Acta Polytechnica **53**(4):344–346, 2013

© Czech Technical University in Prague, 2013
available online at http://ctn.cvut.cz/ap/

ON AN ALGORITHM FOR MULTIPERIODIC WORDS

Štěpán Holub*

Department of Algebra, Faculty of Mathematics and Physics, Charles University, Sokolovská 83, 175 86 Praha

* corresponding author: holub@karlin.mff.cuni.cz

Abstract. We consider an algorithm by Tijdeman and Zamboni constructing a word of length k that has periods $p_1, \dots, p_r$, and the richest possible alphabet. We show that this algorithm can be easily stated and its correctness briefly proved using the class equivalence approach.

Keywords: periodicity, combinatorics on words.

1. A short (personal) history

Non-trivial words with a given set of periods $P = \{p_1, p_2, \dots, p_r\}$ have received a lot of attention in the past decade. The motivation was to generalize the result by Fine and Wilf dealing with two periods, which has become part of the folklore. A word with periods P is called *trivial* if $\gcd(P)$ is its period too. Papers [1] and [2] are two (independent) results considering non-trivial such words with maximal length and maximal cardinality of the alphabet. These papers supplemented some older research of Castelli, Mignosi, Restivo and Justin (see, e.g., [3] for more details and references). Already in 1998, I wrote a short manuscript giving an analogous result (without considering publishing it), which I showed to Sorin Constantinescu during the WORDS 2003 conference in Turku, where he presented their results. Since this was passed without notice in the subsequent publication, and since I considered my approach simpler and more natural, I later decided to publish it in [4]. There was a gap in my paper, discovered by Gwénaël Richomme, which is fixed in [5].

The present paper extends the same approach to the construction of the richest word with a given set of periods and a given length. The basic idea is to consider relations defined by the periods and understand letters as (names of) equivalence classes generated by those relations. The idea is obvious and well known, usually expressed using the graph terminology (edges and connected components), rather than the algebraic terminology (relations and equivalence classes). Tijdeman and Zamboni [3] point out that the straightforward algorithm based on the graph approach is "simple but inefficient", and then present an algorithm based on less transparent combinatorial analysis. The aim of this paper is to give a short description of their algorithm, as well as a short and intuitive proof of its correctness, using consistently the graph/equivalence viewpoint.

2. Notation

Let w be a word of length k over an alphabet A. The set of all letters that occur in w is denoted by $\mathrm{alph}(w)$. The i-th letter of w is denoted by $w[i-1]$ so that $w = w[0]w[1]\cdots w[k-1]$. The prefix of w of length n is denoted by $\mathrm{pref}_n(w)$.

We say that a positive integer p is *a period* of a word w if $w[i] = w[i+p]$ for all $0 \le i \le |w| - p - 1$ (where $|w|$ denotes the length of the word). Note that any $p \ge |w|$ is a period of u. If P is a set of positive integers such that each $p \in P$ is a period of w, we say that w has periods P.

The word of length k having periods P and the maximal possible cardinality of $\mathrm{alph}(w)$ is called an *FW-word relative to P* (where FW stands for "Fine and Wilf" for historic reasons). The word is called *trivial with respect to P* if $\gcd(P)$ is a period of w. The longest non-trivial FW-word relative to P is called an *extremal FW-word relative to P*. We denote its length by $\mathcal{L}(P)$ (note that $\mathcal{L}(P) = L(P) - 1$ where $L(P)$ is the notation adopted in [3]).

3. Classes of Equivalence

Let w be a word which has periods P. For the rest of the paper we denote $m = \min P$. Obviously, if $i \equiv j \bmod m$ or $|i - j| \in P$, then $w[i] = w[j]$. These two conditions induce the relation $\sim_{P,k}$ on integers $\{0, \dots, k-1\}$ defined by:

$i \sim_{P,k} j$ if

a) $i \equiv j \bmod m$, or

b) there are integers $i', j' \in \{0, \dots, k-1\}$ such that

$$i \equiv i' \bmod m, \quad j \equiv j' \bmod m$$

and

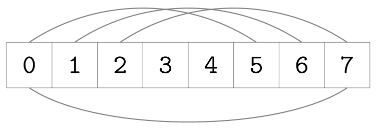
$$|i' - j'| \in P.$$

Let $\approx_{P,k}$ be the equivalence closure of $\sim_{P,k}$. In other words, we have $i \approx_{P,k} j$ if and only if i and j lie in the same connected component of the graph defined by edges $i \sim_{P,k} j$. The class of $\approx_{P,k}$ containing i will be denoted by $[i]_{P,k}$ and represented by its minimal element $\min[i]_{P,k}$. Then we define a word $\mathtt{FW}(P,k)$ of length k over the alphabet $\mathbb{N}$ by

$$\mathtt{FW}(P,k)[i] = \min[i]_{P,k}.$$

The construction immediately yields that $\mathtt{FW}(P,k)$ is the unique (up to renaming of letters) FW-word of length k relative to P. The alphabet eventually used in $\mathtt{FW}(P,k)$ depends on the number of equivalence classes. In fact, its cardinality is part of the information yielded by the algorithm.

