Journal of Statistical Software

August 2004, Volume 11, Issue 5. *http://www.jstatsoft.org/*

Note on Marsaglia's Xorshift Random Number Generators

Richard P. Brent
Oxford University

Abstract

Marsaglia (2003) has described a class of "**xorshift**" random number generators (RNGs) with periods $2^n - 1$ for $n = 32, 64$, etc. We show that the sequences generated by these RNGs are identical to the sequences generated by certain linear feedback shift register (LFSR) generators using "exclusive or" (xor) operations on n-bit words, with a recurrence defined by a primitive polynomial of degree n.

Keywords: random number generators, LFSR sequences, linear feedback shift registers, primitive polynomials, Xorshift RNGs.

1. Introduction

Marsaglia (2003) suggests "**xorshift** RNGs" using the "exclusive or" operation on 32-bit or 64-bit words with left- or right-shifted versions of the same word. The generators have period $2^n - 1$ where n is 32 or a small multiple of 32. For example, in the case $n = 64$, the generators have period $2^{64} - 1$ and produce all possible 64-bit words except the word of all zero bits. Note that the same is true for a linear feedback shift register (LFSR) generator (Menezes, van Oorschot, and Vanstone 1997) using a recurrence defined by a primitive polynomial $P(z)$ of degree 64 and operating in parallel on 64-bit words. This suggests that the two RNGs might be related. In fact, as we show in §5, there is a primitive polynomial and starting conditions such that the two generators produce identical sequences of pseudo-random numbers. Thus, Marsaglia's **xorshift** RNGs inherit all the good (and bad) theoretical properties of LFSR generators. They have better statistical properties than LFSR generators based on primitive trinomials of degree n because the number $W(P(z))$ of nonzero terms in $P(z)$ is typically much larger than 3 (see the examples in §4).

From the point of view of a software developer, Marsaglia's idea is useful, because his implementation requires less space than a standard implementation of the corresponding LFSR generator. This is possible because the initial conditions are special. Marsaglia's implementation may also be faster, requiring only about three xor and shift operations (and a comparable number of loads and stores), whereas the standard implementation of an LFSR generator requires $W(P(z)) - 2$ xor operations.

First we introduce some notation and describe LFSR and **xorshift** random number generators, then we show how the LFSR and **xorshift** generators are related.

2. Some Notation and Theory

Let $F_2 = \mathrm{GF}(2)$ be the finite field with two elements $\{0, 1\}$. We write the field operations as $+$ and $\times$. If 0 is regarded as "false" and 1 as "true", then the field operations are "exclusive or" (`xor` or $\oplus$) and "and" ($\wedge$). In the following, vectors and matrices have elements in F_2, and polynomials have coefficients in F_2. For consistency with Marsaglia (2003), we use row rather than column vectors.

If a polynomial $P(z)$ has degree $n > 1$ and the powers $z^k \bmod P(z)$ are distinct for $0 \le k \le 2^n - 2$, then $P(z)$ is *primitive.* If $P(z)$ is primitive then its *reverse* $\widetilde{P}(z) = z^n P(1/z)$ is also primitive. For more background on polynomials over finite fields, see for example Lidl and Niederreiter (1994) or Menezes *et al.* (1997).

Let $A \in F_2^{n\times n}$ be an $n \times n$ matrix over F_2. The *characteristic polynomial* $C(z)$ of A is defined by

$$C(z) = \det(A - zI) \ .$$

The Cayley-Hamilton theorem states that A satisfies its own characteristic polynomial, that is

$$C(A) = 0 \ .$$

The *minimal polynomial* of A is the monic polynomial $P(z)$ of mimimal degree such that $P(A) = 0$. Clearly $P(z)$ divides $C(z)$.

Suppose that A is nonsingular. The *period* of A is the minimal positive integer ρ such that $A^\rho = I$. From the Cayley-Hamilton theorem, any positive power of A can be expressed as a linear combination of $\{I, A, A^2, A^3, \ldots, A^{n-1}\}$, and there are at most $2^n - 1$ nonzero possibilities. Thus, $\rho \le 2^n - 1$. The maximum period $\rho = 2^n - 1$ is attained iff the minimal polynomial $P(z)$ is a primitive polynomial of degree n.

