SUT Journal of Mathematics
Vol. 47, No. 1 (2011), 73–90

A pseudorandom number generator using an Artin-Schreier tower

Huiling Song, Hiroyuki Ito and Yukinori Kitadai

(Received April 27, 2011; Revised May 18, 2011)

Abstract. In this paper, we propose a new pseudorandom number generator (AST) using an Artin-Schreier tower, which is a modified version of the twisted generalized feedback shift register (TGFSR). In TGFSR, the period is depending on the order of its multiplied matrix, and it is difficult to get the theoretical upper bound in general. Using the recursive structure of Artin-Schreier towers, we define a matrix with a parameter whose order is expected to be close to fairly near the upper bound. After examining some properties of this matrix, we give an algorithm of a new pseudorandom number generator AST which produce a sequence of numbers with a long period. Finally, we report the results of TestU01 to qualify that it has a good statistical property, although its generation speed is rather slower than TGFSR.

AMS 2010 *Mathematics Subject Classification.* 12Y05, 11K45.

Key words and phrases. Artin-Schreier tower, pseudorandom number generator.

§1. Introduction

A pseudorandom number generator is an algorithm for generating a sequence x_i $(i = 0, 1, 2, \dots)$ of numbers which is widely used in various fields. Each x_i of a sequence is a word with components 0 and 1 of size w, and is produced by a generator from initial seeds. One of important parameters which represent the performance of a pseudorandom number generator is its period, which is the smallest integer p such that $x_{i+p} = x_i$ holds for every integer i. A good generator must have a sufficiently long period. The generalized feedback shift register (GFSR) algorithm suggested by Lewis and Payne ([5]) is a widely used pseudorandom number generator. But the period of a GFSR sequence $2^n - 1$ is far smaller than the theoretical upper bound $2^{nw} - 1$, where n is the number of initial seeds $x_{n-1}, \cdots, x_1, x_0$. The twisted GFSR (TGFSR) generator ([7], [8]) resolves this drawback. Furthermore, Mersenne Twister (MT) introduced by Matsumoto and Nishimura provides a super astronomical period and has good statistical properties (cf. [9]). In this paper we propose a new generator AST using an Artin-Schreier tower. This generator produces a sequence with a long period which is conjectured to be fairly near to the theoretical upper bound, and gives a sequence whose period is longer than MT's by choosing suitable parameters. In addition to the theoretical properties, the standard statistical test for pseudorandom number generators, TestU01 [11], shows that our new generator AST has a good experimental property as a pseudorandom number generator.

Here is the plan of this paper. In sections 2 and 3, we will briefly introduce several examples of pseudorandom number generators related with our new generator AST and define a linear recurrence equation. Next, we will describe a construction of finite fields using the specific Artin-Schreier tower starting from the binary field $\mathbb{F}_2$ and a multiplication algorithm in section 4. In section 5, we will define certain matrices as an application of section 4, prove the properties of these matrices and propose a new pseudorandom number generator using them. In section 6, we will exhibit the results of TestU01.

Here are some conventions. Throughout this paper, the notation $\mathbb{F}_q$ is used as a finite field of q elements, and 2^{2^n} stands for $2^{(2^n)} = \exp(2^n \log 2)$ as usual. Basic notations and definitions on pseudorandom numbers are refered to [2] and [4] and general facts on finite fields are refered to [6].

§2. Pseudorandom number generators related to AST

We will briefly introduce several related pseudorandom number generators according to the formulation by Matsumoto-Kurita [7], [8] since our new pseudorandom number generator AST is based on the TGFSR.

2.1. GFSR

Definition 1. (Lewis-Payne [5]) Let n, m and w be positive integers with $n > m$. The generalized feedback shift register (GFSR) generator is based on the linear recurring equation

$$x_{j+n} := x_{j+m} + x_j \qquad (j = 0, 1, \dots), \tag{2.1}$$

where each x_j is a word with components 0 or 1 of size w and $+$ means the addition as $\mathbb{F}_2$-vectors. Thereby, this algorithm generates the same number of m-sequences as the word length in parallel.

The period of a GFSR sequence is $2^n - 1$ which is far smaller than the theoretical upper bound $2^{nw} - 1$.

2.2. TGFSR

Definition 2. (Matsumoto-Kurita [7], [8]) Let n, m and w be positive integers with $n > m$. The twisted GFSR generator (TGFSR) is the same as the GFSR generator except that it is based on the linear recurrence equation

$$x_{j+n} := x_{j+m} + x_j A \qquad (j = 0, 1, \dots), \tag{2.2}$$

where each x_j is a word regarded as a row vector over $\mathbb{F}_2$ of size w, A is a $w \times w$ matrix with entries in $\mathbb{F}_2$ and $+$ means the addition as $\mathbb{F}_2$-vectors. With a suitable choice of n, m, and A, the TGFSR generator attains the maximal period $2^{nw} - 1$, that is, it produces all possible states except the zerostate in a period.

The TGFSR generator improves on the drawback of the GFSR generator as the period of the generated sequence attains the theoretical upper bound $2^{nw} - 1$. However, the matrix A should be chosen so that $x_j A$ can be calculated fast for practical use. For example, one may choose A of the form

$$\begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ a_0 & a_1 & \cdots & a_{w-2} & a_{w-1} \end{pmatrix}$$

whose characteristic polynomial is given by $\varphi_A(t) = t^w + \sum_{i=0}^{w-1} a_i t^i$. It is hard to find a matrix in general which has the both properties that the calculation of $x_j A$ can be done fast and the period of it attains the theoretical upper bound.

2.3. Mersenne Twister

Mersenne Twister is a pseudorandom number generator which adds two ideas to the TGFSR to attain these records. One is the adoption of an incomplete array to realize a Mersenne-prime period, and the other is the existence of fast algorithm to test the primitivity of the characteristic polynomial of a linear recurrence.

