Use the Multiplicative Cyclic Group to Generate Pseudo Random Digital Sequences

Faez Hassan Ali
Al-Rafidain University College
Baghdad-Iraq
E.Mail:faez64h@yahoo.com

Abstract:

The Multiplicative Cyclic Group [3] is one of the algebraic systems which can be used to generate a various long period digital sequences with elements ranged 0..m-1, where $m\in Z^+$, $m \geq 2$ that's done by using one (or more than one) primitive element(s) of the group. The generated sequences can be used in Stream Cipher Systems. In this paper, we introduce the mathematical process to generate a sequence from one generator of The Multiplicative Cyclic Group (MCG). The generated sequence may have no good statistical properties, so we suggest a two generators sequence. The two generators with some initial variables (keys) make a unit called MCG unit. A number of MCG units are combined with each other by a combining logical function to get MCG system. This paper includes some algorithms to describe the mentioned process and some tables describe the tests results of the generated sequences.

1 Introduction

Let $q\in Z^+$, $G\neq\phi$ be a set s.t. $G=\{1,2,\ldots,q-1\}$. Let $*$ be a multiplication operation defined as follows:

$c=a*b=a.b \pmod q$ s.t. $a,b,c\in G$, and $a^n=\underbrace{a*a*\ldots*a}_{\text{n-times}} \pmod q = b$ s.t. $a,b\in G$.

Let $\langle G,*\rangle$ be a mathematical system, G has order q-1, $\langle G,*\rangle$ is a Multiplicative Cyclic Group (MCG) iff q is a prime number.

The element $\alpha\in G$ called a primitive (generator) element iff $\alpha^i=\beta$, s.t. $1\leq i\leq q-1, \forall\beta\in G$. Notice that the generating process depend on α,i and q, so we can defined a function f represents the generating process s.t. $\beta=f(\alpha,i,q)$, $f:G\rightarrow G$, its clear that f is 1-1 and onto function.

If α a generator element of G then $\forall\beta=\alpha^i$, s.t. $\gcd(i,q-1)=1$, β is another generator of G. Therefore, there are $\phi(q-1)$ generator elements of G [1,3].

Now we can introduce ***FIND-GEN Algorithm*** ("GEN" means Generator) to find all other generators of G for prime number q from one generatorα.

FIND-GEN Algorithm

INPUT **:** q ,α ;
PROCESS **:** i := 2 ;
Repeat
i := i +1 ;
β := f (α , i , q) ;
if gcd(i,q-1) = 1 then β is another generator ;
Until i = q-2 ;
OUTPUT **:** another primitive element β ;
END.

2 One Generator's Sequence

Let $m \in Z^+$ s.t. $2 \le m \le q-1$ (prefer $m \le (q-1)/2$), the set G partitioned into m subsets name N_i, $0 \le i \le m-1$which is consists of some ordered elements $\beta_j \in G$, $1 \le j \le q-1$. The subsets

$N_i = \{\beta_j : \frac{i}{m} \cdot q < \beta_j < \frac{(i+1)}{m} \cdot q\}$ are disjoint s.t. $\bigcap_{i=0}^{m-1} N_i = \varphi$ and $\bigcup_{i=0}^{m-1} N_i = G$ [5].

Its clear that $\frac{i}{m} \cdot q, \frac{(i+1)}{m} \cdot q \in R$.

From definition of N_i we have $i < \frac{m}{q} \cdot \beta_j < i+1$, then $i \le s_j \le i+1$ s.t. $s_j = (m.\beta_j)$ div q, j=1,…,q-1.

s_j is the element j of the sequence S. s.t. $S = \{s_j\}_{j=1}^{q-1}$, $0 \le s_j \le m-1$.

The term "**div**" gives the integer part of $(m.\beta_j)/q$. Its clear that the period P(S)=q-1.

For example, let q=13, m=3, then N_0={1,2,3,4}, N_1={5,6,7,8} and N_2={9,10,11,12}, then S={0,0,0,0,1,1,1,1,2,2,2,2}.

