TECHNION - Israel Institute of Technology
Computer Science Department

ALGORITHMS FOR THE GENERATION OF
FULL-LENGTH SHIFT-REGISTER SEQUENCES

by
Tyvi Etzion and Abraham Lempel
Technical Report #245

June 1982
ABSTRACT

This paper presents two algorithms for the generation of full-length, shift-register cycles, also referred to as de Bruijn sequences. The first algorithm generates \(2^k \cdot g(n,k)\) full cycles of length \(2^n\), using \(3n + k \cdot g(n,k)\) bits of storage, where \(k\) is a free parameter in the range \(1 \leq k \leq 2\), and \(g(n,k)\) is of the order of \(n - 2\log k\). The second algorithm generates about \(2^{\frac{n^2}{4}}\) full cycles of length \(2^n\), using about \(\frac{n^2}{2}\) bits of storage. In both algorithms, the time required to produce the next bit from the last \(n\) bits is close to \(n\). A possible application to the construction of stream ciphers is indicated.
1. INTRODUCTION

This paper deals with the construction of full-length, non-linear, shift-register cycles, also referred to as de Bruijn sequences. A comprehensive survey of past work on this subject can be found in [1]. The common approach to this construction is to consider a shift-register producing many short cycles, e.g. the pure cycling register, which are then joined together to form a full cycle.

The same practice is followed in this paper. We propose two methods of constructing full cycles. One produces full cycles by joining those generated by the pure cycling register (PCR); the other employs the pure summing register (PSR) for the same purpose. It is well known [2] that the number of full cycles of length $n$ is $2^{2n-1} - n$.

The various methods proposed so far differ in the number of distinct full cycles of same length that the method produces, and in the complexity per produced cycle. Fredricksen [3] shows how to generate $2^{2n-5}$ full cycles of length $2^n$ from a PCR of length $n$ ($\text{PCR}_n$) using 6$n$ bits of storage, and $n$ units of time to produce the next cycle bit from the last $n$ bits.

In Section 3 of this paper, we show how to construct $2^{k \cdot g(n,k)}$ full cycles of length $2^n$ from those of PCR using $3n + k \cdot g(n,k)$ bits of storage, where $k$ is a constant in the range $1 \leq k \leq 2 \cdot 2^{n/4}$, $g(n,k)$ is approximately $(n - 2 \log k)(1 - \frac{1}{1 + \log k})$, and log means logarithm to base 2. The time required to produce the next bit from the last $n$ bits is $o(n)$.

In Section 4, we propose a method of constructing full cycles by joining the cycles of a PSR. To the best of the authors' knowledge,
about $2^{n^2/4}$, or more exactly,

$$\prod_{k=1}^{[n-2]/2} \binom{n-1}{2k}$$

full cycles of length $2^n$ using about $\frac{n^2}{2}$ bits of storage and $o(n)$ units of time to produce the next bit.

The number of full cycles and the amount of stored information required to generate them via the proposed algorithms, make it worthwhile to consider their use in cryptographic applications. The main problem in the design of a stream cipher [4] is the construction of the key stream from a, so called, key seed. In our case, the full cycle acts as the key stream, while the stored information required to run the algorithm plays the role of the key seed. The full cycles produced by the proposed algorithms possess some of the important properties desired of key streams. For instance, the number of distinct keys is exponential in the length of the seed which, in the algorithm of Section 3, depends on a controllable parameter $k$.

In addition any full cycle has many of the randomness properties [5] and a large linear span [6] required of key streams.
2. THE JOINING OF CYCLES

A feedback shift-register (FSR) of length \( n \) has \( 2^n \) states forming the set \( B^n \) of all binary \( n \)-tuples. The feedback function \( f(x) = (x_1, x_2, \ldots, x_n) \), of the FSR induces a mapping \( F: B^n \rightarrow B^n \) under which \( xF = y \) iff

\[
\begin{align*}
  y_i &= x_{i+1}, & i &= 1, \ldots, n - 1 \\
  y_n &= f(x).
\end{align*}
\]

The conjugate \( \hat{x} \) and the companion \( x' \) of a state \( x = (x_1, x_2, \ldots, x_n) \) are defined by

\[
\hat{x} = (x_1 \oplus 1, x_2, \ldots, x_n)
\]

\[
x' = (x_1, \ldots, x_{n-1}, x_n \oplus 1),
\]

where \( \oplus \) denotes mod-2 addition.

