\) for the second polynomial where the width of “Medium” membership function is relatively small (it contains less elements belong to its fuzzy group from 15 till 17); and the same thing for the second linguistic variable \(|f_1-f_2|\) where we consider “Excellent” membership function as the most important one.

Table 2. Experimental results of tuning process.

| Exp. № | Primitive polynomials used in constructing the FRNG | Setting of MFs of \((f_o)\) when FRNG is balanced | Setting of MFs of \(|f_1-f_2|\) when FRNG is balanced | \(P\) |
|--------|--------------------------------------------------|-----------------------------------------------|--------------------------------------------------|-----|
| 1      | \(P_1 = (1+x)(1+x')(1+x')(1+x')(1+x') + x^n\)     | Low: \{0,...,12\}                            | Excellent: \{0,1\}                               | 0.50 |
|        |                                                  | Medium: \{13,...,19\}                        | Good: \{2,...,6\}                                |     |
|        |                                                  | High: \{20,...,32\}                          | Bad: \{7,...,10\}                               |     |
| 2      | \(P_2 = (1+x)(1+x')(1+x')(1+x')(1+x') + x^n\)     | Low: \{0,...,14\}                            | Excellent: \{0,1,2\}                             | 0.46 |
|        |                                                  | Medium: \{15,...,17\}                        | Good: \{3,...,6\}                               |     |
|        |                                                  | High: \{18,...,32\}                          | Bad: \{7,...,10\}                               |     |
| 3      | \(P_3 = (1+x)(1+x')(1+x')(1+x')(1+x') + x^n\)     | Low: \{0,...,7\}                             | Excellent: \{0,...,2\}                          | 0.52 |
|        |                                                  | Medium: \{8,...,22\}                        | Good: \{3,4\}                                   |     |
|        |                                                  | High: \{23,...,32\}                          | Bad: \{5,...,10\}                               |     |
| 4      | \(P_4 = (1+x)(1+x')(1+x')(1+x')(1+x') + x^n\)     | Low: \{0,...,8\}                             | Excellent: \{0,...,3\}                          | 0.49 |
|        |                                                  | Medium: \{9,...,23\}                        | Good: \{4,5\}                                   |     |
|        |                                                  | High: \{24,...,32\}                          | Bad: \{6,...,10\}                               |     |
| 5      | \(P_5 = (1+x)(1+x')(1+x')(1+x')(1+x') + x^n\)     | Low: \{0,...,8\}                             | Excellent: \{0,...,3\}                          | 0.51 |
|        |                                                  | Medium: \{9,...,23\}                        | Good: \{4,5\}                                   |     |
|        |                                                  | High: \{24,...,32\}                          | Bad: \{6,...,10\}                               |     |

So we can conclude that using the fuzzy logic techniques in constructing RNGs added a very useful and significant effect that enables us to adapt the structure of the suggested generator by tuning the MFs of every used LFSR regarding the balance of the output sequence to get a secure version against the correlation attacks. At the same time with using such type of primitive characteristic polynomials defined by (1) and the associated conditions (2) we could sufficiently increase the performance of the FRNG as showed in [3].
4. Conclusion

According to the obtained results of this study we can conclude that there is a very strong and mutual relationship between the configurations of the MFs that are related to the fuzzy linguistic variables and the degree of the characteristic primitive polynomial of the associated LFSR. And applying the tuning process is very important and it sufficiently increases the security of the proposed pseudo-random number generator based on fuzzy logic against the correlation attacks. As a result the generated sequences by the balanced version of FRNG will have low correlation magnitude, high linear complexity, less power consumption, and they will have very good statistical properties that make the resulting FRNG fits the requirements of most telecommunication applications such as cryptography, authentication, etc.

As shown in this paper that using the fuzzy logic in building the non-linear combining function of the outputs of a number of LFSRs made the suggested generator more flexible in order to get a secure pseudo-random number generator against the correlation attacks, in addition to the very good statistical properties that were guaranteed by using the suggested type of primitive polynomials defined by (1) and (2) that fit all the requirements of high performance due to the high diffusion capacity and low number of XOR gates as described in [3] that are needed to implement it hardwarily. It’s worth mentioning that the resulting FRNG after accomplishing the tuning process was tested using NIST statistical tests suite [10], which contains 15 statistical tests intended to detect predefined patterns of defects in the tested binary sequences. The resulting FRNG was successfully passed all 15 tests of packet tests NIST.

In the future, we will try to use greater number of LFSRs in constructing the generator (for example 4 or 8) in order to increase the period of the suggested FRNG to get a pseudo-random number series that are very close to true random series. Also in order to increase the speed of the suggested generator we will try in future to use the distributed computing techniques especially with the expanded versions of FRNG (4, 8 LFSRs).

5. References

[1] Anikin I V and Alnajjar K 2015 Fuzzy stream cipher system Proc. Int. Siberian Conf. on Control and Communications (Omsk)
[2] Anikin I V and Alnajjar K 2016 Pseudo-random number generator based on fuzzy logic Proc. Int. Siberian Conf. on Control and Communications (Moscow)
[3] Anikin I V and Alnajjar K 2018 Primitive polynomials selection method for pseudo-random number generator Journal of Physics: Conf. Series 944 012003
[4] Zadeh L A 1996 Fuzzy Sets, Fuzzy Logic, Fuzzy Systems (World Scientific Press)
[5] Siegenthaler T 1984 Correlation-Immunity of Nonlinear Combining Functions for Cryptographic Applications IEEE Transactions on Information Theory 30(5) 776-780
[6] Klapper A and Xu J 1999 Algebraic Feedback Shift Registers Theoretical Computer Science 226 61-93
[7] Menezes A J van Oorschot P C and Vanstone S A 1997 Handbook of Applied Cryptography (CRC Press, Boca Raton) p 810
[8] MATLAB and Statistics Toolbox Release 2012 MathWorks (Natick, Massachusetts, United States)
[9] MATLAB Documentation MathWorks (14.08.2013)
[10] NIST SP 800-22 Revision 1a 2010 A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications p 131

Acknowledgments

Authors would like to show their gratitude to professors: Kuznetsov, V.M., Pesoshin, V.A. and Zakharov V.M. from Kazan National Research Technical University named after A.N. Tupolev for their useful and valuable notes about the paper.