AN ALGORITHM FOR GENERATING BINARY PSEUDO-RANDOM SEQUENCES

Wiktor Dańko

Faculty of Computer Science, Bialystok University of Technology, Białystok, Poland

Abstract: In the paper it is presented an algorithm for generating pseudo-random binary sequences. There are formulated theorems concerning properties of the sequence generated by the algorithm. The sequence is not periodic. Moreover, for any natural number $n > 0$, the initial fragment of the generated sequence of the length $(2 \cdot n) \cdot 2^{(2 \cdot n)}$ contains all (binary) series of the length n.

Keywords: pseudo-random computer generators, computer simulations, probabilistic algorithms

1. Introduction

Computer modeling of real stochastic processes is mainly based on a pseudo-random computer generator.

In the present paper we shall be concerned with uniform binary pseudo-random generators, i.e., with the set $\{0,1\}$ of values.

We can restrict ourselves to the case of uniform binary pseudo-random generators because by means of such a binary generator, we can construct any uniform generator with the set of values of the form $\{0,1,...,m\}$, for arbitrary integer $m > 1$ (cf. [2], [5]). This construction essentially depends on the fact that the results of two successive random assignments

```
x:= random; x:= random;
```

realized by means of a pseudo-random generator, treated as random events, are independent.

Each uniform binary pseudo-random generator will be viewed in terms of an infinite sequence

$$b_0\, b_1\, b_2\, \ldots\, b_n\, \ldots$$

Zeszyty Naukowe Politechniki Białostockiej. Informatyka, vol. 9, pp. 17-29, 2012.

of values b_n from $\{0,1\}$, $n = 0,1,2,...$, produced by the generator in succession, one after the other. We shall be interested in the case, where the sequence

$$\{b_n\}_{n=0,1,2,...}$$

is not periodic.

Typical information on a periodic pseudo-random computer generator concerns the set of obtained values and the period of the generator. Some users of computer generators believe, in an intuitive manner, that the credibility of results of computer simulations seems to essentially depend on the length of the period of the generator. But the dependence is not evident. Let us consider an example of a sequence produced by a generator with the following period sequence:

$$010011000111 \ldots \underbrace{00\ldots0}_{l}\underbrace{11\ldots1}_{l}$$

of the length $p = l \cdot (l+1)$.
Speaking informally, since the frequencies of appearing of "0" and "1" are the same then this generator is uniform. On the other hand, it is easy to see that the sequence produced by generator does not contain any subsequence of the form

$$\underbrace{01\ldots01}_{2k}$$

for $k \geq 2$. This means that this periodic sequence cannot be used as a straightforward (pseudo) random sequence, in spite of the fact that its period could be arbitrarily large. Other remarks related periodic pseudo-random generator will be formulated in the next section.

2. Remarks on periodic generators

Let us consider a fixed periodic binary pseudo-random generator producing a sequence

$$\{b_n\}_{n=0,1,2,...} .$$

We recall that this generator will be called periodic (cf. [13], [14]) if and only if there exist numbers q and $p > 0$ such that for $i \geq q$, $b_i = b_{i+j \cdot p}$ for $j = 0,1,2,....$ The sequence $b_q\, b_{q+1} \ldots b_{q+p-1}$ is called the period sequence of the sequence $\{b_n\}_{n=0,1,2,...}$.

We shall now introduce some definitions and notation.

By a series we shall understand any finite sequence c of the form

$$c_0\, c_1\, \ldots\, c_{k-1}$$

where $c_i \in \{0,1\}$ for $i = 0,1,...,k-1$. The number k will be called the length of the series.

We shall say that a pseudo-random generator, producing a sequence $\{b_n\}_{n=0,1,2,...}$, realizes the series $c_0\, c_1\, \ldots\, c_{k-1}$ if and only if there exists an index m, $m \geq 0$, such that the sequence $b_m\, b_{m+1}\, \ldots\, b_{m+k-1}$ and the sequence $c_0\, c_1\, \ldots\, c_{k-1}$ are identical. In the case, where such an index m does not exist, we shall say that the generator omits the series $c_0\, c_1\, \ldots\, c_{k-1}$.

