Iliopoulos, C.S., Moore, D. and Smyth, W.F. (1998) The covers of a circular Fibonacci string. Journal of Combinatorial Mathematics and Combinatorial Computing, 26 . pp. 227-236.

THE COVERS OF A CIRCULAR FIBONACCI STRING

Costas S. Iliopoulos

Department of Computer Science
King's College London, University of London
e-mail: `csi@dcs.kcl.ac.uk`

School of Computing
Curtin University of Technology
e-mail: `csi@cs.curtin.edu.au`

Dennis Moore

School of Computing
Curtin University of Technology
e-mail: `moore@cs.curtin.edu.au`

W. F. Smyth

Department of Computer Science & Systems
McMaster University
e-mail: `smyth@mcmaster.ca`

School of Computing
Curtin University of Technology
e-mail: `smyth@cs.curtin.edu.au`

ABSTRACT

Fibonacci strings turn out to constitute worst cases for a number of computer algorithms which find generic patterns in strings. Examples of such patterns are repetitions, Abelian squares, and "covers". In particular, we characterize in this paper the covers of a circular Fibonacci string $C(F_k)$ and show that they are $\Theta(|F_k|^2)$ in number. We show also that, by making use of an appropriate encoding, these covers can be reported in $\Theta(|F_k|)$ time. By contrast, the fastest known algorithm for computing the covers of an arbitrary circular string of length n requires time $O(n \log n)$.

1. Introduction

For any nonnegative integer k, a Fibonacci string F_k is defined as follows: $F_0 = b$, $F_1 = a$, while for $k \geq 2$, $F_k = F_{k-1}F_{k-2}$. The number of elements in F_k is called its *length*, denoted by $f_k = |F_k|$, where of course f_k is a Fibonacci number. For every pair of integers i and j satisfying $1 \leq i \leq j \leq f_k$, $F_k[i..j]$ denotes a *substring* of F_k; when $i = j$, we write $F_k[i..i] \equiv F_k[i]$, the element at the i^{th} *position* in F_k.

Fibonacci strings are important in many contexts [B86], but our main interest in them here will be as examples of the worst case behaviour for algorithms which compute repetitions or (in some well-defined sense) "approximate" repetitions in arbitrary given strings. If x is a string of length n which contains a substring $x[i..j] = u^m$ for some greatest integer $m \geq 2$, then u^m is said to be a *repetition* in x if and only if u is nonempty and not itself a repetition. Thus $F_5 = abaababa$ contains the four repetitions $F_5[1..6] = (aba)^2$, $F_5[3..4] = a^2$, $F_5[4..7] = (ab)^2$, and $F_5[5..8] = (ba)^2$. Note also that, according to this definition, $x = a^n$ contains only the single repetition a^n. There are three well-known algorithms which compute all the repetitions in a given string x of length n [AP83,C81,ML84]; each of these algorithms executes in time $\Theta(n \log n)$, a bound that is known to be lowest possible [ML84]. Thus $\Theta(n \log n)$ is an upper bound on the number of repetitions which can possibly occur in any string x, and, as Crochemore has shown [C81], this bound is in fact achieved by the Fibonacci strings. In fact, the squares in a Fibonacci string have recently been completely characterized [IMS95].

The idea of a repetition can be weakened in the following way: if for some greatest integer $m \geq 2$, $y = u_1u_2 \cdots u_m$ is a substring of x such that for every integer $i \in 2..m$, u_i is a permutation of u_1, then y is said to be a *weak repetition* in x. (In the case that $m = 2$, y is sometimes called an *Abelian square*.) Clearly every repetition is a weak repetition, and, in addition to the four repetitions listed above, F_5 also contains the weak repetitions $F_5[2..5] = (ba)(ab)$ and $F_5[3..8] = (aab)(aba)$. There is only one known algorithm [CS95] to compute all the weak repetitions in a given string x. This algorithm requires $\Theta(n^2)$ time and, as shown in [CS95], F_k in fact contains $\Theta(f_k^2)$ weak repetitions, thus again achieving the upper bound.

