Permutation on binary de Bruijn sequence

Musthofa
Department of Mathematics Education, Universitas Negeri Yogyakarta, Indonesia

musthofa@uny.ac.id

Abstract. Binary de Bruijn sequences are studied by many researchers since it has many applications such as in DNA sequencing, robotics, and radar. In this paper we study the properties of binary de Bruijn sequences in term of permutation. We showed that it has a nice property that we hope it will be useful in further research.

1. Introduction

Early published by Saint Marry [1], the concept of de Bruijn sequence has become widely observed by many researcher. A binary de Bruijn sequence of order \( n \) is binary sequence with property that every \( n \)-tuple appear exactly once in every one period of the sequence[2]. For example, there is only one binary de Bruijn sequence of order two that is 0011 and for order three, there are two different sequences, 00010111 and 00011101. In general, for any positive integer \( n \), there are \( 2^{2^{n-1}-n} \) different sequences [3].

Some applications of this sequence are in cryptography, robotics, radar and DNA based storage. In cryptography, this sequence is used as a key stream generator [4]. In robotics, it is showed that finding the maximum number of robot that can be used is equivalent to finding the maximum number of disjoint \( k \)-cycles in the de Bruijn graph [5]. The application of this sequence in radar can be found in [6] and [7]. Furthermore, in DNA based storage de Bruijn sequence becomes basic tool for read process in DNA sequencing [8].

The number of de Bruijn sequence grows exponentially for large \( n \). For certain \( n \), one of the interesting problems is how to construct the sequence efficiently. Several methods to construct the sequence have been developed by many researchers. Golomb constructed de Bruijn sequence by using primitive polynomial called linear feedback shift register [6]. The other methods are developed by Martin [9] and Alhakim [10], which are based on greedy algorithm. Recently, Swada, Williams, and Wong proposed new methods on construction binary de Bruijn sequence called successor rules [11].

The main aspects of determining the goodness of construction methods are based on two criteria, that are the efficiency of the algorithm and the number of different constructed sequence. The linear feedback shift register can be implemented efficiently for constructing the sequence, however it needs different primitive polynomial for different sequence. The greedy algorithm is an efficient way to construct the sequence, but need large memory to remember the previous stages. The successor rule provided by Swada, Williams and Wong is more efficient since it does not need to store the previous sequence, however it has to check the lexicographically smallest string in the stage called necklace.

In this paper we look at each stage of the binary de Bruijn sequence based on linear feedback shift register and we consider that these are element of \( \mathbb{Z}_2^n \). Therefore, we consider each element as an element of permutation group. We obtain some nice properties of the sequence in terms of element of
permutation. Moreover, we expect that these results can be combined with the existing algorithms to construct the binary de bruijn sequence efficiently.

2. Binary De Bruijn Sequence

A binary de bruijn sequence of order \( n \) is a circular string of length \( 2^n \) which every substring of length \( n \) occurs exactly one. The string is taken from \{0,1\}. Here are some examples of them. First, 01 and 0011 is the only binary de bruijn sequence of order one and order two respectively. The substrings are 0 and 1 for order one and 00, 01, 11, 10, for order two. There are two different binary sequences for order three, that are 00010111 and 00011101 which every substring of length 3, i.e string from the \{000, 001, 011, 110, 101, 010, 101, 100\} appears exactly one.

One of the problems in the de bruijn sequence is how to construct the sequence efficiently. Many researchers proposed various algorithms. However, the drawbacks of the proposed algorithms are interested to be resolved. Some of the algorithms are discussed below.

2.1. Linear Feedback Shift Register

This method is proposed by Golomb based on primitive polynomial [6]. By this method, constructing different order of de bruijn sequence needs different primitive polynomial. For example, take primitive polynomial \( x^3 + x + 1 \) over GF(2) and choose the initial stage 001. The sequence generates by this polynomial is 0010111. Adding 0 in the most zero term of the sequence then we get the binary de bruijn sequence 00010111.

If we choose \( x^3 + x^2 + 1 \) to generate the sequence, with initial stage 001, then we will get the other binary de bruijn sequence 00011101. Furthermore, different initial stages with the same polynomial provide the same sequence. Suppose we take \( x^4 + x + 1 \) and 0001 as the initial stage. This polynomial generates the sequence 000100110111111. However, if we select all possible initial stages, it results the same sequence as seen in the table below [6]. Using this method, it can be concluded that it need different primitive polynomial to construct the de bruijn sequence and adding 0 in the most zero term at the end.

| Initial stage | Sequence       |
|---------------|----------------|
| 0001          | 0001001101011111 |
| 0010          | 001001101011101 |
| 0100          | 010011010111011 |
| 0100          | 100100110111011 |
| 1000          | 100010011011101 |
| 0011          | 001010111110001 |
| 0101          | 010111100010011 |
| 0110          | 011010111100010 |
| 1001          | 100110111110000 |
| 1010          | 101011100010010 |
| 1100          | 110001001101101 |
| 0111          | 011100010011011 |
| 1011          | 101110001001101 |
| 1101          | 110101111000100 |
| 1110          | 111000100110101 |
| 1111          | 111100010011010 |