If $v = (v_1, v_2, \ldots, v_n) \in F_2^{1\times n}$ is an n-vector over F_2, then we define the norm $||v||$ to be the *Hamming weight* of v, that is the number of nonzero components of v. Thus, for two vectors u, v, the usual *Hamming distance* is $||u - v||$.

3. LFSR Generators

A *Linear Feedback Shift Register* (LFSR) sequence (Menezes *et al.* 1997, §6.2.1) is a sequence (x_j) satisfying a linear recurrence of the form

$$\sum_{k=0}^{d} \alpha_k x_{j-k} = 0 \ \text{ for } \ j \ge d, \tag{1}$$

where $\alpha_0, \alpha_1, \ldots, \alpha_d \in F_2$ and we assume that $\alpha_0 = 1$. The recurrence defines x_j as a linear combination of $x_{j-1}, \ldots, x_{j-d}$. If $x_0, x_1, \ldots, x_{d-1}$ are given as *initial conditions*, then all x_j for $j \ge d$ are uniquely defined by the recurrence.

In hardware implementations of LFSR sequences, the x_j are usually single bits (elements of F_2), but in software implementations it is easy and more efficient to operate on whole words. In the literature (Marsaglia 2003; Menezes *et al.* 1997), the term “LFSR generator” or “shift register generator” is used to describe random number generators that operate either on single bits or on words. Thus, we assume that the x_j can be scalars or vectors of any fixed size (the recurrence applies independently to each component of the vectors).

The *connection polynomial* $P(z)$ corresponding to the recurrence (1) is the polynomial

$$P(z) = \sum_{k=0}^{d} \alpha_k z^k ,$$

and by standard techniques (Knuth 1997, §1.2.9) the *generating function*

$$G(z) = \sum_{m=0}^{\infty} x_m z^m ,$$

regarded as a formal power series, is given by

$$G(x) = P_0(z)/P(z) .$$

Here $P_0(z)$ is a polynomial (or vector of polynomials) of degree at most $d-1$, depending on the initial conditions. If $P(z)$ is primitive of degree d and $P_0(z) \neq 0$, then the sequence (x_j) is periodic with period $2^d - 1$.

4. Marsaglia’s Xorshift Generators

Let $\beta \in F_2^{1\times n}$ be a nonzero row-vector whose components are in F_2. If we are using a computer with word-length n bits, then we can regard β as a computer word. In the following, β is the *seed* for one of Marsaglia’s **xorshift** RNGs.

Let $T \in F_2^{n\times n}$ be any nonsingular $n \times n$ matrix over F_2. A pseudo-random sequence of n-bit vectors $(x_j)_{j\geq 0}$ can be defined by

$$x_j = \beta T^j \tag{2}$$

and computed using the recurrence $x_0 = \beta$, $x_j = x_{j-1}T$ for $j \geq 1$. With a suitable choice of T, we get Marsaglia’s 32-bit and 64-bit generators. If $n > 64$ then Marsaglia’s generators return only 32 or 64 bits of x_j to the user, but the mathematical theory is similar, so for simplicity we assume that $n \leq 64$.

Marsaglia’s idea is to take T of the form[1]

$$T = (I + L^a)(I + R^b)(I + L^c) , \tag{3}$$

where

$$L = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 \end{pmatrix}$$

[1] There is a typo in Marsaglia (2003, line 15 of §3), where $(I + L^a)(I + R^b)(I + R^c)$ should be $(I + L^a)(I + R^b)(I + L^c)$.

is the "left shift" matrix such that

$$(v_1v_2\dots v_{n-1}v_n)L = (v_2v_3\dots v_n0),$$

$R = L^T$ is the "right shift" matrix such that

$$(v_1v_2\dots v_{n-1}v_n)R = (0v_1\dots v_{n-2}v_{n-1}),$$

and (a, b, c) is a suitable triple of positive integers.

Marsaglia considers T acceptable if its period is the maximum possible, that is $\rho = 2^n - 1$. In other words, $T^\rho = I$ but $T^j \neq I$ for $0 < j < \rho = 2^n - 1$. From §2, this occurs if the minimal polynomial of T has degree n and is primitive.