The idea of a repetition can be generalized in another way. If every position of a given string x of length n lies within an occurrence of a substring u within x, then u is said to be a *cover* of x. If, in addition, $|u| < n$, we call u a *proper cover* of x. For example, x is always a cover of x, and $u = aba$ is a proper cover of F_5. We see that if $x = u^m$ is a repetition, then it follows that u is a cover of x. There exists a linear time algorithm to compute all the covers of x [MS95], and it is not difficult to show that x has at most $O(\log n)$ covers; it follows from Lemma 2.5 of [IMS95] that F_k has $\lfloor (k-3)/2 \rfloor = \Theta(\log f_k)$ proper covers, and so here also F_k attains the upper bound.

The *circular string*, denoted $C(x)$, corresponding to a given string x, is the string formed by concatenating $x[1]$ to the right of $x[n]$. It turns out also to be of interest to compute the covers (of length at most $|x|$) of a circular string $C(x)$ [IMP93], but, surprisingly, the number of covers of $C(x)$ can greatly exceed the number of covers of x. In this paper we characterize the covers of $C(F_k)$ and, as a byproduct, show that they are $\Theta(f_k^2)$ in number. Notwithstanding this fact, the algorithm described in [IMP93] reports $\Theta(n^2)$ covers in $\Theta(n \log n)$ time by making use of an appropriate encoding of the output. As we shall see, in the particular case $x = F_k$, the covers of $C(F_k)$ can actually be reported in time $\Theta(f_k)$ provided a certain encoding of the output is acceptable to the user.

2. Characterizing The Covers

Our results are based on two fundamental lemmas already proved in [IMS95]:

Lemma 2.1 For any integer $k \geq 2$, let

$$P_k = F_{k-2}F_{k-3}\cdots F_1. \qquad \ldots(2.1)$$

Then $F_k = P_k\delta_k$, where $\delta_k = ab$ if k is even, and $\delta_k = ba$ otherwise.

Proof Easily proved by induction: see Proposition 1 of [L81] and Lemma 2.8 of [IMS95]. □

In order to state the second lemma, we introduce the idea of a "rotation" of a given string x of length n: for every integer $j \in \{0...n-1\}$,

$$R_j(x) = x[j+1..n]x[1..j]$$

is called the j^{th} *rotation* of x. We observe that $R_0(x) \equiv x$ and that $C(x) = C(R_j(x))$ for every value of j; thus $R_j(x)$ is a cover of $C(x)$.

Lemma 2.2 For every integer $k \geq 2$, $F_k \neq R_j(F_k)$ for any integer $j \in 1..f_k - 1$.

Proof See Lemma 2.6 of [IMS95]. □

A third technical lemma also turns out to be useful.

Lemma 2.3 For every integer $k \geq 2$, F_{k-2} covers F_k with exactly 3 occurrences: as prefix of F_k, as a suffix of F_k and at position $f_{k-2}+1$. These are the only occurrences of F_{k-2} in F_k.

Proof One can see that

$$F_k = F_{k-1}F_{k-2} = F_{k-2}F_{k-2}F_{k-5}F_{k-4} \quad .$$

Thus three occurrences of F_{k-2} actually cover F_k (see Fig. 1(a)). That there are no other occurrences of F_{k-2} in F_k follows from the observation that any other occurrence of F_{k-2} would necessarily equal a rotation $R_j(F_k)$, $j > 0$, in contradiction to Lemma 2.2. Observe that in $C(F_k)$, the first occurrence of F_{k-2} and the second occurrence of F_{k-2} are preceded by $\delta_k = \delta_{k-2}$, while the third occurrence of F_{k-2} is preceded by δ_{k-1}. See also Theorem 2.2 of [IMS95]. ▫