Definition 3 (Matsumoto-Nishimura [9])**.** Mersenne Twister is based on the following linear recurrence equation

$$x_{j+n} = x_{j+m} + (x_j^u | x_{j+1}^l) A \qquad (j = 0, 1, \dots), \tag{2.3}$$

where x_k^l stands for the extraction of the lower r bits of x_k, x_k^u stands for the extraction of the upper $w-r$ bits of x_k, and $(x_j^u|x_{j+1}^l)$ stands for the concatenation of x_j^u and x_{j+1}^l. It requires several constants, an integer n which is the degree of the recurrence equation, an integer r with $0 \leq r \leq w-1$, an integer m with $1 \leq m \leq n$, and a $w \times w$ matrix A with entries in $\mathbb{F}_2$. Let $x_{n-1}, \cdots, x_1, x_0$ be initial seeds. Then, the generator produces x_n by the above recurrence equation with $j = 0$. By putting $j = 1, 2, \ldots$, the generator determines $x_{n+1}, x_{n+2}, \ldots$.

If one eliminates the lower r bits from the $(n \times w)$-array $x_{j+n-1}, \ldots, x_{j+1}, x_j$, then the dimension of the state space is $nw-r$, which can be taken any number. This is the great advantage of MT. See [9] for details.

§3. Linear recurrence equations on finite fields

In this section, we explain how to translate n-th order linear recurrence equations into first order linear recurrence equations. We continue to use the notations as in the previous section.

Definition 4. Let W be a w-dimensional vector space over $\mathbb{F}_2$ which we regard as the state space of the generator. Let $g: W^n \to W$ be a linear state map. Let $x_{n-1}, \ldots, x_1, x_0$ be initial nw-arrays with $x_0, \ldots, x_{n-1} \in W$. We define the linear recurrence equation

$$x_{j+n} = g(x_{j+n-1}, \ldots, x_j)$$

as n-th order linear recurrence.

Let $S := W^n$, and $f : S \to S$ be a linear state transition map. Then n-th order linear recurrence equation can be transformed into the first order linear recurrence equation as follow:

$$f(x_{j+n-1}, \ldots, x_j) = (g(x_{j+n-1}, \ldots, x_j), x_{j+n-1}, \ldots, x_{j+1}) \quad (j = 0, 1, \cdots).$$

For example, TGFSR (2.2) can be transformed into the first order linear recurrence map as:

$$f : (x_{j+n-1}, \ldots, x_{j+1}, x_j) \mapsto (x_{j+m} + x_j A, x_{j+n-1}, \ldots, x_{j+2}, x_{j+1}) \qquad (j = 0, 1, \cdots),$$

where f is a linear state transition map, which multiply nw-bit vector by matrix B,

$$B = \begin{pmatrix} & I_w & & & \\ & & I_w & & \\ I_w & & & \ddots & \\ & & & & I_w \\ A & & & & \end{pmatrix}.$$

Since we have the equation

$$(x_{j+n}, x_{j+n-1}, \ldots, x_{j+1}) = (x_{j+n-1}, x_{j+n-2}, \ldots, x_j)B$$

for $j = 0, 1, \ldots$, it is clear that the period of the sequence of numbers is equal to just the order of the matrix B.

Similarly, Mersenne Twister (2.3) can be transformed as:

$$\begin{aligned} &f : (x_{j+n-1}, x_{j+n-2}, \ldots, x_{j+1}, \{x_j^u\}) \\ &\mapsto (x_{j+m} + (x_j^u | x_{j+1}^l)A, x_{j+n-1}, \ldots, x_{j+2}, \{x_{j+1}^u\}) \ (j = 0, 1, \cdots). \end{aligned}$$

Now, let B be $(nw - r) \times (nw - r)$-array matrix as follows:

$$B = \begin{pmatrix} & I_w & & & \\ & & I_w & & \\ I_w & & & \ddots & \\ & & & & I_{w-r} \\ S & & & & \end{pmatrix},$$

where $S = \begin{pmatrix} 0 & I_r \\ I_{w-r} & 0 \end{pmatrix} A$, then, one gets

$$(x_{j+n}, x_{j+n-1}, \ldots, x_{j+1}^u) = (x_{j+n-1}, x_{j+n-2}, \ldots, x_j^u)B.$$

Note that MT can attain the maximal period (see [9]) with a suitable choice of B, but it is difficult to find such B with maximal period for TGFSR.

Using the expression by linear recurrence equations, we will introduce a pseudorandom number generator in the following sections which can produce a pseudorandom number sequence whose period is expected to be close to the maximum, and even longer than MT with suitable parameters. It also has the merit that it can be easily implemented to computers because of its simple structure.

§4. Artin-Schreier towers

In this section we give a construction of finite fields using the Artin-Schreier tower, which has a beautiful recursive structure. We also give the multiplication algorithm using this recursive structure.

4.1. Definition of the Artin-Schreier tower

Definition 5 (Ito-Kajiwara-Song [3])**.** Let K_0 be the prime field $\mathbb{F}_2 = \{0, 1\}$ and $f_1(x) := x^2 + x + 1$ be a polynomial in $\mathbb{F}_2[x]$, we define

$$K_1 := K_0[x]/(f_1(x)) = K_0(\alpha_1) = \mathbb{F}_2(\alpha_1) = \mathbb{F}_{2^2},$$

where $\alpha_1 := \bar{x} \in K_1$ be the image of x in K_1. Suppose that α_{r-1} and $f_{r-1}(x)$ are defined for $r \geq 2$. Define $f_r(x)$, K_r and α_r as follows:

$$\begin{aligned}
f_r(x) &:= x^2 + x + (\alpha_1 \cdots \alpha_{r-1}),\\
K_r &:= K_{r-1}[x]/(f_r(x)),\\
\alpha_r &:= \bar{x} \in K_r = K_{r-1}(\alpha_r).
\end{aligned}$$

Then we have the tower of finite fields inductively:

$$K_0 \subset K_1 = K_0(\alpha_1) \subset K_2 = K_1(\alpha_2) \subset \cdots \subset K_r = K_{r-1}(\alpha_r) \subset \cdots .$$

We call this sequence of extensions the *Artin-Schreier tower*.

The polynomial f_r in the definition is known to be irreducible over K_{r-1} by analyzing the Artin-Schreier extensions (see [3]). Because of its natural definition of the tower, this Artin-Schreier tower has a beautiful recursive structure which is a key structure of our current work.