A \( k \)-cycle \( C \) of a FSR is a (cyclic) sequence of \( k \) distinct states

\[
C = \{ S_1, S_2, \ldots, S_k \}
\]

such that \( S_1 = S_kF \) and \( S_{i+1} = S_iF \), \( i = 1, 2, \ldots, k-1 \). The state-diagram of a FSR is called a factor if each state belongs to a cycle. Two cycles \( C_1 \) and \( C_2 \) are said to be adjacent if they are (state) disjoint and there exists a state \( x \) on \( C_1 \) whose conjugate \( \hat{x} \) is on \( C_2 \).

**Lemma 1:** Let \( C_1 \) and \( C_2 \) be adjacent cycles. Then, there exists a state \( z \) on \( C_1 \) whose companion \( z' \) is on \( C_2 \).

**Proof:** Since \( C_1 \) and \( C_2 \) are adjacent, there exists a state \( y \) on \( C_1 \) whose conjugate \( \hat{y} \) is on \( C_2 \). Let \( z = yF \). Then \( zF \) is either \( z \) or its companion \( z' \). Since, by definition, \( C_1 \) has no state in common with \( C_2 \), \( z = yF \) implies \( z' = \hat{y}F \). Thus, \( z \) and \( z' \) form an asserted pair of states.
Theorem 1 [5]: Two adjacent cycles \( C_1 \) and \( C_2 \), with \( x \) on \( C_1 \) and \( x' \) (\( x' \)') on \( C_2 \), are joined into a single cycle when the successors (predecessors) of \( x \) and \( x' \) (\( x' \)) are interchanged.

Example 1: Consider the PCR_3 with \( f(x_1, x_2, x_3) = x_1 \). Its state-diagram is the factor of Fig. 1(a). Its four cycles are

\[
\begin{align*}
C_1 &= \{000\} \\
C_2 &= \{001, 010, 100\} \\
C_3 &= \{011, 110, 101\} \\
C_4 &= \{111\}
\end{align*}
\]

\( C_1 \) and \( C_2 \) are adjacent, with \( 000 \) being the conjugate of \( 100 \).
Similarly, \( C_2 \) and \( C_3 \) are adjacent, with \( 010 \) being the companion of \( 011 \). Applying Theorem 1 to, say, \( C_2 \) and \( C_3 \) we obtain the cycle \( C = \{001, 011, 110, 101, 010, 100\} \). The new factor, consisting of \( C_1, C, \) and \( C_4 \), is shown in Fig. 1(b).

![Diagram](image-url)
3. CONSTRUCTION OF FULL CYCLES FROM PCR

The PCR is an $n$-stage FSR whose feedback function

$$f(x_1, x_2, \ldots, x_n) = x_1.$$  

It is well known [5] that the length of a cycle from PCR is a divisor of $n$.

The weight $W(S)$ of a state $S$ is the number of ONES in

$$S = (s_1, s_2, \ldots, s_n),$$  

i.e., $W(S) = \sum_{i=1}^{n} s_i$.

Clearly, states belonging to the same cycle of PCR have the same weight.

The weight $W(C)$ of a cycle $C$ from PCR is the weight of each of its states.

Fredricksen [3] shows how to join the cycles of PCR to form a full cycle of length $2^n$. There are 4 possibilities for consecutive states on a full cycle (or any other cycle) of a FSR:

(a) $(0, x_1, \ldots, x_{n-1}) \rightarrow (x_1, \ldots, x_{n-1}, 0)$
(b) $(1, x_1, \ldots, x_{n-1}) \rightarrow (x_1, \ldots, x_{n-1}, 1)$
(c) $(0, x_1, \ldots, x_{n-1}) \rightarrow (x_1, \ldots, x_{n-1}, \bar{x})$
(d) $(1, x_1, \ldots, x_{n-1}) \rightarrow (x_1, \ldots, x_{n-1}, 0)$

In (a) and (b) both states come from the same PCR cycle. In (c) the weight of the second state exceeds by 1 that of the first state. In (d) the weight of the first state exceeds by 1 that of the second state. Hence, in (c) and (d) the two states come from different PCR cycles.