The following remark shows a shortcoming of periodic pseudo-random generators.

Remark 1.
Any periodic pseudo-random generator omits some series.

□

To argue this fact let us suppose that

$$b_q\, b_{q+1}\, \ldots\, b_{q+p-1}$$

is the period sequence of a generator. It is easy to observe that the generator omits the sequence

$$b_q\, b_{q+1}\, \ldots\, b_{q+p-1}\, (1-b_q)$$

(its length is length $p+1$). In reality, the minimal length of omitted series is essentially less than that period of the generator (cf. [5]).

Using the above remark we shall show that the use of periodic generators may lead to important differences between results of computer simulations and theoretically determined facts.

Let us consider a fixed binary pseudo-random generator producing a sequence $\{b_n\}_{n=0,1,2,...}$ and let $c_0\, c_1\, \ldots\, c_{k-1}$ be a series omitted by the generator.

Let `P` denote the following program below, where the initial values of the variables $\mathtt{C}_0$, $\mathtt{C}_1$, ..., $\mathtt{C}_{k-1}$ are $c_0\, c_1\, \ldots\, c_{k-1}$, respectively, and the realization of the instructions `x:=random{0,1}` consists in assigning to the variable `x` the values produced by the considered generator in succession, one after the other.

```
begin
     t:= 0;
     while (t = 0) do
        begin
           t:= 1;
           x:= random{0,1};
           if ((t=1)&(x=C0)) then t:= 1 else t:= 0;
           x:= random{0,1};
           if ((t=1)&(x=C1)) then t:= 1 else t:= 0;

                              .  .  .

           x:= random{0,1};
           if ((t=1)&(xCk-1)) then t:= 1 else t:= 0;
        end;
end;
```

It is easy to observe (cf. [3], [5]) that in the case, where the generator realizes the series $c_0\ c_1\ ...\ c_{k-1}$, the program P ends its computation and in the case, where the generator omits the series $c_0\ c_1\ ...\ c_{k-1}$ this program does not end its computation. This means that the results of computer simulations based on periodic pseudo-random generators may essentially differ from the result of computer simulations using straightforward random generators.

3. Grey codes

We start with the definition of Grey codes (cf. [1], [11]). First, we give examples of the sequences $C^{(l)} =< c_i^l,\ i = 0,1,...,2^l-1 >$ of Grey codes of the length $l = 1,2,3$.

$l = 1,\ C^{(1)} =< c_0^1, c_1^1 >=< 0,1 >,$
$l = 2,\ C^{(2)} =< c_0^2, c_1^2, c_2^2, c_3^2 >=< 00,10,11,01 >,$
$l = 3,\ C^{(3)} =< c_0^3, c_1^3, ..., c_7^3 >=< 000,100,110,010,011,111,101,001 > .$

The induction step in defining the sequence $C^{(l+1)}$, provided that the sequence $C^{(l)}$ is defined, is the following:

for $C^{(l)} =< c_0^{(l)}, c_1^{(l)}, ..., c_{2^l-2}^{(l)}, c_{2^l-1}^{(l)} >$
we define $C^{(l+1)} =< c_0^{(l)}0, c_1^{(l)}0, ..., c_{2^l-1}^{(l)}0, c_{2^l-1}^{(l)}1, ..., c_1^{(l)}1, c_0^{(l)}1 > .$

Let us denote by $g(l,i,j)$ the j-th position of the i-th sequence $c_i^{(l)}$ of the sequence $C^{(l)}$, $i = 0,1,...,2^l-1$, $j = 0,1,...,l-1$. It is known that the binary values $g(l,i,j)$ can be defined inductively as follows (cf. [1], [11]):

$$g(1,0,0) = 0,\ g(1,1,0) = 1,$$
$$g(l+1,i,j) = \begin{cases} 0 & \text{for } i = 0,1,...,2^l-1 \text{ and } j = l+1 \\ g(l,i,j) & \text{for } i = 0,1,...,2^l-1 \text{ and } j = 0,1,...,l-1 \\ 1 & \text{for } i = 2^l,2^l+1,...,2^{l+1}-1 \text{ and } j = l+1 \\ g(l,2^{l+1}-1-i,j) & \text{for } i = 2^l,2^l+1,...,2^{l+1}-1 \\ & \quad \text{and } j = 0,1,...,l-1. \end{cases}$$