Let us explain the recursive structure. Since the basis of K_1 over K_0 is 1 and α_1, we have an expression

$$K_1 = \{s_0 1 + t_0 \alpha_1 \mid s_0, t_0 \in K_0\}.$$

The basis of K_2 over K_1 is 1 and α_2, then the basis of K_2 over K_0 is $1, \alpha_1, \alpha_2, \alpha_1\alpha_2$, thus we have an expression

$$\begin{aligned}
K_2 &= \{s_1 1 + t_1 \alpha_2 \mid s_1, t_1 \in K_1\}\\
&= \{s_{01} 1 + t_{01}\alpha_1 + s_{02}\alpha_2 + t_{02}\alpha_1\alpha_2 \mid s_{01}, t_{01}, s_{02}, t_{02} \in K_0\}.
\end{aligned}$$

Similarly, the basis of K_r over K_{r-1} is 1 and α_r, so that we have the basis of K_r over K_0 as

$$\underbrace{\underbrace{(\overbrace{\overbrace{\overbrace{1,\alpha_1}^{2}\;\Big|\;\alpha_2,\alpha_1\alpha_2}^{2^2}\;\Big|\;\ldots}^{2^{r-2}}\;\Big|\;\alpha_{r-1},\alpha_1\alpha_{r-1},\ldots,\alpha_1\cdots\alpha_{r-1}}_{2^{r-1}}\;\Big|\;\alpha_r,\alpha_1\alpha_r,\ldots,\alpha_1\cdots\alpha_r)}_{2^r}.$$

Note that the last half of this basis is given by multiplying α_r with the first half of this basis which is the recursive structure of the basis of this extensions.

4.2. The multiplication algorithm

Using the recursive structures of the basis exhibited above, we can make an algorithm of multiplication on the Artin-Schreier extensions without the power expression of each element. First we recall a vector expression of the elements of the fields (cf. [10]). We write $s_1 + s_2\alpha_r \in K_r$ as (s_1, s_2) with $s_1, s_2 \in K_{r-1}$. And we also write the multiplication of two elements of K_r, $(s_1 + s_2\alpha_r)(t_1 + t_2\alpha_r)$ as $(s_1, s_2)(t_1, t_2)$. Taking the multiplication of two elements $s_1 + s_2\alpha_r, t_1 + t_2\alpha_r \in K_r$ inside the field K_r, we have

$$(s_1 + s_2\alpha_r)(t_1 + t_2\alpha_r) = s_1t_1 + (s_1t_2 + s_2t_1)\alpha_r + s_2t_2\alpha_r^2.$$

Since α_r is a root of $f_r(x) = x^2 + x + \alpha_{r-1}\cdots\alpha_1$, we have

$$\alpha_r^2 = \alpha_r + \alpha_{r-1}\cdots\alpha_1,$$

thus we can write

$$(s_1 + s_2\alpha_r)(t_1 + t_2\alpha_r) = (s_1t_1 + s_2t_2\alpha_{r-1}\cdots\alpha_1) + (s_1t_2 + s_2t_1 + s_2t_2)\alpha_r.$$

Since $(s_1t_1 + s_2t_2\alpha_{r-1}\cdots\alpha_1)$ and $(s_1t_2 + s_2t_1 + s_2t_2)$ are the elements of K_{r-1} we have

$$(s_1, s_2)(t_1, t_2) = (s_1t_1 + s_2t_2\alpha_{r-1}\cdots\alpha_1, s_1t_2 + s_2t_1 + s_2t_2).$$

Doing the above operation recursively, we can express an element of K_r as a vector of length 2^r over K_0 with the basis shown in the end of the last subsection. Note that $\alpha_{r-1}\cdots\alpha_1$ can be regarded as the vector $(0, \cdots, 0, 1)$ over K_0. By the argument above, we have the matrix which expresses the multiplication of two elements.

Theorem 1 (Song-Ito [10]). *For elements (s_1, s_2) and (t_1, t_2) of K_1, let $A^{(1)}(t_1, t_2)$ be the 2×2 matrix defined by $A^{(1)}(t_1, t_2) := \begin{pmatrix} t_1 & t_2 \\ t_2 & t_1 + t_2 \end{pmatrix}$. Then the multiplication of (s_1, s_2) and (t_1, t_2) is expressed as*

$$(s_1, s_2)(t_1, t_2) = (s_1, s_2)\begin{pmatrix} t_1 & t_2 \\ t_2 & t_1 + t_2 \end{pmatrix}.$$

Similarly, for each $r \geq 1$ and two elements $(s_1, s_2, \ldots, s_{2^r})$ and $(t_1, t_2, \cdots, t_{2^r})$ of K_r, define the $2^r \times 2^r$ matrix $A^{(r)}(t_1, \ldots, t_{2^r})$ as $\begin{pmatrix} S & T \\ U & V \end{pmatrix}$, where $2^{r-1} \times 2^{r-1}$ matrices S, T, U, V are defined recursively as follows:

$$\begin{aligned} S &= A^{(r-1)}(t_1, \ldots, t_{2^{(r-1)}}), \\ T &= A^{(r-1)}(t_{2^{(r-1)}+1}, \ldots, t_{2^r}), \\ U &= A^{(r-1)}(t_{2^{(r-1)}+1}, \ldots, t_{2^r}) \cdot A^{(r-1)}(0, \ldots, 1), \\ V &= A^{(r-1)}(t_1, \ldots, t_{2^{(r-1)}}) + A^{(r-1)}(t_{2^{(r-1)}+1}, \ldots, t_{2^r}). \end{aligned}$$

Then the matrix $A^{(r)}(t_1,\dots,t_{2^r})$ *gives the multiplication of* $(s_1,s_2,\dots,s_{2^r})$ *and* $(t_1,t_2,\cdots,t_{2^r})$ *as*

$$\begin{aligned}(s_1,s_2,\dots,s_{2^r})(t_1,t_2,\dots,t_{2^r}) &= (s_1,s_2,\dots,s_{2^r})\cdot A^{(r)}(t_1,\dots,t_{2^r})\\ &= (s_1,s_2,\dots,s_{2^r})\begin{pmatrix} S & T\\ U & V\end{pmatrix}.\end{aligned}$$

Proof. The case for K_1 is clear from the argument above the theorem.