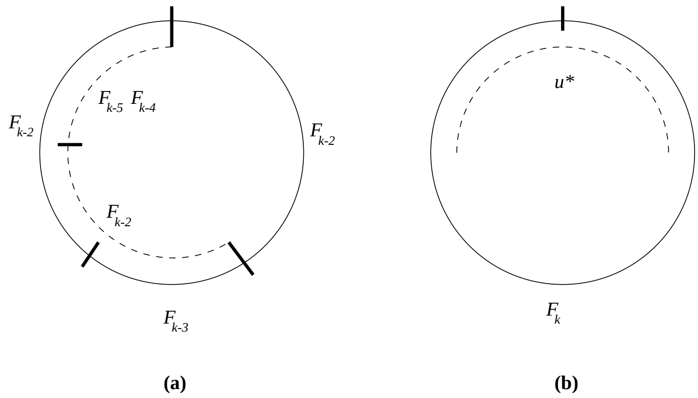

Fig. 1

The circle represents the cyclic string $C(F_k)$.

A circular string $C(x)$ has n possible representations: $x[i..n]x[1..i-1]$ for $i \in \{1,...,n\}$ (see [IS92]). Here, we use the convention that the first position of $C(x)$ is the position that a (randomly chosen) occurrence of x starts and that the positions in $C(x)$ increase clockwise. Note that in general, the string x may be the prefix of more than one representation (see [IS]). It also convenient to use $s^{(h)}, h = 1, 2...$ to denote the h-th occurrence of a substring s in $C(x)$. For example $F_{k-2}^{(2)}$ occurs at position $f_{k-2}+1$ of $C(F_k)$ and $F_{k-2}^{(3)}$ occurs at position $f_{k-1}+1$ of $C(F_k)$ (see Fig. 1(a)).

In establishing our results, we employ the following strategy:

- Making use Lemmas 2.1, 2.2, and 2.3, we first show that every cover u of $C(F_k)$ is necessarily a substring of P_k as defined in (2.1); that is, u cannot contain occurrences of both δ_k and δ_{k-1}.
- We then show that a string u of length less than f_k is a cover of $C(F_k)$ if and only if it is a cover of $C(F_{k+1})$; thus, for each value of k, we need concern ourselves only with those proper covers of length at least f_k.

- Finally, we characterize the covers of $C(F_k)$ of length at least f_{k-1}.

This latter result then enables us easily to count all the proper covers of $C(F_k)$.

Lemma 2.4 Every proper cover of $C(F_k)$ is a substring of P_k.

Proof The lemma is trivially true for $k \leq 3$ and true by inspection for $k = 4$. We suppose then that $k \geq 5$ and further that u is a cover of $C(F_k)$, but not a substring of P_k. Hence $u \geq f_k/2$. Since u is not a substring of P_k, one occurrence of u in $C(F_k)$, say u^*, must contain a nonempty prefix of F_k as a suffix (see Fig. 1(b)). (We exclude the case $u = F_k[1..f_k - 1] = F_k[2..f_k]$, clearly an impossibility.) Let j be the starting position of u^*.

(a) Case of u^* containing no occurrence of F_{k-2} (see Fig. 2(a)). Since $F_k = F_{k-2}F_{k-3}F_{k-2}$, it follows that

$$u = u^* = F_{k-2}[j..f_{k-2}]F_{k-2}[1..i],$$

for integers $i \in \{1, ..., f_{k-2} - 1\}$, $j \in \{2, ..., f_{k-2}\}$. But since $F_k = F_{k-2}F_{k-2}F_{k-5}F_{k-4}$, we see that therefore u must be a substring of F_k^2, hence of P_k, a contradiction.

(b) Case of u^* starting at position $f_{k-1} + 1$ (see Fig. 2(b)). In this case u^* contains an occurrence of F_{k-2} and $u^* = F_{k-2}u'$, where u' is a prefix of F_k. But

$$F_k = F_{k-1}F_{k-2} = P_{k-1}\delta_{k-1}F_{k-2},$$

by Lemma 2.1, and so $F_{k-2}F_k = P_k\delta_{k-1}F_{k-2}$. Hence u^* is a prefix of $P_k\delta_{k-1}$ and since, as above, $u \neq F_k[1..f_k - 1]$, we arrive again at the contradiction that u is a substring of P_k.