**Lemma 2:** Let $C_1$ be a cycle of weight $k > 0$ from PCR. Then there exists a state $S$ on $C_1$ such that its companion $S'$ is on a cycle $C_2$ whose weight is $k - 1$. Since $W(C_1) > 0$, there exists a state of the form $S = (s_1, \ldots, s_n)$
on $C_1$. Hence, $S' = (s_1, \ldots, s_{n-1}, 0)$ and $W(S') = W(S) - 1 = k-1$.

Therefore $S'$ is on a $P_{n}$ cycle $C_2$, with $W(C_2) = k-1$.

Q.E.D.

Lemma 2 and Theorem 1 lead to a simple way of constructing a full cycle. At each step we have a main cycle, obtained by joining a subset of $P_{n}$ cycles, and the remaining $P_{n}$ cycles. Initially, the main cycle is chosen to be the unique $P_{n}$ cycle of weight zero. Next, the main cycle is extended by joining to it the (unique) cycle of weight one. In a general step $i$, we extend the main cycle by joining to it all the $P_{n}$ cycles of weight $i$ (in arbitrary order). This is always possible because the current main cycle contains all of the states whose weight is less than $i$ and, since each $P_{n}$ cycle of weight $i > 1$ has a state ending in a ONE, it can be joined (see Theorem 1 and Lemma 2) to the current main cycle.

This procedure ends when all the $P_{n}$ cycles have been joined together.

We proceed now to a precise and detailed description of the proposed construction.

Consider the ordered set $V = \{V(i)\}_{i=0}^{k-1}$ of $k$ states, $1 \leq k \leq 2^{n-4}$, constructed as follows:

1. The first $\lceil \log k \rceil + 1$ bits of $V(i)$ form the base-2 representation of $i$. (Note that the first bit is always ZERO.)

2. The last $\lceil \log k \rceil + 2$ bits of each $V(i)$ are ONES preceded by a single ZERO.

3. In positions $\lceil \log k \rceil + 2 + (\lceil \log k \rceil + 1)j$, for integers $j$ satisfying $0 \leq j < \left\lfloor \frac{n-\lceil \log k \rceil - 1}{\lceil \log k \rceil + 1} \right\rfloor$, each $V(i)$ has a ZERO.
Example 2: \( n = 16, \ k = 8 \). The set \( V \) for these values of \( n \) and \( k \) takes the form

\[
\begin{align*}
00000x_1^{(1)}x_2^{(1)}x_3^{(1)}x_4^{(1)}x_5^{(1)}01111 \\
00010x_1^{(2)}x_2^{(2)}x_3^{(2)}x_4^{(2)}x_5^{(2)}01111 \\
00100x_1^{(3)}x_2^{(3)}x_3^{(3)}x_4^{(3)}x_5^{(3)}01111 \\
00110x_1^{(4)}x_2^{(4)}x_3^{(4)}x_4^{(4)}x_5^{(4)}01111 \\
01000x_1^{(5)}x_2^{(5)}x_3^{(5)}x_4^{(5)}x_5^{(5)}01111 \\
01010x_1^{(6)}x_2^{(6)}x_3^{(6)}x_4^{(6)}x_5^{(6)}01111 \\
01100x_1^{(7)}x_2^{(7)}x_3^{(7)}x_4^{(7)}x_5^{(7)}01111 \\
01110x_1^{(8)}x_2^{(8)}x_3^{(8)}x_4^{(8)}x_5^{(8)}01111
\end{align*}
\]

where the \( x_j^{(i)} \) are free parameters.