We shall now investigate whether the "infinite concatenation"

$$C^{(1)} \circ C^{(2)} \circ ... \circ C^{(l)} \circ ... .$$

of the sequences $C^{(l)}$, $l = 1,2,3,...$, could play the role of an uniform binary pseudo-random generator. Denote by

$$\{r_n\}_{n=0,1,2,...}$$

the sequence corresponding to the "infinite concatenation" of the sequences $C^{(l)}$, $l = 1,2,3,....$ Let us first note that the length $p(l)$ of the sequence

$$C^{(1)} \circ C^{(2)} \circ ... \circ C^{(l)}.$$

satisfies the inductive equation

$$p(0) = 0,\ p(1) = 2,$$
$$p(l+1) = p(l) + ((l+1) \cdot 2^{l+1}).$$

Let n be an arbitrary natural number. Denote by l the natural number such that

$$p(l) \leq n < p(l+1)$$

and let

$$i = (n - p(l))\ div\ (2^l) \text{ and } j = (n - p(l))\ mod\ (2^l).$$

The sequence $\{r_n\}_{n=0,1,2,...}$ corresponding to the "infinite concatenation" of the sequences $C^{(l)}$ can be defined in the following way:

$$r_n = g(l,i,j)$$

where $l, g(l,i,j), p(l), i, j$ have the meaning as in the above. The following algorithm `R(n)` computes the elements of the sequence $\{r_n\}_{n=0,1,2,...}$. We shall assume that the program functions `P(l)`, `G(l,i,j)`, used in `R(n)`, have been previously programmed and compute the function $p(l)$, $g(l,i,j)$ defined above.

```
function R(n);
    begin
       if (n=0) then return(0)
       else if (n=1) then return(1)
            else
               begin
                  l:= 1;
                  while (P(l+1) <= n) do l:= l+1;
                  i:= (n-P(l))div(2^l);
                  j:= (n-P(l))mod(2^l);
                  return(G(l,i,j));
               end;
    end;
```

Remark 2.
The complexity of the algorithm `R(n)` *is* $\mathrm{O}(n)$.

□

To argue this remark it is sufficient to note that the complexity of computation of the functions $p(l)$ and $g(l,i,j)$ is of the order $\mathrm{O}(l)$.

Now, we recall the well known property o Grey codes of the length l (cf. [1], [11]):

> *Let* $C^{(l)} =< c_0^{(l)}, c_1^{(l)}, ..., c_{2^l-2}^{(l)}, c_{2^l-1}^{(l)} >$ *be the sequence of Grey codes of the length* l. *For each* $j = 1,2,...,2^l-1$, *the codes* $c_{j-1}^{(l)}$ *and* $c_j^{(l)}$ *differ on only one position.*

This property causes that the sequence $\{r_n\}_{n=0,1,2,...}$ cannot be used as a pseudo-random generator.

Remark 3.
The algorithm `R(n)` *is of any use in pseudo-random generating binary sequences.*

It can be used in testing whether a binary pseudo-random generator realizes or omits particular binary series. Moreover, since `R(n)` *produces all finite binary sequences, it can be used in algorithm testing.*