When $r=2$, let $(s_1,s_2,s_3,s_4),(t_1,t_2,t_3,t_4)$ be two elements of K_2, then

$$\begin{aligned}&(s_1,s_2,s_3,s_4)(t_1,t_2,t_3,t_4) = ((s_1,s_2)+(s_3,s_4)\alpha_2)((t_1,t_2)+(t_3,t_4)\alpha_2)\\ &= (s_1,s_2)(t_1,t_2)+((s_1,s_2)(t_3,t_4)+(s_3,s_4)(t_1,t_2))\alpha_2+(s_3,s_4)(t_3,t_4)\alpha_2^2\\ &= ((s_1,s_2)(t_1,t_2)+(s_3,s_4)(t_3,t_4)\alpha_1)\\ &\qquad +((s_1,s_2)(t_3,t_4)+(s_3,s_4)(t_1,t_2)+(s_3,s_4)(t_3,t_4))\alpha_2\\ &= ((s_1,s_2)\cdot A^{(1)}(t_1,t_2)+(s_3,s_4)\cdot A^{(1)}(t_3,t_4)\cdot A^{(1)}(0,1))\\ &\quad +((s_1,s_2)\cdot A^{(1)}(t_3,t_4)+(s_3,s_4)\cdot A^{(1)}(t_1,t_2)+(s_3,s_4)\cdot A^{(1)}(t_3,t_4))\alpha_2\\ &= (s_1,s_2,s_3,s_4)\begin{pmatrix} A^{(1)}(t_1,t_2) & A^{(1)}(t_3,t_4)\\ A^{(1)}(t_3,t_4)\cdot A^{(1)}(0,1) & A^{(1)}(t_1,t_2)+A^{(1)}(t_3,t_4)\end{pmatrix}\\ &= (s_1,s_2,s_3,s_4)\cdot A^{(2)}(t_1,t_2,t_3,t_4).\end{aligned}$$

For the case K_r, let $(s_1,\dots,s_{2^r})$ and $(t_1,\dots,t_{2^r})$ be two elements of K_r. We obtain the following by induction:

$$\begin{aligned}&(s_1,\dots,s_{2^r})(t_1,\dots,t_{2^r})\\ &\quad = ((s_1,\dots,s_{2^{r-1}})+(s_{2^{r-1}+1},\dots,s_{2^r})\alpha_r)\\ &\qquad\quad \times((t_1,\dots,t_{2^{r-1}})+(t_{2^{r-1}+1},\dots,t_{2^r})\alpha_r)\\ &\quad = (s_1,\dots,s_{2^{r-1}})(t_1,\dots,t_{2^{r-1}})\\ &\qquad +((s_1,\dots,s_{2^{r-1}})(t_{2^{r-1}+1},\dots,t_{2^r})+(s_{2^{r-1}+1},\dots,s_{2^r})(t_1,\dots,t_{2^{r-1}}))\alpha_r\\ &\qquad +(s_{2^{r-1}+1},\dots,s_{2^r})(t_{2^{r-1}+1},\dots,t_{2^r})\alpha_r^2\end{aligned}$$

$$
\begin{aligned}
&= (s_1, \ldots, s_{2^{r-1}})(t_1, \ldots, t_{2^{r-1}}) \\
&\quad + ((s_1, \ldots, s_{2^{r-1}})(t_{2^{r-1}+1}, \ldots, t_{2^r}) + (s_{2^{r-1}+1}, \ldots, s_{2^r})(t_1, \ldots, t_{2^{r-1}}))\alpha_r \\
&\quad + (s_{2^{r-1}+1}, \ldots, s_{2^r})(t_{2^{r-1}+1}, \ldots, t_{2^r})(\alpha_r + (\alpha_{r-1} \cdots \alpha_1)) \\
&= (s_1, \ldots, s_{2^{r-1}})(t_1, \ldots, t_{2^{r-1}}) \\
&\quad + ((s_1, \ldots, s_{2^{r-1}})(t_{2^{r-1}+1}, \ldots, t_{2^r}) + (s_{2^{r-1}+1}, \ldots, s_{2^r})(t_1, \ldots, t_{2^{r-1}}))\alpha_r \\
&\quad + (s_{2^{r-1}+1}, \ldots, s_{2^r})(t_{2^{r-1}+1}, \ldots, t_{2^r})(\alpha_r + A^{(r-1)}(0, \ldots, 1)) \\
&= (s_1, \ldots, s_{2^r}) \begin{pmatrix} S & T \\ U & V \end{pmatrix} \\
&= (s_1, \ldots, s_{2^r}) \cdot A^{(r)}(t_1, \ldots, t_{2^r}).
\end{aligned}
$$

□

Along the way, we get an algorithm for multiplication of two elements of K_r as below:

Algorithm
Input: $r, (s_1, \ldots, s_{2^r}), (t_1, \ldots, t_{2^r})$
Output: $(u_1, \ldots, u_{2^r})$
Procedure:
1. $M_i^0 \leftarrow t_i$ $(1 \leq i \leq 2^r)$, $U^0 \leftarrow 1$;
2. for $(j = 1, j \leq r, j = j + 1)$;
   for $(i = 1, i \leq 2^{r-j}, i = i + 1)$;

$$M_i^j \leftarrow \begin{pmatrix} M_{2i-1}^{(j-1)} & M_{2i}^{(j-1)} \\ M_{2i}^{(j-1)} U^{(j-1)} & M_{2i-1}^{(j-1)} + M_{2i}^{(j-1)} \end{pmatrix}$$

$$U^j \leftarrow \begin{pmatrix} 0 & U^{(j-1)} \\ (U^{(j-1)})^2 & U^{(j-1)} \end{pmatrix}$$

3. $(u_1, \ldots, u_{2^r}) \leftarrow (s_1, \ldots, s_{2^r}) M_1^r$
4. return $(u_1, \ldots, u_{2^r})$

§5. New generator using the Artin-Schreier tower

In this section, we define a matrix, called B_r, with a parameter $r > 0$, which can be proved to have a large order, and give a new linear recurrence. By the new linear recurrence, we have a new pseudorandom number generator, which conjecturally attains near the maximum period.