2.2. Prefer- opposite algorithm

Proposed by Alhakim [10], the steps of this algorithm for sequence of order \( n \) are as follows:

Step 1. Start with initial stage \( x_1x_2\ldots x_n = 00\ldots0 \)

Step 2. \( i = n + 1 \)
Step 3. If \( x_{i-n+1} \ldots x_{i-1} \bar{x}_{i-1} \) is a pattern that has not appeared before than \( x_i = \bar{x}_{i-1} \), increment \( i \) by one and repeat step 3.

Step 4. Otherwise, if \( x_{i-n+1} \ldots x_{i-1} x_{i-1} \) is a pattern that has not appear earlier in the sequence then \( x_i = x_{i-1} \), increment \( i \) by one and go to step 3.

Step 5. Otherwise, stop.

Compared to other greedy algorithm such as prefer one algorithm established by several researchers [12], Alhakim stated that this algorithm tends to keep balance between 1's and 0's. Prefer one algorithm is a very simple algorithm that can generate de Bruijn sequence. Start with \( n \) zeros and append 1 in the next stage if the pattern has not appeared before, otherwise take 0. The comparison of the constructed sequence from both algorithms is shown in the table below for \( 1 \leq n \leq 5 \) [10].

| \( n \) | Prefer one | Prefer opposite |
|--------|------------|-----------------|
| 1      | 01         | 01              |
| 2      | 0011       | 0011            |
| 3      | 0001110101 | 00010111        |
| 4      | 000011110110010101 | 0000101001101111 |
| 5      | 000001111101110011010110010101 | 00000101011010010001100111011111 |

2.3. **Necklace based successor rule**

The prefer-opposite algorithm needs big space to remember all of the string in stage. Swada, William and Wong developed new algorithm based on the set of equivalence cyclic strings. A Necklace is lexicographically smallest element in the equivalence class of cyclic string. For example, 0001, 0010, 0100, 1000 are in the same class and 0001 is the necklace.

Constructing binary de Bruijn sequence by the **necklace based successor rule** is very simple by applying the following rule:

\[
 f(x_1x_2 \ldots x_n) = \begin{cases} 
 x_2 \ldots x_n \bar{x}_1, & \text{if } x_2 \ldots x_n 1 \text{ is a necklace} \\
 x_2 \ldots x_n x_1, & \text{otherwise}
\end{cases}
\]

The symbol \( \bar{x} \) means the complement of \( x \) or in binary it means \( 1 - x \). Suppose we want to obtain binary de Bruijn sequence of order 3 by applying the **necklace based successor rule**. Let 000 be the initial string, then the next strings are 001, 011, 111, 110, 101, 010, 100. Finally, by concatenating the first symbol in each string we get the binary de Bruijn of order 3, that is 00011101.

3. **De Bruijn Graph**

De Bruijn problems become famous topic after Nicolaas De Bruijn published his article in 1946, “A combinatorial problem”. He interested in finding the shortest sequence that contains every substring of length \( n \) occurs exactly once in the sequence. He solved this problem by using graph theory. Every substring is placed as vertex and there is an edge from vertex \( A \) to vertex \( B \) if the last \( (n-1) \) string of \( A \) is equal to the first \( (n-1) \) string of \( B \). For example, if \( A = 001 \) and \( B = 011 \) then the last two string of \( A \) is \( 01 \) which is the same as the first two string in \( B \). Consequently, there is an edge from \( A \) to \( B \). The edge then labelled as 0011.

By this method, the shortest sequence is the hamiltonian cycle or the eulerian cycle in the graph. Therefore, if want to generate all of the possible sequences, then we have to find all of the eulerian cycle in the graph. For example, if \( n = 3 \), then the de Bruijn graph is as follow:
By determining all of the eulerian cycles in the graph, we obtain all of the de bruijn cycles. However, for large $n$, this method is not practical since it needs large memory to store the graph.

4. Result and Discussion
Let $S = \{1, 2, \ldots, n\}$. Permutation on $S$ is a bijective mapping from $S$ to $S$. Usually, a permutation on the set $S$ can be written as $\left( f(1) \ f(2) \ f(3) \ldots \ f(n-1) \ f(n) \right)$. In this paper we see that each stage in de bruijn sequence or each vertex in the de bruijn graph is an element of the group $\mathbb{Z}_2^n$. For example, the sequence 00011101 can be seen as a permutation of $\{000, 001, 010, 011, 111, 110, 100\}$. This can be written as $\left( 000 \ 001 \ 010 \ 011 \ 111 \ 110 \ 100 \right)$. In this part we treat de bruijn sequence as a cycle of permutation. Consider the previous sequence 00010111. It consists of 8-stages: $000 \rightarrow 001 \rightarrow 010 \rightarrow 101 \rightarrow 011 \rightarrow 111 \rightarrow 110 \rightarrow 100$. The cycle form of this sequence is $\left( \begin{array}{cccccccc} 000 & 001 & 010 & 011 & 111 & 110 & 100 \\ 000 & 001 & 010 & 011 & 111 & 110 & 100 \\ \end{array} \right)$. In order to observe the property of the sequence, we identify the element as follow: $000 = e; 001 = a; 010 = b; 101 = c; 011 = d; 111 = f; 110 = g; 100 = h$. Thus, we have the group $= \{e, a, b, c, d, f, g, h\}$. Now, we have the following results:

$\begin{align*}
    a + b &= d \\
    b + c &= f \\
    c + d &= g \\
    d + f &= h \\
    f + g &= a \\
    g + h &= b \\
    h + a &= c.
\end{align*}$
From the result above, we obtain that if we know the results of any addition of any nonzero two consecutive stages than we also know the other results for any addition of nonzero two consecutive stages.

Continuing the investigation, we try to check the result of the following addition:

\[
\begin{align*}
 a + c &= h \\
 b + d &= a \\
 c + f &= b \\
 d + g &= c \\
 f + h &= d \\
 g + a &= f \\
 h + b &= g \\
 a + d &= b \\
 b + f &= c \\
 c + g &= d \\
 d + h &= f \\
 f + a &= g \\
 g + b &= h \\
 h + c &= a \\
\end{align*}
\]

Continuing the pattern, we get a nice property that is every addition of two elements which is in the same “distance” yields the element in the same “distance”. Thus we can say that all of the results are also the permutation of the elements. Later, we hope that the result will be useful in the DNA based storage in term of random access. For example, if we know the position of sequence \(a\) and \(b\) than hopefully we also know the position of sequence \(a + b\).

Moreover, we can combine the result and the existing algorithms such as greedy algorithm to construct de bruijn sequence. Suppose we want to construct the binary de bruijn sequence 00011101 with initial stage 000. By using prefer-one method, the next four consecutive stages are 001, 011, 111 and 110. By using the above property, we obtain the future consecutive stages in the sequence are 101, 010 and 100. The result is obtained by doing the following computation: 001 + 011 = 010, 011 + 111 = 100, 111 + 110 = 001, 001 + 111 = 110, 011 + 110 = 101.

5. Conclusion

Binary de bruijn sequence can be seen as an element of permutation on \(Z_2^n\). In terms of addition in the cycle, the elements are ordered in the nice way that have the same “distance”. Combined with the existing algorithms, we can generate the de bruijn sequence. Furthermore, we hope that the result will be useful in the other problem such as DNA based storage in term of random access.

References

[1] C.Flye Sainte Marie. Solution to Problem Number 58. Intermediare Math.1 (1894), 107-110
[2] Z. Chang, M.F Ezerman, S. Ling, and H. Wang. Construction of de Bruijn sequences from product of two irreducible polynomials. Cryptography and Communications, pages 1–25, 2017.
[3] N. G. de Bruijn. A combinatorial problem. Proc. Kon. Ned. Akad. Wetensh., vol. 49, no. 7, pp. 758–764, Jun. 1946.
[4] Menezes, A.J., Vanstone, S.A., Oorschot, P.C.V.: Handbook of Applied Cryptography, 1st edn. CRCPress, Inc., Boca Raton, FL, USA (1996).
[5] T Grubman, Y A Şekercioğlu, "Wood D R. Partitioning de Bruijn graphs into fixed-length cycles for robot identification and tracking", Discrete Applied Mathematics, vol. 213, pp. 101-113, 2015.
[6] Golomb, S.W., Gong, G.: Signal Design for Good Correlation: forWireless Communication,
Cryptography, and Radar. Cambridge Univ. Press, New York (2004)

[7] S. Andrenacci, E. Gambi, C. Sacchi, S. Spinsante, "Application of de Bruijn sequences in automotive radar systems: Preliminary evaluations", Proc. of 2010 IEEE Radar Conference, pp. 959-964, 2010.

[8] Compeau, Phillip, Pavel A. Pevzner and Glenn Tesler. “How to apply de Bruijn graphs to genome assembly,” 2011.

[9] M. H. Martin, A problem in arrangements, Bull. Amer. Math. Soc., 40 (1934), pp. 859-864.

[10] A. Alhakim, A simple combinatorial algorithm for de Bruijn sequences, Amer. Math. Monthly 117 (8) (2010) 728–732.

[11] J. Sawada, A. Williams, and D. Wong. A surprisingly simple de Bruijn sequence construction. Discrete Math., 339:127–131, 2016.

[12] H. Fredricksen. A survey of full length nonlinear shift register cycle algorithms. SIAM review, vol. 24, no. 2, pp. 195–221, 1982.