□

The main motive for the above detailed presentation of the algorithm `R(n)` is the fact, that the idea of the construction of the binary pseudo-random generator, presented in Section 5, is analogous, in a way, to the construction of `R(n)`. Namely, similarly to the case of `R(n)`, the result sequence of the proposed generator can be

viewed as an "infinite" concatenation

$$S^{(1)} \circ S^{(2)} \circ S^{(3)} \circ \ldots \circ S^{(l)} \circ \ldots .$$

of sequences, where the length of $S^{(i)}$ is equal to 8, for $i = 1$ and $(i \cdot 2^i - \frac{1}{2} \cdot i \cdot 2^{l-1})$ for $i > 1$. We shall formulate (cf. Theorem 1, Section 5) the fact, that the initial fragment

$$S^{(1)} \circ S^{(2)} \circ S^{(3)} \circ \ldots \circ S^{(l)} \circ \ldots .$$

of the generated sequence, being of the length $l \cdot 2^l$, realizes all series of the length $\frac{1}{2} \cdot l$.

To describe precisely the algorithm we need to introduce a notation and formulate some auxiliary mathematical facts. This will be done in the next section.

4. Mathematical preliminaries

The main fact formulated in this section is a lemma concerning a manner of natural number representation.

To formulate this lemma we need a notation. By d we shall denote natural numbers of the form 2^u, where $u = 0, 1, 2, .$ W shall now define:

$$\begin{aligned} l(d) &= 2^d, \\ w(d) &= l(d) \cdot d, \\ s(d) &= 2 \cdot l(d) \cdot l(d) \cdot d, \end{aligned}$$

For example, we have:

$$\begin{array}{llll} d = 1, & l(1) = 2, & w(1) = 2, & s(1) = 8, \\ d = 2, & l(2) = 4, & w(2) = 8, & s(2) = 64, \\ d = 4, & l(4) = 16, & w(4) = 64, & s(4) = 2048, \\ d = 8, & l(8) = 256, & w(8) = 2048, & s(8) = 1048576. \end{array}$$

Using the above notation we shall formulate the main.

Lemma
Let n be an arbitrary natural number, greater than 7 and let d be the number of the form 2^u, where $u = 1, 2, \ldots$, such that

$$s(d/2) \leq n < s(d).$$

Then there exist natural numbers i, j, k, such that

$$n = i \cdot (2 \cdot w(d)) + j \cdot d + k$$

and such that

$$0 \leq i < l(d),\ 0 \leq j < 2 \cdot l(d),\ 0 \leq k < d.$$

The numbers i, j, k, are determined univocally.

□

The inductive proof of the lemma is rather technical and is omitted. We end this section with two examples illustrating the lemma.

Example 1.

$n = 47$.

For $d = 2$ we have:

$8 = s(1) \leq 47 < s(2) = 64$,

$2 \cdot w(2) = 2 \cdot 8 = 16$.

For the values $i = 2, j = 7, k = 1$, satisfying

$0 \leq i = 2 < l(d) = 4,\ 0 \leq j = 7 < 2 \cdot l(d) = 8,\ 0 \leq k = 1 < d = 2$,

we have

$$47 = i \cdot (2 \cdot w(d)) + j \cdot d + k = i \cdot 16 + j \cdot 2 + k = 2 \cdot 16 + 7 \cdot 2 + 1.$$

Example 2.

$n = 1831$.

For $d = 4$ we have:

$64 = s(2) \leq 1831 < s(4) = 2048$,

$2 \cdot w(4) = 2 \cdot 64 = 128$.

For the values $i = 14, j = 9, k = 3$, satisfying

$0 \leq i = 14 < l(d) = 16,\ 0 \leq j = 9 < 2l(d) = 32,\ 0 \leq k = 3 < d = 4$,

we have

$$1831 = i \cdot (2 \cdot w(d)) + j \cdot d + k = i \cdot 128 + j \cdot 4 + k = 14 \cdot 128 + 9 \cdot 4 + 3.$$

5. The algorithm

Using the notation introduced in the preceding section we can describe the proposed algorithm generating a binary pseudo-random sequence. For readability of inductive equations, describing the sequence generated by the algorithm, elements of the sequence will be denoted by

$$\{B(n)\}_{n=0,1,2,\ldots}.$$

The recursive definition of the sequence is as follows:

(A) Initial values:
We define the values $B(n)$, for n satisfying $n < s(1) = 8$, i.e.,
for $n = 0,1,2,3,4,5,6,7$:

$$B(0) = 1,\ B(1) = 1,\ B(2) = 0,\ B(3) = 1,$$
$$B(4) = 0,\ B(5) = 0,\ B(6) = 1,\ B(7) = 0.$$

(B) Induction step:
Assume that the elements $B(n)$ are defined for all values n satisfying

$$n < s(d/2).$$

We shall now define the values for n satisfying

$$s(d/2) \leq n < s(d),$$

i.e., for

$$n = i \cdot (2 \cdot w(d)) + j \cdot d + k,$$

where $i = 0,1,...,l(d)-1$, $j = 0,1,...,2 \cdot l(d)-1$, $k = 0,1,...,d-1$.

(B_0) We first define the values $B(n)$, for n satisfying

$$w(d) = s(d/2) \leq n < 2 \cdot w(d) :$$

Namely,

$$B(n) = B(w(d) + j \cdot d + k) = B([(l(d) - j - 1) mod l(d)] \cdot d + k)$$

for $j = 0,1,...,l(d)-1$, $k = 0,1,...,d-1$.
Thus we have defined the values $B(n)$, for $n = i \cdot (2 \cdot w(d)) + j \cdot d + k$, where $i = 0$ and $j = 0,1,...,2l(d)-1$, $k = 0,1,...,d-1$.

(B_1) Now, we define

$$B(n) = \begin{cases} B((i-1) \cdot 2 \cdot w(d) + j \cdot d + k) \\ \qquad \text{if } (i \text{ is even and } j \text{ is even}), \\ B((i-1) \cdot 2 \cdot w(d) + [(j+2) mod (2 \cdot l(d)] \cdot d + k) \\ \qquad \text{if } (i \text{ is even and } j \text{ is odd}), \\ B((i-1) \cdot 2 \cdot w(d) + [(j-2) mod (2 \cdot l(d)] \cdot d + k) \\ \qquad \text{if } (i \text{ is odd and } j \text{ is even}), \\ B((i-1) \cdot 2 \cdot w(d) + j \cdot d + k) \\ \qquad \text{if } (i \text{ is odd and } j \text{ is odd}) \end{cases}$$

for

$$n = i \cdot (2 \cdot w(d)) + j \cdot d + k,$$

where

$$\begin{aligned} &i = 1, 2, ..., l(d) - 1, \\ &j = 0, 1, ..., 2 \cdot l(d) - 1, \\ &k = 0, 1, ..., d - 1. \end{aligned}$$

This ends the induction step.

The above inductive definition of the sequence $\{B(n)\}_{n=0,1,2,...}$ is not convenient for computer implementation. However, it enables us to formulate theorems concerning properties of the sequence.

6. Properties of the sequence generated by the algorithm

In this section we formulate (without proofs) several facts about the sequence $\{B(n)\}_{n=0,1,2,...}$ that concern its probabilistic properties.

Theorem 1.
For any natural number $n > 0$, the initial fragment of the generated sequence of the length $(2 \cdot n) \cdot 2^{(2 \cdot n)}$ contains all (binary) series of the length n.

□

Corollary 1.
The sequence $\{B(n)\}_{n=0,1,2,...}$ is not periodic.

□

Let a denote a fixed series $a_0 a_1 ... a_{d-1}$ of the length d being of the form $d = 2^u$, where $u = 0, 1, 2, ...$.

By $N_a(n)$ we shall denote the number of occurrences of the series a in the initial fragment $B(0)B(1)...B(n)$.

Let us also define

$$F_a(n) = \frac{N_a(n)}{n}.$$

The intuitive meaning of $F_a(n)$ is the frequency of appearing of the series a in the initial fragment $B(0)B(1)...B(n)$.

Theorem 2.
Let a and b denote fixed series of the same length d.
Then

$$\lim_{n\to\infty} \frac{N_a(n)}{N_b(n)} = 1$$

□

Let us treat the results of successive execution of two random assignments

```
x:= random; x:= random;
```

realized by means of a pseudo-random generator, as random events.
The following theorem concerns, in a way, the independence of such events.