5.1. The definition and the order of B_r

Definition 6. The multiplication of $\boldsymbol{x}, (1+\alpha_r) \in \mathbb{F}_{2^{2^r}}$ can be written as follows:

$$\boldsymbol{x}(1+\alpha_r) = \boldsymbol{x}(\underbrace{1,0,\dots,0}_{2^{r-1}}|1,0,\dots,0) = \boldsymbol{x}\cdot A^{(r)}(\underbrace{1,0,\dots,0}_{2^{r-1}}|1,0,\dots,0)$$
$$= \boldsymbol{x}\cdot\begin{pmatrix} I & I \\ A^{(r-1)}(\underbrace{0,\dots,0,1}_{2^{r-1}}) & O \end{pmatrix}.$$

Then we define the $2^r \times 2^r$ matrix B_r as $A^{(r)}(\underbrace{1,0,\dots,0}_{2^{r-1}}|1,0,\dots,0)$, and the $2^{r-1}\times 2^{r-1}$ matrix A_{r-1} as $A^{(r-1)}(\underbrace{0,\dots,0,1}_{2^{r-1}})$. Here I is the identity matrix, and O is the zero matrix.

Note that A_{r+1} can be written as $A_{r+1} = \begin{pmatrix} 0 & A_r \\ {A_r}^2 & A_r \end{pmatrix}$ by Theorem 1, and B_{r+1} can be written as $B_{r+1} = \begin{pmatrix} I & I \\ A_r & 0 \end{pmatrix}$ by the definition. Although the order of B_r cannot reach the maximum order $2^{2^r}-1$ of $2^r\times 2^r$ matrices, it is fairly big and conjecturally near to the upper bound. We are going to evaluate both the orders of A_r and B_r after preparing some lemmas.

Lemma 1. *For $n \geq 2$ and $k \geq 1$, write $\varphi_k(A_n) := A_n^{2^{k-1}} + A_n^{2^{k-2}} + \cdots + A_n^2 + A_n$. Then we have*

$${A_{n+1}}^{2^k} = \begin{pmatrix} A_n^{2^k}\varphi_k(A_n) & A_n^{2^k} \\ A_n^{2^k+1} & A_n^{2^k}(\varphi_k(A_n)+I) \end{pmatrix}, \; {B_{n+1}}^{2^k} = \begin{pmatrix} \varphi_k(A_n)+I & I \\ A_n & \varphi_k(A_n) \end{pmatrix}.$$

Proof. Since $A_{n+1} = \begin{pmatrix} O & A_n \\ {A_n}^2 & A_n \end{pmatrix}$ by definition, then

$${A_{n+1}}^2 = \begin{pmatrix} {A_n}^3 & {A_n}^2 \\ {A_n}^3 & {A_n}^3 + {A_n}^2 \end{pmatrix} = \begin{pmatrix} A_n^{2^1}\varphi_1(A_n) & A_n^{2^1} \\ A_n^{2^1+1} & A_n^{2^1}(\varphi_1(A_n)+I) \end{pmatrix}.$$

Suppose that ${A_{n+1}}^{2^{k-1}} = \begin{pmatrix} A_n^{2^{k-1}}\varphi_{k-1}(A_n) & A_n^{2^{k-1}} \\ A_n^{2^{k-1}+1} & A_n^{2^{k-1}}(\varphi_{k-1}(A_n)+I) \end{pmatrix}$. Then we can calculate $A_{n+1}^{2^k}$ as follows:

$$\begin{aligned}
{A_{n+1}}^{2^k} &= ({A_{n+1}}^{2^{k-1}})^2 \\
&= \begin{pmatrix} A_n^{2^k}\varphi_{k-1}^2(A_n) + A_n^{2^{k-1}+2^{k-1}+1} & A_n^{2^{k-1}+2^{k-1}} \\ A_n^{2^{k-1}+2^{k-1}+1} & A_n^{2^{k-1}+2^{k-1}+1} + A_n^{2^k}\varphi_{k-1}^2(A_n) + A_n^{2^k} \end{pmatrix} \\
&= \begin{pmatrix} A_n^{2^k}\varphi_k(A_n) & A_n^{2^k} \\ A_n^{2^k+1} & A_n^{2^k}(\varphi_k(A_n)+I) \end{pmatrix}.
\end{aligned}$$

We get the first assertion by induction, and we can prove the remaining assertion similarly. □

Lemma 2. *If* $A_n^{\frac{2^{2^n}-1}{3}} = I$ *is satisfied, then* $\varphi_{2^n}(A_n) + I = O$ *holds.*

Proof. We prove by induction. When $n = 1$, $\varphi_{2^1}(A_1) + I = A_1^2 + A_1 + I = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = O$.

Suppose that $\varphi_{2^n}(A_n) + I = O$. Let us express

$$\varphi_{2^{n+1}}(A_n) + I = A_{n+1}^{2^{2^{n+1}-1}} + A_{n+1}^{2^{2^{n+1}-2}} + \cdots + A_{n+1}^2 + A_{n+1} + I$$

in the form $\begin{pmatrix} T_1 & T_2 \\ T_3 & T_4 \end{pmatrix}$.

By Lemma 1, we can express each term as a block matrix, thus we have

$$\begin{aligned}
T_3 &= A_n^{2^{2^{n+1}-1}+1} + A_n^{2^{2^{n+1}-2}+1} + \cdots + A_n^{2^{2^{n+1}-2^n}+1} + A_n^{2^{2^n-1}+1} \\
&\qquad + \cdots + A_n^{2+1} + A_n^{1+1} + O \\
&= A_n(A_n^{2^{2^{n+1}-1}} + A_n^{2^{2^{n+1}-2}} + \cdots + A_n^{2^{2^{n+1}-2^n}} + A_n^{2^{2^n-1}} + \cdots + A_n^2 + A_n) \\
&= A_n[(\varphi_{2^n}(A_n))^{2^{2^n}} + (\varphi_{2^n}(A_n))] \\
&= A_n(I^{2^{2^n}} + I) \\
&= O,
\end{aligned}$$