It can be easily verified that the righthand block of \([\log k] + 1 \) ONES from the unique largest run of ONES in each \( V(i) \), and that every pair of states differ in their first \([\log k] + 1 \) bits. Therefore we have the following lemma:

**Lemma 3:** No two states of \( V \) belong to the same cycle of \( \text{PCR}_n \).

The construction of a full cycle from the \( \text{PCR}_n \) cycles proceeds by a sequence of joins where at each step a cycle of least weight from among the remaining \( \text{PCR}_n \) cycles is joined to the current main cycle. A join is performed by means of a pair of companion states \( S \) and \( S' \), with \( S \) on the next \( \text{PCR}_n \) cycle \( C \) in line and \( S' \) on the current main cycle. The states \( S' \) and \( S \) are called the bridging states of the join. The bridging state \( S \) on \( C \) is determined as follows: If \( C \) contains a state from \( V \) then it is chosen as the bridging state of \( C \). Otherwise, the choice of \( S \) is as in Fredrickson [3,7].
and \( r \geq 0 \), then the state \( S \) such that \( |S| = 1 \) is also on \( C \), and we take \( S \) to be the bridging state of \( C \).

In any case the chosen bridging state \( S \) for the current \( \text{PCR}_n \) cycle \( C \) always ends in a ONE. By Lemma 2, its companion \( S' \) belongs to a \( \text{PCR}_n \) cycle whose weight is smaller than that of \( C \). Therefore, \( S' \) must be on the current main cycle. By Theorem 1, interchanging the predecessors of \( S \) and \( S' \) will create the next main cycle by joining the current one with \( C \).

A full cycle obtained by joining \( \text{PCR}_n \) cycles as described above, can be generated bit-by-bit following a procedure based on the underlying rules for the joining of cycles. In this procedure, the \((i+n)\)-th bit, \( b_{i+n} \), of the full cycle is determined from the preceding \( n \)-bit state \( \beta_i = (b_i, b_{i+1}, \ldots, b_{i+n-1}) \). If \( \beta_i \) served as a predecessor of a bridging state (\( S \) or \( S' \)) then \( b_{i+n} = b_i \oplus 1 \); otherwise, \( b_{i+n} = b_i \). The formal steps for determining \( b_{i+n} \) are presented in the following algorithm.

ALGORITHM A

Choose a constant \( k \) such that \( 1 \leq k \leq \frac{n-1}{2} \). Choose and store an ordered set of bridging states \( V = \{V(i)\}_{i=0}^{k-1} \). Initially, set \( \beta_0 = (0, 0, \ldots, 0) = 0^n \). Given \( \beta_i = (b_i, b_{i+1}, \ldots, b_{i+n-1}) \), proceed to produce \( \beta_{i+1} = (b_{i+1}, \ldots, b_{i+n-1}, b_{i+n}) \) as follows:

(A1) Examine the cyclic shifts of \( \beta_i = (b_{i+1}, \ldots, b_{i+n-1}, 1) \) for the existence of a shift \( \alpha \) that begins with a ZERO and ends with \( 1 + \lfloor \log_k k \rfloor \) ONES. If no such \( \alpha \) exists go to (A3).

(A2) Let \( \alpha^* \) be the first \( 1 + \lfloor \log_k k \rfloor \) bits of \( \alpha \) and let \( |\alpha^*| = i \), the base-2 value of \( \alpha^* \). If \( i > k-1 \) go to (A3); otherwise,

(A3) Set \( \beta_{i+1} = (b_{i+1}, \ldots, b_{i+n-1}, b_{i+n}) \) as follows:

(A4) If \( b_{i+n} \) is set to \( b_i \oplus 1 \). If \( b_{i+n} \) is set to \( b_i \). The full cycle is obtained by joining the \( \text{PCR}_n \) cycles as described above.
(A3) Let \( M \) be the cyclic shift of \( \beta_i^* \) with the largest base-2 value \( |M| = l \cdot 2^r \), \( l \) odd, \( r \geq 0 \). Let \( S \) be the shift of \( \beta_i^* \) such that \( |S| = l \). If \( S = \beta_i^* \) go to (A5).

(A4) Set \( b_{i+n} = b_i \).

(A5) Set \( b_{i+n} = b_i \oplus 1 \).

Theorem 2: (i) For every choice of \( x \) in the indicated range, and of the set \( V \) Algorithm A produces a full cycle of length \( 2^n \).