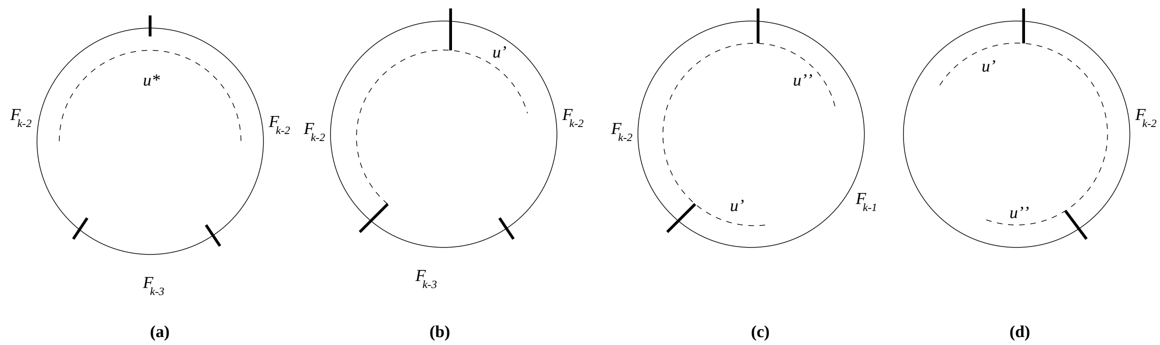

Fig. 2

The circle represents $C(F_k)$, the internal arc represents u^*.

(c) Case of u^* starting at position $j < f_{k-1}$ (see Fig. 2(c)). Then we have $u^* = u'F_{k-2}u''$ for some nonempty u' and $u'' = F_k[1..i]$, for some integer

$i \in \{1, ..., f_{k-1} - |u'| - 2\}$. Observe by Lemma 2.1 that u' has suffix a if k is even, suffix b otherwise. But this case is impossible, since any other occurrence of u, say $\hat{u}$, must take the form (see Lemma 2.3)

$$\hat{u} = u' F_{k-2}^{(h)} u''', \qquad h = 1, 2$$

again by Lemma 2.1, u' has suffix b if k is even, suffix a otherwise.

(d) Case of u^* starting at position $j > f_{k-1}$ (see Fig. 3(d)). Then we have

$$u^* = u' F_{k-2} u'',$$

for nonempty strings u' and u''. But then another occurrence of u must be (see Lemma 2.3)

$$\hat{u} = u' F_{k-2}^{(2)} u''',$$

whose final term u''' contains δ_k in the same position that u'' contains δ_{k-1}. Thus this case also is impossible, and so we conclude that if u is a cover of $C(F_k)$, it must also be a substring of P_k. □

The proof of our first main lemma was lengthy, but it will simplify the proof of the remaining results:

Lemma 2.5 A proper substring u of F_k is a cover of $C(F_k)$ if and only if it is a cover of $C(F_{k+1})$.

Proof We consider the string $C(F_k^2)$ and in particular the occurrences of P_k at positions 1 and $f_{k+1}+1$ of $C(F_k^2)$ (see Fig. 3(a)) :

$$\begin{aligned} P_k &= F_k^2[1..f_k - 2]; \\ P_k &= F_k^2[f_{k+1}+1..f_{2k}]F_k^2[1..f_{k-1}-2]. \end{aligned} \qquad \ldots(2.2)$$