$$\begin{aligned}
T_1 &= A_n^{2^{2^{n+1}-1}} \varphi_{2^{n+1}-1}(A_n) + \cdots + A_n^{2^{2^{n+1}-2^n+1}} \varphi_{2^{n+1}-2^n+1}(A_n) \\
&\quad + A_n^{2^{2^{n+1}-2^n}} \varphi_{2^{n+1}-2^n}(A_n) + A_n^{2^{2^n-1}} \varphi_{2^n-1}(A_n) + \cdots + A_n^2 \varphi_1(A_n) \\
&\quad + O + I \\
&= A_n^{2^{2^{n+1}-1}} (A_n^{2^{2^{n+1}-2}} + A_n^{2^{2^{n+1}-3}} + \cdots + A_n^{2^{2^{n+1}-2^n}} + A_n^{2^{2^n-1}} + \cdots + A_n) \\
&\quad + \cdots \\
&\quad + A_n^{2^{2^{n+1}-2^n+1}} (A_n^{2^{2^{n+1}-2^n}} + A_n^{2^{2^n-1}} + \cdots + A_n) \\
&\quad + A_n^{2^{2^{n+1}-2^n}} (A_n^{2^{2^n-1}} + \cdots + A_n) \\
&\quad + A_n^{2^{2^n-1}} \varphi_{2^n-1}(A_n) + \cdots + A_n^2 \varphi_1(A_n) + O + I.
\end{aligned}$$

Since the factor $\varphi_{2^n}(A_n) = A_n^{2^{2^n-1}} + \cdots + A_n$ appears many times in the expression of T_1, we substitute it to the above equation to get the following.

$$\begin{aligned}
T_1 &= (A_n^{2^{2^{n+1}-1}} + \cdots + A_n^{2^{2^{n+1}-2^n}}) \varphi_{2^n}(A_n) \\
&\quad + \{(A_n^{2^{2^n-1}} \varphi_{2^n-1}(A_n))^{2^{2^n}} + \cdots + (A_n^2 \varphi_1(A_n))^{2^{2^n}}\} \\
&\quad + A_n^{2^{2^n-1}} \varphi_{2^n-1}(A_n) + \cdots + A_n^2 \varphi_1(A_n) + I \\
&= I^{2^{2^n}} I + I \\
&\quad + A_n^{2^{2^n-1}} \varphi_{2^n-1}(A_n) \left((A_n^{2^{2^n-1}} \varphi_{2^n-1}(A_n))^{2^{2^n}-1} + I \right) \\
&\quad + \cdots + A_n^2 \varphi_1(A_n) \left((A_n^2 \varphi_1(A_n))^{2^{2^n}-1} + I \right).
\end{aligned}$$

Since $(A_n)^{(2^{2^n}-1)/3} = I$, the last row above equals O.

For T_2 (resp. T_4), one can get the result by the same argument as for T_3 (resp. T_1). $\square$

Lemma 3. *If* $(A_n)^{\frac{2^{2^n}-1}{3}} = I$ *is satisfied, then we have* ${A_{n+1}}^{2^{2^n}+1} = \begin{pmatrix} A_n^3 & O \\ O & A_n^3 \end{pmatrix}$ *and* ${B_{n+1}}^{2^{2^n}+1} = \begin{pmatrix} A_n & O \\ O & A_n \end{pmatrix}$.

Proof. By the assumption $(A_n)^{\frac{2^{2^n}-1}{3}} = I$, it holds $A_n^{2^{2^n}} = A_n$. Then using Lemma 1 and Lemma 2, we have

$${A_{n+1}}^{2^{2^n}} = \begin{pmatrix} A_n^{2^{2^n}} \varphi_{2^n}(A_n) & A_n^{2^{2^n}} \\ A_n^{2^{2^n}+1} & A_n^{2^{2^n}} (\varphi_{2^n}(A_n) + I) \end{pmatrix} = \begin{pmatrix} A_n & A_n \\ A_n^2 & O \end{pmatrix}.$$

Thus

$${A_{n+1}}^{2^{2^n}+1} = \begin{pmatrix} A_n & A_n \\ A_n^2 & O \end{pmatrix} \begin{pmatrix} O & A_n \\ A_n^2 & A_n \end{pmatrix} = \begin{pmatrix} A_n^3 & O \\ O & A_n^3 \end{pmatrix}.$$

Similarly,

$$ {B_{n+1}}^{2^{2^n}} = \begin{pmatrix} \varphi_{2^n}(A_n) + I & I \\ A_n & \varphi_{2^n}(A_n) \end{pmatrix} = \begin{pmatrix} O & I \\ A_n & I \end{pmatrix}. $$

Thus ${B_{n+1}}^{2^{2^n}+1} = \begin{pmatrix} O & I \\ A_n & I \end{pmatrix} \begin{pmatrix} I & I \\ A_n & O \end{pmatrix} = \begin{pmatrix} A_n & O \\ O & A_n \end{pmatrix}$. □

Theorem 2. *For $n \geq 2$, we have*

$$ A_n^{\frac{2^{2^n}-1}{3}} = B_n^{\frac{2^{2^n}-1}{3}} = I. $$

Proof. We prove by induction. For $n = 2$, it is clear that $o(A_2) = 5$ by direct calculation.

Suppose the assertion holds for A_n, that is, $A_n^{\frac{2^{2^n}-1}{3}} = I$ holds.

Then by Lemma 3 we have ${A_{n+1}}^{2^{2^n}+1} = \begin{pmatrix} A_n^3 & O \\ O & A_n^3 \end{pmatrix}$, and by using the induction hypothesis again, we have

$$ {A_{n+1}}^{(2^{2^n}+1)\frac{2^{2^n}-1}{3}} = {A_{n+1}}^{\frac{2^{2^{n+1}}-1}{3}} = I. $$

Using Lemma 3 again, we have the same assertion for the matrix B_n. □

Remark 1. *It is clear that $2^{2^n} - 1 = F_{n-1} \cdot F_{n-2} \cdot \dots \cdot F_1 \cdot F_0$, where $F_n = 2^{2^n} + 1$ is the n-th Fermat number. Therefore we have*

$$ \frac{2^{2^n} - 1}{3} = F_{n-1} \cdot F_{n-2} \cdot \dots \cdot F_1. $$

From Theorem 2, we can write $o(A_n) = s_{n-1} \cdot t_{n-1}$, where s_{n-1} is a factor of F_{n-1} and t_{n-1} is a factor of $\frac{2^{2^{n-1}}-1}{3}$. One can show that $A_n^{F_{n-1}} \neq I$ and $A_n^{\frac{2^{2^{n-1}}-1}{3}} \neq I$ holds by Lemmas above, thus we have $s_{n-1} \neq 1$ and $t_{n-1} \neq 1$.