Theorem 3.
Let a and b denote fixed series of the same length d.
Then

$$\lim_{n\to\infty} n \cdot \frac{N_{a\circ b}(n)}{N_a(n)\cdot N_b(n)} = \lim_{n\to\infty} \frac{F_{a\circ b}(n)}{F_a(n)\cdot F_b(n)} = 1$$

□

The independence of results of random assignments based on a pseudo-random generator is of great importance for modeling real stochastic processes (cf. [3]).

7. Final remarks

The proofs of mathematical facts formulated in the paper are too long for presentation in this publication and therefore are omitted here. They will be presented in separate papers.

The question of nonrecursive implementation of the inductive definition of the sequence $\{B(n)\}_{n=0,1,2,\ldots}$ will be discussed in a separate paper. This paper will also contain an analysis of the complexity of such an implementation.

References

[1] Cormen T., Leiserson C., Riverst R., Stein C., Introduction to Algorithms, Massachusetts Institute of Technology, 2001.

[2] Dańko A., Dańko W., Improving Pseudo-Random Generators, International Conference on Biometrics and Kansei Engineering, 24-28 June Cieszyn, Poland, pp. 163-166, 2009.

[3] Dańko W., The Set of Probabilistic Algorithmic Formulas Valid in a Finite Structure is Decidable with Respect to Its Diagram, Fundamenta Informaticae, vol. 19 (3-4), pp. (417-431), 1993.

[4] Dańko W., Remarks on Computer Simulations, Biometrics, Computer Security Systems & Artificial Intelligence Applications, red.: Khalid Saeed, Jerzy Pejaś, Romuald Mosdorf, Springer, Science+Business Media, LLC, New York, pp. 197-206, 2006.

[5] Dańko W., Algorithmic Models of Probabilistic Processes, Politechnika Białostocka, (in Polish, to appear).

[6] Feller W., An Introduction to Probability Theory and Its Applications, John Wiley and Sons, Inc., New York, London, 1961.

[7] Gentle J.E., Random Number Generation and Monte Carlo Methods, Springer, 2003.

[8] Jessa M., Designing Security for Number Sequences Generated by Means of The Sawtooth Chaotic Map, IEEE Transactions on Circuits and Systems - I: Regular Papers, vol. 53 (5), pp. 1140-1150, 2006.

[9] L'Ecuyer P., Random Numbers for Simulation, Comm. ACM 33 (10), pp. 85-97.

[10] L'Ecuyer P., Tezuka S., Structural Properties for Two Classes of Combined Random Number Generators, Math. Comput. 57, pp. l35-746, 1991.

[11] Lipski W., Combinatorics for Computer Scientists, WNT, Warszawa, 1985, (in Polish).

[12] Zieliński R., Random Numbers Generators, WNT, Warszawa, 1972, (in Polish), Transaction on Circuit and Systems - I:Regular Papers, vol. 53, 5, pp. 1140-1150.

[13] Zieliński R, Wieczorkowski R., Computer Random Numbers Generators, WNT, Warszawa, 1997, (in Polish).

[14] Tezuka S., A Unified View of Large-Period Random Number Generators, J. Oper. Res. Soc. Japan. 37, pp. 211-227.

ALGORYTM GENEROWANIA PSEUDOLOSOWYCH CIĄGÓW BINARNYCH

Streszczenie: W pracy został przedstawiony algorytm generowania pseudo-losowego ciągu binarnego. Sformułowane zostały twierdzenia dotyczące własności otrzymanego ciągu. Nie jest to ciąg okresowy i dla dowolnego $n > 0$, początkowy odcinek ciągu o długości $(2 \cdot n) \cdot 2^{(2 \cdot n)}$ zawiera wszystkie serie binarne o długości n.

Słowa kluczowe: pseudolosowy generator komputerowy, symulacja komputerowa, algorytm probabilistyczny

Artykuł zrealizowano w ramach pracy badawczej S/WI/1/2011.