Novel Pseudorandom Number Generation Using Variant Logic Framework

Jeffrey Zheng

Abstract Cybersecurity requires cryptology for the basic protection. Among different ECRYPT technologies, stream cipher plays a central role in advanced network security applications; in addition, pseudorandom number generators are placed in the core position of the mechanism. In this chapter, a novel method of pseudorandom number generation is proposed to take advantage of the large functional space described using variant logic, a new framework for binary logic. Using permutation and complementary operations on classical truth table to form relevant variant table, numbers can be selected from table entries having pseudorandom properties. A simple generation mechanism is described and shown, and pseudorandom sequences are analyzed for their cycle property and complexity. Applying this novel method, it can play a useful role in future applications for higher performance of cybersecurity environments.

Keywords Pseudorandom number generation · Variant logic · Cryptology

1 Introduction

In advanced cyber environment, cybersecurity mechanism plays a guider role to protect the secure information communicated and stored in network facilities [1, 2]. To achieve adequate network security effects, cryptology has to be placed in the essential position [1]. Different from block ciphers that operate with a fixed transformation on a large block of plaintext, stream ciphers operate with a time-varying transformation on individual plaintext digits. Under the stream cipher methodology, Pseudorandom Number Generator (PRNG) is placed in the central part of the mechanism.

This work was supported by the Key Project on Electric Information and Next Generation IT Technology of Yunnan (2018ZI002), NSF of China (61362014), Yunnan Advanced Overseas Scholar Project.

J. Zheng (✉)
Key Laboratory of Software Engineering of Yunnan, Yunnan University, Kunming, China
e-mail: conjugatelogic@yahoo.com

J. Zheng (ed.), *Variant Construction from Theoretical Foundation to Applications*,
https://doi.org/10.1007/978-981-13-2282-2_18

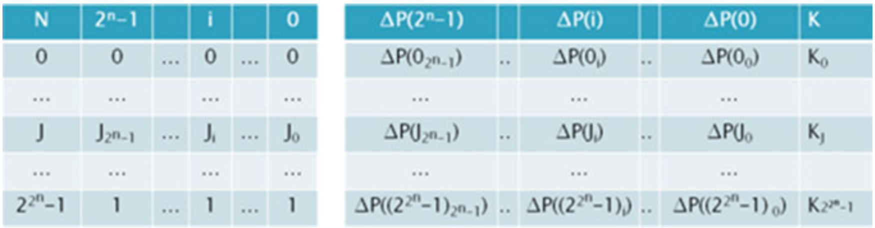
From 2000 to 2003, New European Schemes for Signatures, Integrity, and Encryption (NESSIE) were started [3]. During 2004–2008, another European stream cipher project: eSTREAM selected four software and three hardware schemes for ECRYPT stream ciphers [4]. Such extensive international activities on ECRYPT methodologies are showing the ultra-importance of stream cipher technologies in cyber environments for wider security applications.

From a cyber resilience viewpoint [5–7], a set of researchers focus attention on leakage-resilient pseudorandom generator. This direction has shown interesting results to protect valuable information from side-channel attack aspects.

Since PRNG plays a key role in stream cipher applications and is the heart of cryptology [1, 8–10]. Many mathematical methodologies are applied to this field such as linear automata, cellular automata, Galois fields, and other algebraic constructions [1, 9, 11–14]. In cryptology, Boolean logic operations are essential to create highly effective cryptology systems [1, 9, 15, 16] as binary logic generates the greatest efficiency through manipulation of only 1's and 0's. Therefore, it is advantageous to investigate potential mechanisms in binary logic due to the follow-on effect it has in cryptology.

2 Classical Logic Function Table

A classic logic function in n variables can be represented as a truth table [8, 9]. For a classic sequence in an ordinary number sequence, each table contains 2^n columns and 2^{2^n} rows with a total of $2^n \cdot 2^{2^n}$ bits, respectively. An example of the standard truth table can be seen in Fig. 1a.