Furthermore, we have $o((A_{n+1})^{F_n}) = o(A_n^3) = o(A_n)$ by Lemma 3 because 3 is relatively prime to $\frac{2^{2^n}-1}{3}$. Let us write $F_n = s_n \cdot u_n$. Since u_n is relatively prime to $\frac{2^{2^{n+1}}-1}{3}$ and s_n is a factor of the order of A_{n+1}, we have $o(A_{n+1})/s_n = o(A_n)$. Thus we have the following theorem by induction.

Theorem 3. *The order of the matrix A_n is given as follows:*

$$ o(A_n) = s_{n-1} \cdot s_{n-2} \cdot \dots \cdot s_1. $$

Here, s_i is a nontrivial factor of the i-th Fermat number $F_i = 2^{2^i} + 1$.

By the same argument above, we know the order of the matrix B_{n+1} is of the form $s'_n \times t'_n$ with $s'_n \neq 1$ and $t'_n \neq 1$. Then we have $o(B_{n+1})/s'_n = o(A_n)$ by Lemma 3, and have the following theorem for the matrix B_n.

Theorem 4. *The order of the matrix B_n is given as follows:*

$$o(B_n) = s'_{n-1} \cdot o(A_n) = s'_{n-1} \cdot s_{n-2} \cdot \cdots \cdot s_1,$$

where s'_{n-1} is a nontrivial factor of the n-th Fermat number $F_n = 2^{2^{n-1}} + 1$ and s_l's are same as in Theorem 3.

By the work of Lucas (see for example, [1] Theorem 1.3.5), every nontrivial factor of F_n must have the form $k \cdot 2^{n+2} + 1$ with $k \geq 3$, thus we have the evaluation of the order from the below.

Corollary 1. *$o(A_n)$ and $o(B_n)$ are bounded below by $3^{n-1} \cdot 2^{\frac{1}{2}(n+1)(n+2)-3}$.*

By various calculations and the known facts on Fermat numbers, we expect all s_l's and s'_l's are equal to F_l's for every l.

Conjecture 1. *The orders of the matrices A_n and B_n both are equal to*

$$\frac{2^{2^n} - 1}{3} = F_{n-1} \cdot F_{n-2} \cdot \cdots \cdot F_1.$$

5.2. New pseudorandom number generator AST

From now on, we propose a new pseudorandom number generator AST using the matrix B_r defined above and give a linear recurrence equation of AST.

Definition 7. Let W be a w-dimensional vector space over $\mathbb{F}_2$ which is the state space of the generator. Let n, w and r be positive integers with $n \geq 2$ and $r \geq 2$ so that $nw := 2^r$. Define a linear state map $g : W^n \to W^{\frac{n}{2}}$ as below.

Put $x_{n-1}, \ldots, x_1, x_0 \in W$ which we regard as an initial nw-array. We define the linear recurrence by

$$\begin{aligned} (x_{j+\frac{3}{2}n-1}, \ldots, x_{j+n}) &:= g(x_{j+n-1}, \ldots, x_j) \\ &:= (x_{j+\frac{1}{2}n-1}, \ldots, x_j) \times A_{r-1} \\ &\quad + (x_{j+n-1}, \ldots, x_{j+\frac{1}{2}n}) \quad (j = 0, 1, \ldots), \end{aligned} \tag{5.1}$$

where A_{r-1} is the matrix defined in Definition 6.

Put $S := W^n$, then the equation (5.1) can be transformed into the first order linear recurrence from S to S:

$$\begin{aligned} f(x_{j+n-1}, \dots, x_j) &= (g(x_{j+n-1}, \dots, x_j), x_{j+n-1}, \dots, x_{j+\frac{1}{2}n}) \\ &= (x_{j+\frac{3}{2}n-1}, \dots, x_{j+n}, x_{j+n-1}, \dots, x_{j+\frac{1}{2}n}) \qquad (j = 0, 1, \dots). \end{aligned} \tag{5.2}$$

We call this pseudorandom number generator the *Artin-Schreier Tower (AST)*.

This f is a linear state transition map. Since $2^r = nw$, the linear recurrence equation (5.2) is same as multiplying an nw-bit vector by B_r, where B_r is is already defined in Definition 6 as

$$B_r = \begin{pmatrix} I_{r-1} & I_{r-1} \\ A_{r-1} & O \end{pmatrix}.$$

Thus, for nonnegative integer j, we have

$$\begin{aligned} &(x_{j+\frac{3}{2}n-1}, \dots, x_{j+n}, x_{j+n-1}, \dots, x_{j+\frac{1}{2}n}) \\ &\qquad = (x_{j+n-1}, \dots, x_{j+\frac{1}{2}n}, x_{j+\frac{1}{2}n-1}, \dots, x_j) \times B_r. \end{aligned}$$

Start with initial seeds $x_{n-1}, \dots, x_1, x_0$ the state transition is given as follows:

$$\begin{gathered} (x_{n-1}, \cdots, x_{\frac{n}{2}}, x_{\frac{n}{2}-1}, \cdots, x_0) \times B_r \\ \downarrow \\ (x_{\frac{3}{2}n-1}, \cdots, x_n, x_{n-1}, \cdots, x_{\frac{1}{2}n}) \times B_r \\ \downarrow \\ \vdots \\ (x_{\frac{1}{2}n-1}, \cdots, x_0, \cdots\cdots\cdots\cdots) \times B_r \\ \downarrow \\ (x_{n-1}, \cdots, x_{\frac{1}{2}n}, x_{\frac{1}{2}n-1}, \cdots, x_0). \end{gathered}$$

There are some merits for the new generator AST. One is that AST can generate $n/2$ words by multiplying B_r for each time. And the other is that the period of the sequence is $\frac{n}{2} \times o(B)$ which is conjecturally $n/2 \times (2^{2^r} - 1)/3$.