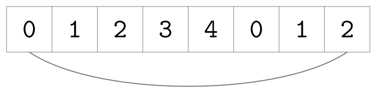
Example 1. Let $P = \{5, 7\}$. The following picture illustrates the construction of the word $\mathtt{FW}(P,8) = \mathtt{01034010}$. The upper edges correspond to $i \equiv j \bmod 5$, and the lower edge corresponds to $|i - j| = 7$.



Note that the condition

- $i \sim_{P,k} j$ if $|i - j| \in P$

would alone be enough in order to generate the equivalence $\approx_{P,k}$. However, conditions a) and b), though more complicated, are convenient since they allow to limit the number of equivalence classes to m (and the alphabet to $\{\mathtt{0}, \mathtt{1}, \ldots, \mathtt{m-1}\}$) from the very beginning. Consider, for example, how $\mathtt{0} \approx_{P,8} \mathtt{2}$ is immediately seen in the following adjusted picture.



4. The algorithm

The basic step of the algorithm is the reduction of P to a new set of periods Q defined by

$$Q = \{p - m \mid p \in P, p \neq m\} \cup \{m\} \tag{1}$$

(where $m = \min P$ according to our convention). This reduction is, in fact, one step in the Eucledean algorithm, and is well known in the literature on multiperiodic words. The key fact about P and Q is expressed in the following lemma, which is an improved version of Lemma 2 from [4].

Lemma 1. *Let $k \geq 0$. Then for all $i, j \in \{0, 1, \ldots, k\}$*

$$[i]_{Q,k} = [j]_{Q,k} \quad \textit{if and only if} \quad [i]_{P,k+m} = [j]_{P,k+m}.$$

Proof. "⇒": If $[i]_{Q,k} = [j]_{Q,k}$, then there is a sequence $i = i_0, \ldots, i_\ell = j$, of numbers from $\{0, 1, \ldots, k-1\}$ such that

$$i_s \sim_{Q,k} i_{s+1}$$

for each $s = 0, \ldots, \ell - 1$. The relation $i_s \sim_{Q,k} i_{s+1}$ implies $i_s \sim_{P,k+m} i_{s+1}$, since either

- $i_s \equiv i_{s+1} \bmod m$, or
- $\max\{i_s, i_{s+1}\} + m - \min\{i_s, i_{s+1}\} \in P$.

Therefore $[i]_{P,k+m} = [j]_{P,k+m}$.

"⇐": On the other hand, let $i = i_0, \ldots, i_\ell = j$, be a sequence of numbers from $\{0, \ldots, k+m-1\}$, with $i, j \in \{0, \ldots, k-1\}$, such that

$$i_s \sim_{P,k+m} i_{s+1}$$

for each $s = 0, \ldots, \ell - 1$. Certainly, we can suppose that the numbers in the sequence are pairwise distinct, whence $|i_s - i_{s+1}| \geq m$ and both $\min\{i_s, i_{s+1}\}$ and $\max\{i_s, i_{s+1}\} - m$ are in $\{0, 1, \ldots, k-1\}$. We now see that

$$\max\{i_s, i_{s+1}\} - m \sim_{Q,k} \min\{i_s, i_{s+1}\}.$$

Therefore the sequence

$$i = i_0, (i_0 \bmod m), \ldots, (i_\ell \bmod m), i_\ell = j$$

proves $[i]_{Q,k} = [j]_{Q,k}$. ∎

We have an immediate corollary.

Corollary 1. *For any $k \geq 0$, the word $\mathtt{FW}(Q,k)$ is a prefix of $\mathtt{FW}(P, k+m)$.*

The following lemma is an easy observation.

Lemma 2. *Let $k - m \leq i \leq m - 1$. Then $[i]_{P,k} = \{i\}$.*

Proof. Both $i - p$ and $i + p$ are out of range $\{0, 1, \ldots, k-1\}$ for any $p \in P$ (including m). Therefore i is not related by $\sim_{P,k}$ to any other element. ∎

From Corollary 1 and Lemma 2, the formula

$$\mathcal{L}_P = m + \max\{\mathcal{L}_Q, m - 1\}$$

can be readily derived (see [4, 5]). In addition, it yields the following construction of $\mathtt{FW}(P,k)$, equivalent to Algorithm B described in [3].

FW-construction (Algorithm B).

(1.) If $k \leq m$, then Lemma 2 gives

$$\mathtt{FW}(P,k) = \mathtt{0} \cdot \mathtt{1} \cdots (\mathtt{k-1}).$$

(Recall that we consider integers as letters. To stress this, we use the typewriter font for them. The multiplication sign means concatenation).