(ii) For a given choice of \( k \) there are \( 2^k \cdot g(n,k) \) distinct choices for the set \( V \), where

\[
g(n,k) = n - 3 - \left[\log k\right] - \left[\log \frac{n-3}{\left[\log k\right]} - \left\lfloor\frac{\left[\log k\right]}{\log k}\right\rfloor + 1\right]
\]

thus, Algorithm A can be used to produce \( 2^k \cdot g(n,k) \) distinct full cycles.

(iii) The working space that Algorithm A requires to produce a full cycle is \( 3n + k \cdot g(n,k) \) bits and the work required to produce the next bit is \( 2n \) cyclic shifts and about the same number of \( n \)-bit comparisons.

Proof: (i) follows directly from the discussion preceding Algorithm A. (ii) is due to the fact that each \( V(i) \) is specified up to exactly \( g(n,k) \) free parameters and no state except for \( 0^{n-a}1^a \), \( a = 1 + \left[\log k\right] \), may serve as a bridging state via both of the two criteria: either by being a member of the set \( V \) or by representing the odd part of a maximal shift. This, together with Lemma 3, imply that distinct choices for the set \( V \) correspond to distinct full cycles.

(iii) follows directly from Algorithm A. Note that only information about members of the set \( V \) has to be stored and, there, only the \( g(n,k) \) free bit values of each \( V(i) \) require storage.
4. CONSTRUCTION OF FULL CYCLES FROM PSR

The PSR is an n-stage PSR whose feedback function \( f(x_1, x_2, ..., x_n) = x_1 \oplus x_2 \oplus ... \oplus x_n \).

An extended representation \( E(C) \) of a cycle \( C \) of PSR is given by an \((n+1)\)-tuple \([x_0 x_1 ... x_{n-1} x_n]\) where \((x_0, x_1, ..., x_{n-1})\) is a state on \( C \) and \( x_n = x_0 \oplus x_1 \oplus ... \oplus x_{n-1} \).

The extended weight \( W_E(C) \) of \( C \) is defined as the number of ONES in \( E(C) = [x_0 x_1 ... x_{n-1} x_n] \), i.e., \( W_E(C) = \sum_{i=0}^{n} x_i \).

The following lemma is an immediate result of the above definitions:

**Lemma 4:** For every cycle \( C \) from PSR, we have \( W_E(C) = 2k \), for some \( 0 \leq k \leq \left\lfloor \frac{n+1}{2} \right\rfloor \), and for each state \( S \) on \( C \) \( 2k-1 \leq W(S) \leq 2k \).

\( C \) is called a run-cycle if the ONES in \( E(C) \) form a cyclic run.

For each cycle \( C \) of PSR, with \( W_E(C) = 2k < n+1 \), we define a unique preferred state \( P(C) \). For a run-cycle, \( P(C) = (1^{2k}0^{n-2k}) \); for a cycle with more than one (cyclic) run of ONES the preferred state is defined as follows:

Let \( E^*(C) = [0^{r}1^{t}0^{1} ... b_{n-t-r-2}^{n-2}] \) be the unique extended representation of \( C \) which satisfies the following properties:

\begin{enumerate}
  \item \( r \geq 0 \),
  \item \( t \) is the length of the longest run of ONES,
  \item Among all extended representations of this form, with the same maximal \( t \), \( E^*(C) \) is the largest when viewed as a number in base-2 notation.
\end{enumerate}

Then, the preferred state for \( C \) is \( P(C) = (0^{r}1^{t}0^{1} ... b_{n-t-r-2}^{n-2}) \).
The companion of $P(C_1)$ are on a cycle $C_2 \neq C_1$, with $W_E(C_2) = W_E(C_1)$. Furthermore, if $t_2$ is the length of the longest run of ONES in $P(C_2)$, then either $t_2 = t_1 + 1$ or $t_2 = t_1$ and $|P(C_2)| > |P(C_1)|$.