Example 1. *Let us consider the case $r = 11$. In this case, B_r is a $2^{11} \times 2^{11}$ matrix and $nw = 2^{11}$ holds. Take the parameter $w = 2^5 = 32$, for example, then $n = 2^6 = 64$. Let $x_{63}, \dots, x_{32}, x_{31}, \dots, x_0$ be initial seeds. The transformation $f : \mathbb{F}_{2^{2^{11}}} \to \mathbb{F}_{2^{2^{11}}}$ produces a pseudorandom number sequence starting* *with the initial seeds* $x_{63}, \ldots, x_{32}, x_{31}, \ldots, x_0$. *More concretely, the* $2^{11} \times 2^{11}$ *matrix* B_{11} *is as follows:*

$$B_{11} = \begin{pmatrix} I_w & & & I_w & & \\ & \ddots & & & \ddots & \\ & & I_w & & & I_w \\ & A_{10} & & & O & \\ & & & & & \end{pmatrix}.$$

Then the conjectured period of AST with $r = 11$ *is* $\frac{64}{2} \cdot \frac{2^{2^{11}} - 1}{3} \approx 3.447 \times 10^{617}$.

Finally, we mention the generation speed using AST compared to MT19937, whose period is approximately 1.3×10^{6001}. The computational results show that its generation speed is rather slower than TGFSR, which is a demerit of AST. In fact, AST with $r = 11$ needs approximately 465 times longer CPU time than Mersenne Twister MT19937. For AST with $r = 14$ which has conjecturally almost same period with MT19937, it needs approximately 1.2×10^4 times longer CPU time than MT19937. For AST with $r = 16$ which has conjecturally 10^{13729} times longer period than MT19937, it needs approximately 8.8×10^4 times longer CPU time than MT19937.

§6. Results of TestU01

In this section, we exhibit the results of TestU01 [11], which is a C library for empirical testing of pseudorandom number generators by P. L'Ecuyer and R. Simard. We evaluate the performance of our pseudorandom number generator AST using this library.

We implemented AST in C Language and tested it by five batteries in TestU01 Alphabit, Rabbit, Small Crush, Crush and Big Crush in the case of $r = 11$, $w = 32$, $n = 64$, whose conjectured period is approximately 3.447×10^{617}. Here is the table of the results.

Battery	Parameters	# Statistics	# Failures
Alphabit	32×10^9 bits	17	0
Rabbit	32×10^9 bits	40	0
Small Crush	Standard	15	0
Crush	Standard	144	0
Big Crush	Standard	160	2

Through these 376 statistical tests in the five batteries, only two tests called `LinearComp` with different parameters failed in the battery Big Crush and all other tests were passed. `LinearComp` measures the $\mathbb{F}_2$-linear dependency of the given sequence, and it is quite natural that `LinearComp` of AST fails since AST is obviously linearly generated. The p-values of the two tests `LinearComp` were $1 - \texttt{eps1}$, where `eps1` means a value less than 1.0×10^{-15}.

Therefore, our new pseudorandom number generator AST has a good statistical property.

Acknowledgements We would like to express our sincere gratitudes to a number of people who gave useful advice as well as encouragements. Those include Professors Makoto Matsumoto and Hiroshi Haramoto. Thanks are also due to the referee for many valuable comments and suggestions. Research of the second author was partially supported by Grant-in-Aid for Scientific Research, Kiban (C) 20540044, Ministry of Education, Science and Culture. Research of the third author was partially supported by Grant-in-Aid for Young Scientist (B) 22740017, Ministry of Education, Science and Culture.

References

[1] Crandall, R. and Pomerance, C. Prime Numbers, A Computational Perspective, Second Edition, Springer-Verlag, 2005.

[2] Gentle, J. E. Random Number Generation and Monte Carlo Methods, Second Edition, Springer-Verlag, 2005.

[3] Ito, H. and Kajiwara, T. and Song, H. A Tower of Artin-Schreier extensions of finite fields and its applications, to appear in JP J. of Algebra, Number Theory and Applications.

[4] Knuth, D. E. The Art of Computer Programming, Volume 2 Seminumerical algorithms, Third Edition, Addison-Wesley, 1997.

[5] Lewis, T. G. and Payne, W. H. Generalized feedback shift register pseudorandom number algorithms, J. ACM 20, 3(July 1973), 456-468.

[6] Lidl, H. and Neiderreiter, H. Finite fields, Second Edition, Cambridge University Press, 1997.

[7] Matsumoto, M. and Kurita, Y. Twisted GFSR Generators, ACM Trans. on Modeling and Computer Simulation 2 (1992), 179-194.

[8] Matsumoto, M. and Kurita, Y. Twisted GFSR Generators II, ACM Trans. on Modeling and Computer Simulation 4 (1994), 254-266.

[9] Matsumoto, M. and Nishimura, T. Mersenne Twister: a 623-dimensionally equidistributed uniform pseudo-random number generator, ACM Trans. on Modeling and Computer Simulation 8 (1998), 3-30.

[10] Song, H. and Ito, H. On the construction of huge finite fields. AC2009 Proceedings (2009), 1-7.
http://tnt.math.se.tmu.ac.jp/ac/2009/proceedings/ac2009-proceedings.pdf

[11] P. L'Ecuyer and R. Simard. TestU01: A C Library for Empirical Testing of Random Number Generators ACM Trans. on Mathematical Software, Vol. 33, article 22, 2007.

Huiling Song
Department of Applied Mathematics, Graduate School of Engineering
Hiroshima University
Kagamiyama 1-4-1, Higashi-Hiroshima, 739–8527, Japan
and
Department of Mathematics, Faculty of Foundation
Harbin Finance University
65 Diantan Road, Xiangfang, Harbin, Heilongjiang, 150030, China
E-mail: huiling@amath.hiroshima-u.ac.jp

Hiroyuki Ito
Department of Mathematics, Faculty of Science and Technology
Tokyo University of Science
Yamazaki 2641, Noda, Chiba, 278–8510, Japan
E-mail: ito_hiroyuki@ma.noda.tus.ac.jp

Yukinori Kitadai
Department of Electronics and Computer Engineering, Faculty of Engineering
Hiroshima Institute of Technology
Miyake 2-1-1, Saeki-ku, Hiroshima, 731–5193, Japan
E-mail: Nyoho@ec.it-hiroshima.ac.jp