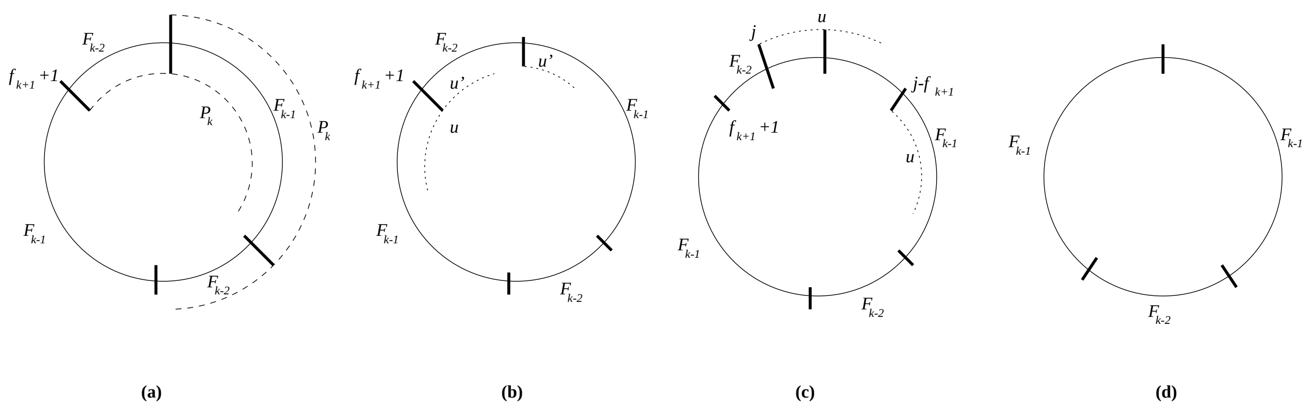

Fig. 3

The circles of (a), (b) and (c) represent the string $C(F_k^2)$.
The circle of (d) represents the string $C(F_{k+1})$.

Suppose first that u is a cover of $C(F_k)$, hence also a cover of $C(F_k^2)$. Note that $C(F_{k+1})$ and $C(F_k^2) = C(F_{k+1}F_{k-2})$ differ only by the suffix F_{k-2} (compare Fig. 3(a) and 3(d)); thus it will suffice to show the following:

(a) If u occurs at position $j \in \{1, ..., f_{k+1}\}$ in $C(F_k^2)$ (see Fig. 3(b)), then u also occurs at the same position in $C(F_{k+1})$. This is trivially true for the occurrences that terminate within F_{k+1}. This is also true for the occurrences that terminate beyond F_{k+1} (see Fig. 3(b)); this follows from the fact that u is shorter than P_k (Lemma 2.4) , and P_k (and thus u', the suffix of u beyond F_{k+1}) occurs at positions 1 and $f_{k+1} + 1$.

(b) If u occurs at position $j \in \{f_{k+1}, ..., 2f_k\}$ in $C(F_k^2)$ (see Fig. 3(c)), then u also occurs at the positions $j - f_{k+1}$ in $C(F_{k+1})$; this follows from the fact P_k occurs at positions 1 and $f_{k+1} + 1$ in $C(F_{k+1})$ (see Fig. 3(c)).

A straightforward reversal of the above argument shows also that it is sufficient. □

We can now complete the picture by characterizing the covers of F_k which are not proper covers of F_{k-1}:

Theorem 2.1 Let u be a cover of F_k such that $f_{k-1} \leq |u| \leq f_k$. Then u is one of the following:

(a) $R_j(F_k)$, for every integer $j = 0, 1, \ldots, f_k - 1$.

(b) $R_j(F_k[1..f_{k-1} + h])$, for every integer $h = 0, 1, \ldots, f_{k-2} - 2$ and every integer $j = 0, 1, \ldots, f_{k-2} - h - 2$;

Proof Note first that (a) is immediate: it merely asserts that every rotation of F_k is a cover of $C(F_k)$. To prove (b), we consider the string $C(F_k^2)$ and in particular the occurrences of P_k at positions $1, f_{k-1} + 1$, and $f_k + 1$ (see Fig. 4).

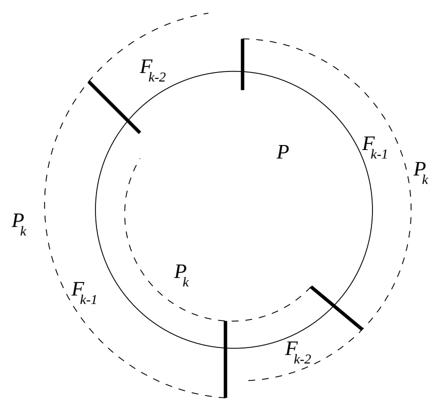

Fig. 4

The circle represents the string $C(F_k^2)$.