N	2^n-1		i		0
0	0	...	0	...	0
...	...		...		...
J	J_{2^n-1}	...	J_i	...	J_0
...	...		...		...
$2^{2^n}-1$	1	...	1	...	1

(a) Truth Table Example

$\Delta P(2^n-1)$		$\Delta P(i)$		$\Delta P(0)$	K
$\Delta P(0_{2^n-1})$	..	$\Delta P(0_i)$	..	$\Delta P(0_0)$	K_0
...		...		...	
$\Delta P(J_{2^n-1})$	..	$\Delta P(J_i)$	..	$\Delta P(J_0$	K_J
...		...		...	
$\Delta P((2^{2^n}-1)_{2^n-1})$	..	$\Delta P((2^{2^n}-1)_i)$	..	$\Delta P((2^{2^n}-1)_0)$	$K_{2^{2^n}-1}$

(b) Variant Table Example

Fig. 1 *n* variable truth table and variant table under *P* and Δ operators

3 Variant Logic Function Table

Variant logic construction is a new proposed theoretical structure [17, 18] to extend classical logic from the three basic operators: $\{\cap, \cup, \neg\}$. Two additional vector operators: permutation P and complementary Δ are included with the original three to form the five basic operators within the novel framework. Let $S(N)$ denote a permutation group with N elements, then $S(N)$ contains a total of $N!$ permutation operators. Let $B_2^N = \{0, 1\}^N$ denote a binary group with N elements, then B_2^N contains a total of 2^N complementary operators.

The permutation (P) and complementary (Δ) operators are two vector operators performed on each column vector of 2^{2^n} bits. For a given P and Δ, two operators transform the truth table into a variant table. Permutation operators change positions of relevant columns but do not change their values. Complementary operators (Δ) do not change the position for each column, but may change entire values of the column. Two given operators can be performed together to generate a variant table for further usages. There are 2^n columns in the table as permutation elements, so this permutation group $S(2^n)$ contains a total of $2^n!$ permutation operators, and its complementary group $B_2^{2^n}$ includes a total of 2^{2^n} complementary operators. An example of the variant table can be seen in Fig. 1b.

4 Variant Method of Pseudorandom Number Generation

Input: n, P, Δ, m, L variables, $n \in N$, $P \in S(2^n)$, $\Delta, L, m \in B_2^{2^n}$
Output: $\{K_m, K_{m+1}, \ldots., K_{m+L-1}\}L \cdot 2^n$ bit sequences
Method: The process for pseudorandom number generation can be seen in Fig. 2. n is the input variable number. Using n variables, a standard truth table can be constructed in 2^n columns and 2^{2^n} rows. P is a given permutation operator $P = (P_{2^n-1} \ldots P_I \ldots P_0)$, $\quad P \in S(2^n)$, where P_I corresponds to the I-th column. A given complementary operator $\Delta \in B_2^{2^n}$, $\Delta = (\Delta_{2^n-1} \ldots \Delta_I \ldots \Delta_0)$, $\Delta_I \in B_2$ shows that the operator is performed on the I-th column, if $\Delta_I = 0$, all values of the column are reversed and if $\Delta_I = 1$, all values are invariant. $0 \leq m < 2^{2^n}$ is an initial position for output sequences; from K_m, L conditions, $\{K_{m+i}\}_{i=0}^{L-1}$ are output generated 0–1 bit sequences.

5 Sequence Generation Example

For convenient understanding procedure, an example is selected to show in the n = 2 case shown in Fig. 3. Parameters are initialized to arbitrary values: $n = 2$, $P = (1203)$, and $\Delta = (0110)$.

After the table is generated, the pseudorandom sequence can read off the table. For $m = 4$ and $L = 6$ conditions, a random number starting at position 4 of the variant table containing six elements can be found.