(2.) Let $k > m$. Since the word $\mathtt{FW}(P,k)$ has a period m, it is determined by its prefix w of length m. Denote $u = \mathtt{FW}(Q, k-m)$. Corollary 1 and Lemma 2 imply that

- $w = \mathrm{pref}_m(u)$ if $m \leq k - m$, and
- $w = u \cdot |\mathtt{u}| \cdot (|\mathtt{u}| + \mathtt{1}) \cdots (\mathtt{m-1})$ otherwise.

This can be succinctly stated as:

$$\mathtt{FW}(P,k)[i] = \begin{cases} \mathtt{FW}(Q, k-m)[i \bmod m] \\ \qquad\qquad \text{if } (i \bmod m) < k - m; \\ i \bmod m \quad \text{otherwise.} \end{cases}$$

Example 2. Let $P = \{5, 7\}$ and $k = 8$ as in Example 1. Recursive definition of $\texttt{FW}(P, 8)$ leads to

$$\begin{aligned} P = Q_0 &= \{5,7\} & k = k_0 &= 8 \\ Q_1 &= \{2,5\} & k_1 &= 3 \\ Q_2 &= \{2,3\} & k_2 &= 1 \end{aligned}$$

In order to obtain the word

$$u_0 = \texttt{FW}(Q_0, k_0) = \texttt{FW}(P, 8)$$

we will need words

$$u_1 = \texttt{FW}(Q_1, k_1) \quad \text{and} \quad u_2 = \texttt{FW}(Q_2, k_2).$$

Since $k_2 = 1$, we have $u_2 = \texttt{0}$. From the point (2.) above we have

$$u_1 = \mathrm{pref}_3(w_1^\omega) \quad \text{where} \quad w_1 = \texttt{01}.$$

Therefore $u_1 = \texttt{010}$. Similarly, we get

$$u_0 = \mathrm{pref}_8(w_0^\omega) \quad \text{where} \quad w_0 = \texttt{01034},$$

whence $\texttt{FW}(P, 8) = \texttt{01034010}$.

Schematically:

$$\begin{tikzcd}
Q_0=\{5,7\} \arrow[dd] & k_0=8 & u_0=\texttt{01034010} & w_0=\texttt{01034} \arrow[l] \\
Q_1=\{2,5\} & k_1=3 & u_1=\texttt{010} \arrow[ur] & w_1=\texttt{01} \arrow[l] \\
Q_2=\{2,3\} & k_2=1 \arrow[r] & u_2=\texttt{0} \arrow[ur] &
\end{tikzcd}$$

From the above example we see that the procedure has two parts: "descending" and "ascending", which are called "Reduction" and "Extension" in [3]. The end of reduction can be defined in several ways. We have seen that we can turn to extension as soon as we know $\texttt{FW}(Q_i, k_i)$. This typically happens if $k_i \le \min Q_i$, or if $\min Q_i = \gcd(Q_i)$.

5. Concluding remarks

As already remarked, the above algorithm is identical with Algorithm B from [3]. Even all arguments we use can be in some way traced back to similar arguments in the literature. Nevertheless, I believe that the description presented here provides further evidence that the equivalence class approach is not only simple but it also yields an intuition sufficient to formulate and understand the construction. (Another elegant example, in my opinion, is the proof of the fact that the extremal FW-word is a palindrome, given in [4].)

That said, one should stress that the inefficiency claim concerning the equivalence class approach is valid if we consider the naïve procedure suggested by Example 1. The precise computational complexity of Algorithm B goes beyond the scope of this paper (see the discussion in [3]).

One possible drawback can be a bit discouraging notation like $\sim_{P,k}$, and the fact that notions like "equivalence closure" may sound "too algebraic" to some ears. Computer theorists could therefore like to translate the exposition into graph language and speak about edges instead of generating relations, and about connected components instead of equivalence classes. The rest will be the same.

Acknowledgements

This work was supported by Czech Science Foundation grant 13-01832S.

References

[1] S. Constantinescu, et al. Generalised Fine and Wilf's theorem for arbitrary number of periods. *Theoret Comput Sci* **339**(1):49–60, 2005.

[2] R. Tijdeman, et al. Fine and Wilf words for any periods. *Indag Math (NS)* **14**(1):135–147, 2003.

[3] R. Tijdeman, et al. Fine and wilf words for any periods II. *Theor Comput Sci* **410**(30-32):3027–3034, 2009.

[4] Š. Holub. On multiperiodic words. *Theor Inform Appl* **40**(4):583–591, 2006.

[5] Š. Holub. Corrigendum: On multiperiodic words. *Theor Inform Appl* **45**(4):467–469, 2011.