One can easily see that every string $P_k[1..i]$ is a cover of $C(F_k)$ for every integer $i \in \{f_{k-1}, ..., f_k - 2\}$. Indeed, it is further clear that every substring of P_k of length i is in fact a cover of $C(F_k)$: these are exactly the strings specified in (b).
$\square$

This result, together with Lemma 2.5, may be used to count the proper covers of $C(F_k)$. From Theorem 2.1(a) we see that the proper covers of lengths $|u| = f_k - 2, f_k - 3, \ldots, f_{k-1}$ may be counted as

$$1 + 2 + \cdots + f_{k-2} - 1 = \binom{f_{k-2}}{2}.$$

Letting ν_k denote the number of proper covers of F_k, Lemma 2.5 then provides the recurrence relation

$$\nu_k = \nu_{k-1} + \binom{f_{k-2}}{2} \qquad \ldots(2.3)$$

with initial condition $\nu_3 = 0$. Solving (2.3) then yields the result that $\nu_k \in \Theta(f_k^2)$:

Theorem 2.2 For every integer $k \geq 4$, the number of proper covers of $C(F_k)$ is given by

$$\nu_k = f_k(f_{k-3} - 1)/2 + (k-1) \bmod 2.$$

$\square$

Finally, we observe that the proper covers of $C(F_k)$ can easily be reported in $\Theta(f_k)$ time by a simple encoding of the output. For example, to specify all the covers described in Theorem 2.1(a), it suffices to give for each length $i = f_{k-1} + h$ the number of rotations of $P_k[1..i]$ that are to be counted as covers. In fact, if it is acceptable to specify only the *range* of i together with the corresponding range of j, then only a constant number of outputs are required for each value of k, and so a total of only $\Theta(\log f_k)$ outputs are necessary.

REFERENCES

[AP83] Alberto Apostolico & F. P. Preparata, **Optimal off-line detection of repetitions in a string**, *TCS 22* (1983) 297-315.

[B86] J. Berstel, **Fibonacci words — a survey**, *Book of L*, Springer-Verlag (1986) 13-27.

[C81] M. Crochemore, **An optimal algorithm for computing the repetitions in a word**, *Inf. Process. Lett. 12-5* (1981) 244-250.

[CS95] L. J. Cummings & W. F. Smyth, **Weak repetitions in strings**, *J. Combinatorial Math. & Combinatorial Computing*, to appear.

[IMP93] Costas S. Iliopoulos, Dennis Moore & Kunsoo Park, **Covering a string**, *Proc. Fourth Annual Symposium on Combinatorial Pattern Matching* (1993) 54-62.

[IMS95] Costas S. Iliopoulos, Dennis Moore & W. F. Smyth, **A characterization of the squares in a Fibonacci string**, submitted for publication.

[IS92] Costas S. Iliopoulos, & W. F. Smyth, **An optimal parallel algorithm for computing the canonical form of a circular string**, *J. of Theoretical Computer-Science 92* (1992) 87-105.

[L81] Aldo de Luca, **A combinatorial property of the Fibonacci words**, *Inf. Process. Lett. 12-4* (1981) 193-195.

[ML84] M. G. Main & R. J. Lorentz, **An $O(n \log n)$ algorithm for finding all repetitions in a string**, *J. Algs. 5* (1984) 422-432.

[MS94] Dennis Moore & W. F. Smyth, **An optimal algorithm to compute all the covers of a string**, *Inf. Process. Lett. 50-5* (1994) 239-246.

[MS95] Dennis Moore & W. F. Smyth, **Correction to: an optimal algorithm to compute all the covers of a string**, *Inf. Process. Lett.* (1995) to appear.

ACKNOWLEDGEMENTS

The work of the first author was supported in part by SERC grants GR/F 00898 and GR/J 17844, NATO Grant No. CRG 900293, ESPRIT BRA Grant No. 7131 for ALCOM II, and MRC Grant No. G 9115730. The work of the third author was supported in part by Grant No. A8180 of the Natural Sciences & Engineering Research Council of Canada and by Grant No. GO-12778 of the Medical Research Council of Canada.