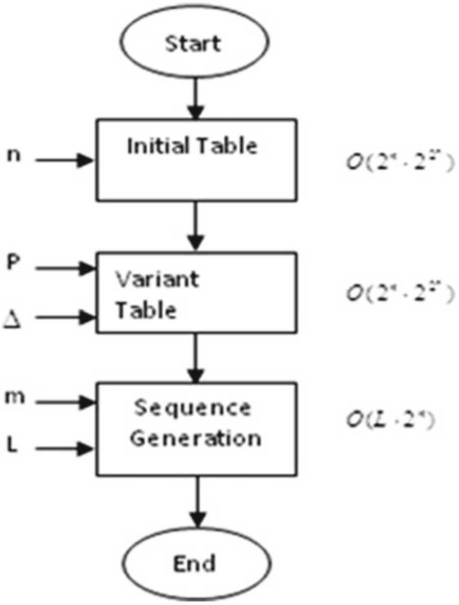
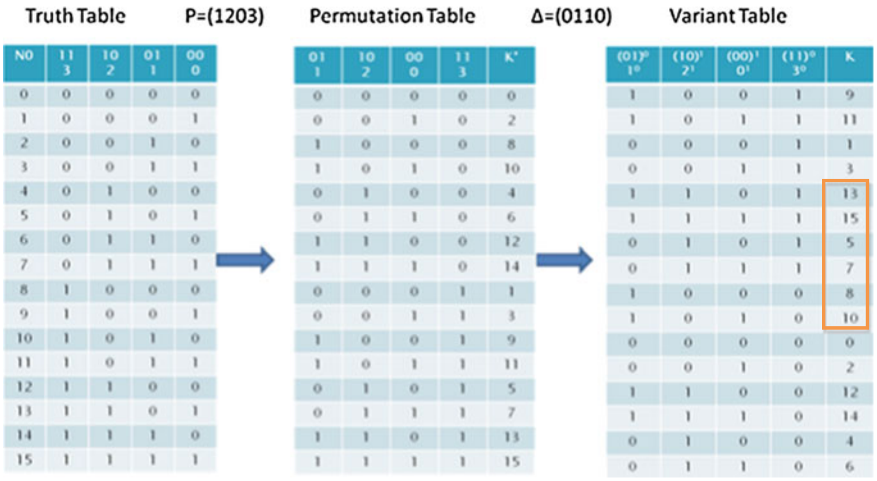


Fig. 2 Variant method of random number generation



Truth Table P=(1203) Permutation Table Δ=(0110) Variant Table

NO	11 3	10 2	01 1	00 0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

01 1	10 2	00 0	11 3	K'
0	0	0	0	0
0	0	1	0	2
1	0	0	0	8
1	0	1	0	10
0	1	0	0	4
0	1	1	0	6
1	1	0	0	12
1	1	1	0	14
0	0	0	1	1
0	0	1	1	3
1	0	0	1	9
1	0	1	1	11
0	1	0	1	5
0	1	1	1	7
1	1	0	1	13
1	1	1	1	15

$(01)^0$ 1^0	$(10)^1$ 2^1	$(00)^1$ 0^1	$(11)^0$ 3^0	K
1	0	0	1	9
1	0	1	1	11
0	0	0	1	1
0	0	1	1	3
1	1	0	1	13
1	1	1	1	15
0	1	0	1	5
0	1	1	1	7
1	0	0	0	8
1	0	1	0	10
0	0	0	0	0
0	0	1	0	2
1	1	0	0	12
1	1	1	0	14
0	1	0	0	4
0	1	1	0	6

Fig. 3 Example for generation of pseudorandom sequence

6 Complexity Analysis

From an application viewpoint, it is important to have the exact complexity evaluation for the method. In the initial stage, it is necessary to manipulate 2^n columns and each column with 2^{2^n} rows; the total numbers of $2^n \cdot 2^{2^n}$ bits are required. The total complexity is of order $O(2^n \cdot 2^{2^n})$.

To generate variant table values, P operations need at least to manipulate bits once and Δ operations to manipulate the same number of bits, i.e., $O(2^n \cdot 2^{2^n})$.

Selecting $L \cdot 2^n$ bits from the variant table, it is necessary to perform $O(L \cdot 2^n)$ operations.

If a full table needs to be generated as a random resource, $O(2^n \cdot 2^{2^n})$ computational complexity is required. In general, their computational complexity is $O(L \cdot 2^n) - O(2^n \cdot 2^{2^n})0 < L < 2^{2^n}$.

Maximal cycle length: under this construction, the maximal length of the pseudorandom number sequence is $2^n \cdot 2^{2^n}$ bits. For any short sequences, the output sequence has a length less than this number. No clear cycle effects can be directly observed.