This sequence generated without using primitive element, since β_j=1,2,…,12. But, if β_j=f(α,j,q) then the sequence will generated randomly.

The ***ONE-GEN Algorithm*** below is designed to generate S randomly by one generator.

ONE-GEN Algorithm

INPUT **:** q , α ,m ;
PROCESS **:** j := 0 ;
Repeat
j := j +1 ;
β := f (α , j , q) ;
s_j := (m.β) div q ;
Until j = q-1 ;
OUTPUT **:** the sequence S ;
END.

Table(1) shows the sequence S generated from q=13 and α=2 for m=2,…,6.

Table(1) MCG sequences with m=2,…,6.

J	β	m_1=2 S_1	m_2=3 S_2	m_3=4 S_3	m_4=5 S_4	m_5=6 S_5
1	2	0	0	0	0	0
2	4	0	0	1	1	1
3	8	1	1	2	3	3
4	3	0	0	0	1	1
5	6	0	1	1	2	2
6	12	1	2	3	4	5
7	11	1	2	3	4	5
8	9	1	2	2	3	4
9	5	0	1	1	1	2
10	10	1	2	3	3	4
11	7	1	1	2	2	3
12	1	0	0	0	0	0

Till now we get sequence with balance frequencies but have low complexity. We can notice that the period of S, theoretically, is q-1, but analytically, its (q-1)/2, since when $\beta_i+\beta_j$=q, i=1,…,(q-1)/2, j=i+(q-1)/2 that implies s_i+s_j=m-1. For example, when i=1 and j=7, so β_1=2 and β_7=11, for m=4, then s_1=0 and s_7=3.

In this manner we want to construct a method to maximize the complexity and the P(S) to be q-1. This Maximization must not effect the good randomness of S.

3 Two Generators' Sequence

Let us choose two generators α_1,α_2 of G, let x=f(α_1,j,q) and y=f(α_2,x,q) s.t. $1\le x,j\le q-1$ so y=f(α_2,x,q)= f(α_2, f(α_1,j,q),q)=g(α_1,α_2,j,q) s.t. $g:G\to G$.

To guarantee that all of the elements of G are generated, we have to prove that g is a 1-1 function. Since f is 1-1 function, $x\in G$, and α_1,α_2 are generators then $y\in G$ too, so we have to prove that if $y\neq y' \Leftrightarrow j\neq j'$.

Let $j\neq j' \Rightarrow x\neq x'$ since f is 1-1 function implies $y\neq y'$.

Let $y\neq y' \Rightarrow x\neq x'$ since f is 1-1 function, which implies $j\neq j'$.

Now the ***TWO-GEN Algorithm*** can be introduced to generate a new sequence generated from two generators with new properties.

TWO-GEN Algorithm:

INPUT **:** q , α_1 , α_2 ,m ;
PROCESS **:** j := 0 ;
Repeat
j := j +1 ;
y := g (α_1 ,α_2 , j , q) ;
s_j := (m.y) div q ;
Until j = q-1 ;
OUTPUT **:** the sequence S ;
END.

Table(2) shows the new sequence generated from q=13, α_1 =2 and α_2=6 for m=2,…,6.

Table(2) MCG sequences with m=2,…,6.

j	y	m_1=2 S_1	m_2=3 S_2	m_3=4 S_3	m_4=5 S_4	m_5=6 S_5
1	10	1	2	3	3	4
2	9	1	2	2	3	4
3	3	0	0	0	1	1
4	8	1	2	3	4	5
5	12	1	1	2	3	4
6	1	0	0	0	0	0
7	11	1	2	3	4	5
8	5	0	1	1	1	2
9	2	0	0	0	0	0
10	4	0	0	1	1	1
11	7	1	1	2	2	3
12	6	0	1	1	2	2
Frequency	0	6	4	3	2	2
	1	6	4	3	3	2
	2	--	4	3	2	2
	3	--	--	3	3	2
	4	--	--	--	2	2
	5	--	--	--	--	2

We noticed that the generated sequences have balance frequencies for different digits, this happened since G divided into m subsets N_i with approximate equal orders.