*Proof:* Clearly $W(B) = W(P(C_1)) = W_E(C_1) = 2k$ for some $k$. Hence, by Lemma 4, $W_E(C_2) = 2k = W_E(C_1)$. It is also clear that $E(C_2) = \mathbf{1}^{t_2} \mathbf{0}_{n-1}$. Hence, if $r = 0$, $t_2 = t_1 + 1$; if $r > 0$, then an alternate extended representation of $C_2$ is given by $E'(C_2) = \mathbf{0}^{r-1} \mathbf{1}^{t_2} \mathbf{0}_{n-1} \mathbf{1}^{r-1}$, which implies $|P(C_2)| \geq |\mathbf{0}^{r-1} \mathbf{1}^{t_2} \mathbf{0}_{n-1} \mathbf{1}^{r-1}| > |P(C_1)|$.

Thus, in any case $C_2 \neq C_1$ and, since the two possible successors of $P(C_1)$ are $P(C_1)$ and the companion of $P(C_1)$, it follows that the companion of $P(C_1)$ is the successor of $P(C_2)$.

Q.E.D.

**Lemma 6:** Let $U = (u_1, \ldots, u_{n-1}, 1)$ be a state on a cycle $C_1$ of $\text{PSR}_n$ with $W(U) + 1 = W_E(C_1) = 2k$ for some $k \geq 1$. Then the companion $U'$ of $U$ is on a $\text{PSR}_n$ cycle $C_2$ with $W_E(C_2) = 2k - 2$.

*Proof:* This lemma follows directly from the definition of $U'$ and Lemma 4.

Q.E.D.

Lemmas 4, 5, 6 lead to a construction of a large class of full cycles from those of $\text{PSR}_n$. Lemma 5 suggests a way of joining all cycles with the same extended weight. For each extended weight $2k$, we start with the run-cycle of this weight as an initial main cycle. In each step the current main cycle is expanded by joining to it the $\text{PSR}_n$ cycle of extended weight $2k$ with the longest run of ONES; if there are two

...
largest preferred state. Recalling the definition of bridging states in Section 3, it is easy to verify that this order of joins is always possible if the preferred state of the PSR cycle in line is chosen as a bridging state S for the join (the described order guarantees that its companion S' belongs to the current main cycle).

Once all the PSR cycles of extended weight 2k are joined together into a corresponding main cycle $MC_k$, $0 \leq k \leq \left\lfloor \frac{n+1}{2} \right\rfloor$, we apply Lemma 6 to joining the $MC_k$ cycles, in order of increasing $k$, to form a full cycle.

We proceed now to describe an algorithm for producing the (i+n)-th bit $b_{i+n}$ of the resulting full cycle from the following inputs:

(i) the preceding n-bit state $\beta_i = (b_i, b_{i+1}, \ldots, b_{i+n-1})$,

(ii) the parity $p_i$ of $\beta_i$, $p_i = b_i \oplus b_{i+1} \oplus \ldots \oplus b_{i+n-1}$, and

(iii) the weight $W(\beta_i)$ of $\beta_i$.

The production of $b_{i+n}$ from the above inputs is based on the fact, that when $(x_0, x_1, \ldots, x_{n-1}) \oplus (x_1, \ldots, x_{n-1}, x_n)$ then $\sum_{i=0}^{n} x_i$ is even if and only if both states are on the same PSR cycle.

Before presenting the formal steps of the algorithm, we remind the reader that the preferred state S of each PSR non-run-cycle and its companion S' serve as bridging states in the process of forming one of the $MC_k$ cycles. In the process of joining the $MC_k$ cycles into a full cycle, the bridging state S on the $MC_k$ cycle in line, $1 \leq k \leq \left\lfloor \frac{n+1}{2} \right\rfloor$, can be chosen as any state of odd weight and a trailing ONE; i.e., the bridging state $S^{(k)}$ for $MC_k$ can be any state of the form $S^{(k)} = (s_1^k, s_2^k, \ldots, s_{n-1}^k, 1)$, with $W(S^{(k)}) = 2k-1$ (see Lemma 6).