7 Conclusion

It is important to design this new PRNG method to use variant logic construction. Since P and Δ potentially have a huge configuration space $2^n! \times 2^{2^n}$ times larger than classical logic function spaces. Exploring how difficulties for this mechanism to be decoded will be the main issue for coming cryptologist's theoretical targets. In addition, it is important to understand what type of distribution will be relevant to this generation mechanism. Owing to intrinsic complexity of variant logic construction, this provides potential barriers to protect this type of sequences decoded directly.

Considering PRNG placed in the central part of stream cipher mechanism, and stream cipher technologies are more and more important in advanced network security environment, higher performance methodology and relevant implementation will be useful in this field. Ongoing approaches will focus on whether this mechanism provides better PRNG methods to help different protections on side-channel attacks [1–7, 19, 20] in wider network applications to resolve practical leakage-resilient issues in the future.

References

1. M. Robshaw, *Stream ciphers*. RSA Laboratories Technical Report TR-701 (1995)
2. Y. Xiao, H. Li, S. Choi, Protection and guarantee for voice and video traffic in IEEE 802.11e Wireless LANs, in *IEEE INFOCOM* (2004), p. 11

3. NESSIE New European Schemes for Signatures, Integrity and Encryption, https://www.cosic.esat.kuleuven.be/nessie/
4. The eSTREAM Project, http://www.ecrypt.eu.org/stream/index.html
5. F.X. Standaert, T. Malkin, M. Yung, A unified framework for the analysis of side-channel key recovery attacks, in *EUROCRYPT*, (2009), pp. 443–461
6. A. Dwivedi, D. Tebben, P. Harshavardhanna, Characterizing cyber-resiliency, in *The 2010 Military Communication Conference-Unclassified Program—Cyber Security and Network Management* (IEEE press, 2010), pp. 1847–1852
7. Y. Yu, F. X. Standaert, O. Pereira, M. Yung. Practical leakage-resilient pseudorandom generator, in *CCS'2010* (ACM, 2010), pp. 141–151
8. G.B. Agnew, Random source for cryptographic systems, in *Advances in Cryptology | EUROCRYPT '87 Proceedings* (Springer-Verlag, 1988), pp. 77–81
9. C. Atkinson, A family of switching algorithms for the computer generation of beta random variables. Biometrika **66**(1), 141–145 (1979)
10. A statistical test suite for random and pseudorandom number generators for cryptographic applications (NIST Special Publication, 800-22 2010)
11. R. Davies. Hardware random number generators, in *International 15th Australian Statistical Conference* (2000)
12. D. Eastlake, S.D. Crocker, J.I. Schiller, Randomness requirements for security, RFC 1750, Internet Engineering Task Force (1994)
13. V. Kachitvichyanukul, B.W. Schmeiser, Binomial random variate generation. Commun. ACM **31**(2), 216–223 (1988)
14. M. Matsumoto, T. Nishimura, Dynamic creation of pseudorandom number generators, in *Proceedings of the Third International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing: Monte Carlo and Quasi-Monte Carlo Methods 1998* (2000), pp. 56–69
15. S.K. Park, K.W. Miller, Random number generators: good ones are hard to find. Commun. ACM **31**(10), 1192–1201 (1988)
16. M. Santha, U.V. Vazirani, Generating quasi-random sequences from slightly random sources. J. Comput. Syst. Sci. **33**, 75–87 (1986)
17. J. Zheng, C. Zheng, T.L. Kunii, *A framework of variant logic construction for cellular automata* (InTech—Open Access Publisher, 2011). http://www.intechopen.com/articles/show/title/a-framework-of-variant-logic-construction-for-cellular-automata. ISBN 978-953-307-172-5
18. J. Zheng, C. Zheng, A framework to express variant and invariant functional spaces for binary logic. Front. Electr. Electron. Eng. China. **5**(2), 163–173 (2010). http://www.springerlink.com/content/91474403127n446u/ (Higher Education Press & Springer)
19. G. Gong, Cryptographic properties of the welch-gong transformation sequence generators. IEEE Trans. Inf. Theor. **48**(11), 2837–2846 (2002)
20. B. Aissa, D. Nouredine, Designing resilient functions and bent function for stream ciphers. Georgian Electron. Sci. J Comput. Sci. Telecommun. **1**(18), 27–33 (2009)