4 MCG Unit

Now we want to introduce the following variables, which are, being useful in our work:

1. Choose q prime number.
2. Choose α_1 as a generator of G.
3. Choose α_2 another generator different from α_1.
4. Choose the initial value k, s.t. $1 \le k \le q-1$.

   Take the cyclic value i = k,…,q-1,1,2,…,k-1 ; j=1,…,q-1.

   Calculate y = g(α_1 ,α_2 , i , q).

   Calculate s_j = (m.y) div q.

We want to formulate these choices in a new algorithm to introduce a new unit in order to generate sequence of period q-1, which we called it, a MCG Unit (MCGU).

MCGU is a function of five variables s.t. S = MCGU (q,α_1,α_2,k,m), which is useful in ***MCGU Algorithm*** to generate S with length $L \le q-1$, these variables ca be considered as variable keys.

MCGU Algorithm

```
INPUT      : q , α1 , α2 , k , m , L ;
PROCESS    :  i := k-1 ; j := 0 ;
               Repeat
                   i := i (mod(q-1))+1 ; j := j +1 ;
                   y := g (α1 ,α2 , i , q ) ;
                  sj := (m.y) div q ;
              Until  j = L ;
OUTPUT     : the sequence S ;
END.
```

Table(3) shows The efficiency criterion, these criterion are: Periodicity, Linear Complicity [8] and Randomness (Frequency, Run and Auto Correlation with 10 shifts) tests [2] for some binary sequences (m=2) which are generate from MCGU with different primes.

Table(3) efficiency criterions for MCG unit output results.

q	α_1	α_2	P(S)	LC	Randomness (P=Pass,F=Fail)		
					Frq	Run	AC
1009	11	102	1008	506	P	P P	F P P P P P P P P F
	601	51	1008	503	P	P P	P P P P P P P P P P
4111	60	2055	4110	2055	P	P P	P P P P F F P P P F
	507	4060	4110	2054	P	P P	P P P P P P P P P P
10771	179	1133	10770	5385	P	P P	P P P P F P F P P P
20707	83	690	20706	10354	P	P P	P P P P P P P P P P

5 MCG System:

A MCG unit can be used as a basic construction unit in MCG System (MCGS) with Combining Function (CF), which is a boolean function [9]. If S is the sequence that is generates from MCG system, the system has a F_n as a combining function with n_MCG units, then,
$S=F_n(S_1,S_2,\ldots,S_n)$ s.t. $S_i=MCGU_i(q_i,\alpha_{1i},\alpha_{2i},k_i,m)$, where $1\le i\le n$.
S_i represents the sequence i generate from the MCG unit i.

We defined the addition (+) and the multiplication (*) operations of the system, as follows:

$$\left.\begin{array}{l} s_j = s_{ij} + s_{kj} \pmod m \\ s_j = s_{ij} * s_{kj} \pmod m \end{array}\right\} \begin{array}{l} s_j\in S,\ j=1,2,\ldots \\ s_{ij}\in S_i \text{ and } s_{kj}\in S_k,\ 1\le i,k\le n. \end{array}$$

Before introducing the MCGS Algorithm to generate S with length L, we should represents the MCGU number i as MCGU(i).

MCGS Algorithm

```
INPUT      : Read n , m , L
             For i := 1 to n
                  Read q_i , α_1i , α_2i , k_i
             Endfor {i};
PROCESS    :  j := 0 ; s_j := 0;
              Repeat
                  j := j + 1 ;
                  For i := 1 to n  CALL MCGU(i) ;
                  s_j := F_n(s_1j,s_2j,…,s_nj) ;
             Until j = L ;
OUTPUT     : the sequence S ;
END.
```

We expect the MCGS has high complexity because of the high non-linearity of the function g. The periodicity of the MCGS can be calculated depending on the l.c.m. of the period of every MCGU combined in MCGU s.t. $P(S)=\text{l.c.m.}(q_1-1,q_2-1,\ldots,q_n-1)$.