In Algorithm B, given below, we first check whether the given state
we set $b_{i+n} = p_i \oplus 1, p_{i+1} = b_i \oplus 1$, and $W(b_{i+1}) = W(b_i) - b_i + (p_i \oplus 1)$; otherwise, $b_{i+n} = p_i, p_{i+1} = b_i$, and $W(b_{i+1}) = W(b_i) - b_i + p_i$.

**Algorithm B**

For every $k$ such that $1 \leq k \leq \left\lfloor \frac{n+1}{2} \right\rfloor$ choose and store a bridging state $U^{(2k)}$ of the form $U^{(2k)} = (u^k_1, u^k_2, \ldots, u^k_{n-1}, 1)$ with $W(U^{(2k)}) = 2k-1$.

Initially, set $\beta_0 = (0, 0, \ldots, 0) = 0^n$, $p_0 = 0$, $W(\beta_0) = 0$. Given $\beta_i = (b_i, b_{i+1}, \ldots, b_{i+n-1})$, $p_i$, $w_i = W(\beta_i)$ proceed to produce $\beta_{i+1} = (b_{i+1}, \ldots, b_{i+n-1}, b_{i+n})$, $p_{i+1}$, $w_{i+1}$ as follows:

- **(B1)** If $p_i \oplus b_i = 1$ go to **(B3)**.
- **(B2)** If $(b_{i+1}, \ldots, b_{i+n-1}, 1) = U(w_i - b_i + 2)$ go to **(B6)**; otherwise go to **(B5)**.
- **(B3)** If $\beta_i^* = [b_{i+1}, \ldots, b_{i+n-1}, 0]$ is a run-cycle go to **(B5)**; otherwise, find the cyclic shift $E_i^* = [0^r \cdot b_s \ldots, b_{n-t-r+s-3}, 0]$ of $\beta_i^*$ whose first $n$ bits form a preferred state.
- **(B4)** If $E_i^* = \beta_i^*$ go to **(B6)**.
- **(B5)** Set $b_{i+n} = p_i, p_{i+1} = b_i, w_{i+1} = w_i - b_i + p_i$.
- **(B6)** Set $b_{i+n} = p_i \oplus 1, p_{i+1} = b_i \oplus 1, w_{i+1} = w_i - b_i + (p_i \oplus 1)$.

**Theorem 3**

(i) For every choice of the set of states $\{U^{(2k)}_{k=1} \left\lfloor \frac{n+1}{2} \right\rfloor$ Algorithm B produces a full cycle of length $2^n$.

(ii) There are $\prod_{k=1}^{n-1} 2k$ distinct choices for the set of states $\{U^{(2k)}_{k=1} \left\lfloor \frac{n-2}{2} \right\rfloor$, thus Algorithm B can be used to produce $\prod_{k=1}^{n-1} 2k$ distinct full cycles.

(iii) The working space that Algorithm B requires to produce a full cycle is about $\frac{n^2}{2}$ bits and the work required to produce the next bit is $n$ cyclic shifts and about the same number of $n$-bit comparisons.
Proof: (i) follows directly from the discussion preceding
Algorithm B.

(ii) is due to the fact that two different sets of bridging states
produce different full cycles. The number of ways to choose the set
\( \{U^{(2k)}\} \) is

\[
\prod_{k=0}^{\frac{n-1}{2}} \left( \frac{n-1}{2k-2} \right) = \prod_{k=0}^{\frac{n-1}{2k}} (2k)! = \prod_{k=0}^{\frac{n-1}{2k}} (2k-1)!
\]

(iii) follows directly from Algorithm B.

Q.E.D.

Example 3: For \( n=6 \), and the bridging states

\[
U^{(2)} = (0,0,0,0,0,1) \quad U^{(4)} = (1,0,1,0,0,1) \quad U^{(6)} = (1,1,0,1,1,1),
\]

the full successive bits of one period of the full cycle generated
by Algorithm B are:

\[
000000110000101000111011100110
\]

\[
11111101100110110101101001001011.
\]

It should be noted that a similar algorithm can be derived for
the complement of the PSR, i.e., the FSR with the feedback function

\[
f(x_1, x_2, \ldots, x_n) = x_1 \oplus x_2 \oplus \ldots \oplus x_n \oplus 1.
\]
REFERENCES