For example, if $n = 32$ we can take $(a, b, c) = (1, 3, 10)$, and the minimal polynomial is

$$x^{32} + x^{29} + x^{28} + x^{27} + x^{21} + x^{19} + x^{18} + x^{16} + x^{12} + x^{11} + x^{10} + x^9 + x^6 + x^5 + 1.$$

If $n = 64$ we can take $(a, b, c) = (1, 1, 54)$, and the minimal polynomial is

$$x^{64} + x^{63} + x^{62} + x^{60} + x^{56} + x^{48} + x^{32} + x^9 + x^5 + x + 1.$$

For many other possible triples, see Marsaglia (2003, §3).

We note a small error in Marsaglia (2003, §3). He considers the simpler candidate

$$T = (I + L^a)(I + R^b)\ , \tag{4}$$

and writes "when n is 32 or 64, no choices for a and b will provide such a T with the required order". This is true for $n = 32$, but when $n = 64$ we can take $(a, b) = (7, 9)$ to get T with order $2^{64} - 1$. In fact T has minimal polynomial

$$P(z) = z^{64} + z^{49} + z^{40} + z^{33} + z^{19} + z^{18} + z^{16} + z^{14} + z^{11} + z^{10} + z^6 + x + 1$$

and $P(z)$ is primitive. The choice (4) of T gives a generator that is slightly faster than the choice (3). We do not necessarily recommend the choice (4) for a high-quality random number generator, because $T = (I + L^a)(I + R^b)$ is very sparse and hence maps vectors with low Hamming weight to other vectors with low Hamming weight, in fact $||xT|| \le 4||x||$. For a matrix T satisfying (3) the corresponding inequality is $||xT|| \le 8||x||$.

5. Xorshift and LFSR Generators

Suppose that (x_j) is any sequence of n-vectors satisfying (2). As we have seen in §4, Marsaglia's **xorshift** generators give such a sequence if β is the seed and T is chosen suitably. Let $P(z) = \sum_{k=0}^{d} \alpha_k z^{d-k}$ be the minimal polynomial of T. We can assume that $P(z)$ is monic of degree $d \le n$, so $\alpha_0 = 1$ and

$$\sum_{k=0}^{d} \alpha_k T^{d-k} = 0\ .$$

Thus, multiplying on the left by βT^{j-d}, we have

$$\sum_{k=0}^{d} \alpha_k \beta T^{j-k} = 0 \quad \text{for all } \ j \ge d.$$

Since $x_j = \beta T^j$, it follows that

$$\sum_{k=0}^{d} \alpha_k x_{j-k} = 0 \quad \text{for all } \ j \geq d.$$

This is just the linear recurrence (1) considered in §3. Thus, we see that the sequence can be generated by a LFSR whose connection polynomial is $\widetilde{P}(z) = \sum_{k=0}^{d} \alpha_k z^k$.

In the case of Marsaglia's **xorshift** generators, the condition that the period is $2^n - 1$ can be satisfied iff $d = n$ and $P(z)$ is primitive.

Acknowledgments

I thank Wolfgang Hartmann for pointing out Marsaglia's paper (Marsaglia 2003). The computations of minimal polynomials and primitivity checks were performed with the aid of Magma (Bosma and Cannon 1997).

References

Bosma WW, Cannon JJ (1997). *Handbook of Magma Functions*. School of Mathematics and Statistics, University of Sydney. URL http://magma.maths.usyd.edu.au/magma/.

Knuth DE (1997). *The Art of Computer Programming*, volume 1. Addison-Wesley, Reading, Massachusetts, 3 edition. ISBN 0-201-89683-4.

Lidl R, Niederreiter H (1994). *Introduction to Finite Fields and their Applications*. Cambridge University Press, Cambridge, 2 edition.

Marsaglia G (2003). "Xorshift RNGs." *Journal of Statistical Software*, **8**, 1–9. URL http://www.jstatsoft.org/v08/i14/.

Menezes AJ, van Oorschot PC, Vanstone SA (1997). *Handbook of Applied Cryptography*. CRC Press, New York. URL http://cacr.math.uwaterloo.ca/hac/.

Affiliation:

Richard P. Brent
Computing Laboratory
Oxford University, UK
E-mail: rng@rpbrent.co.uk

Journal of Statistical Software — *Submitted:* 2004-08-25
August 2004, Volume 11, Issue 5. — *Accepted:* 2004-07-07
http://www.jstatsoft.org/