Table(4) shows the output results of various MCG systems applying Periodicity, Linear Complicity, Frequency, Binary Derivative, Change point, Subblock, Run and Sequence Complicity tests for some binary sequences (m=2) using CRYPT -X'98 package [4].

Table(4) tests results of MCG systems for n=2,3 and 5 using XOR-CF.

N	Primes	P(S)	LC	FT	BDT	CPT	SBT	RT	SCT
2	101 997	24900	1091	P	P	P	P	P	P
3	199 1103 3607	65567878	4898	P	P	P	P	P	P
5	149 509 1051 1301 2003	2565654000	4899	P	P	P	P	P	P

The RNG system found by Mitchell [7], is a digital generator (m=10) with good random sequence, but it's has low complexity, with period less or equal q-1 for some primes, so we expect that the choices of the primes will drop to 35% in order to gain period equal q-1. While the choices of MCGS still open to all primes. Table (5) shows the period of some primes for RNG system with frequencies of the digits.
Table (5) shows the periods of RNG primes and frequencies of sequence digits.

Table (5) Period's of RNG primes and frequencies of sequence digits.

Primes	**Period**	**Frequency**									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
991	495	56	55	60	46	46	53	53	40	47	49
997	166	30	15	19	16	12	12	15	19	14	24
1003	464	54	38	54	62	38	54	33	40	56	50
1013	253	27	29	29	22	20	27	30	27	26	31
1019	1018	104	103	103	103	103	103	102	102	103	102

6 Conclusions & Recommendations

1. If we compare the MCGU and LFSR, we get the following differences:
   i. For unknown algorithm, the LFSR variables are the length, tap and initial values are unknown, but MCGU variables are α_1,α_2,q,k and m are all unknowns.
   ii. For known algorithm, the initial value (basic key) is unknown only, but in MCGU, α_1, α_2, q, k are unknown which are can be considered as initial values.

iii. The periodicity of the sequence generated from LFSR with length r is 2^r-1, but the period of the sequence generated from MCG is q-1 for each choice of two generators, there are ϕ(q-1)*(ϕ(q-1)-1) different choices.
iv. The common generated sequence from LFSR is binary, but in MCGU, the sequence is digital ($1<m\leq q-1$).
v. The length, tap and initial values of LFSR can be found from some available length of the generated sequence by using Massey algorithm [6], but its not easy to find the initial value of the MCGU in spite the availability of the generated sequence because of the high non-linearity of the function g.

2. The MCGU can be developed to increase its periodicity, complexity and randomness by using other non-used generators of G.
3. We have to suggest digital randomness tests in order to test the randomness of the generated digital sequences (m>2) from MCGU.

7 <u>References</u>

[1]. Andrews, G. G, "**Number Theory**", Dover Publications, October 1994.

[2]. Bennett, D. J. "**Randomness**", Harvard University Press, October 1999.

[3]. Gilbert, W. J. "**Modern Algebra with Applications**", Wiley-Interscince, March 2002.

[4]. Gustafson, H., Dawson, E. "**A Computer Package for Measuring the Strength of Encryption Algorithm**", Computer & Security Vol.13, No.8, 1994.

[5]. Johnson, D. W. and Johnson, F. P., "**Joining Together: Group Theory and Group Skills**", Allyn & Bacon, July 2002.

[6]. Massey, J.L., "**Shift Register Synthesis and BCH Decoding**", IEEE Transaction on Information Theory, Vol. IT-15, No.1, 1969.

[7]. Mitchell, D. W., "**A Nonlinear Random Number Generator with Known, Long Cycle Length**", Dept. of Economics, West Virginia University, Morgantown WV 26506-602 USA 1993.

[8]. Schneier B., "**Applied Cryptography**", John Wiley & Sons, 1995.

[9]. Whitesitt, J. E, "**Boolean Algebra and its Application**", Dover Publications, April 1995.