Efficient identification of k-closed strings

Hayam Alamro [1] Mai Alzamel [1] Costas S. Iliopoulos [1] Solon P. Pissis [1] Wing-Kin Sung [2] Steven Watts [1]

EANN 2017

[1]Department of Informatics
King's College London

[2]Department of Computer Science
National University of Singapore

Outline

Background

New Problem

Algorithm

Summary

Background

Closed Strings Background

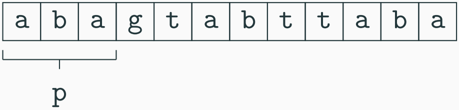
- Closed strings were introduced by Fici [1] as objects of combinatorial interest.
- Closed strings have a relationship with palindromic strings [2].
- Badkobeh et al. [3] factorised a string into a sequence of longest closed factors in time and space $\mathcal{O}(n)$
- Badkobeh et al. [3] computed the longest closed factor starting at every position in a string in $\mathcal{O}(n\frac{\log n}{\log\log n})$ time and $\mathcal{O}(n)$ space.

Prefixes

Definition

A prefix of a string x is a substring p of length m, which occurs at the beginning of x, i.e. at index 0.

$p = x[0\,..\,m-1]$



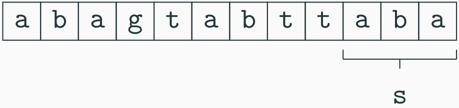
A prefix is called a proper prefix if it does not correspond to the full string x, i.e. $|p| < |x|$.

Suffixes

Definition

A suffix of a string x is a substring s of length m, which occurs at the end of x, i.e. at index $n - m$, where n is the length of x.

$s = x[n - m \mathinner{.\,.} n - 1]$



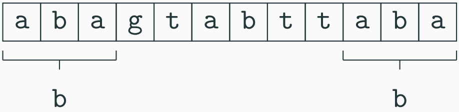
A suffix is called a proper suffix if it does not correspond to the full string x, i.e. $|s| < |x|$.

Bordered Strings

Definition

A bordered string is a string x for which there exists a proper prefix b, which is simultaneously a proper suffix. We call such a b, a border.

$x[0\,..\,b-1] = x[n-b\,..\,n-1]$

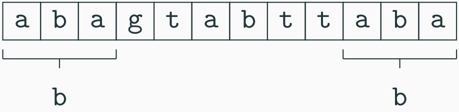


Closed Strings

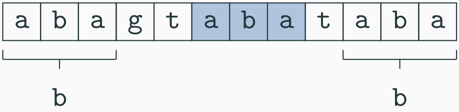
Definition

A closed string is a bordered string x such that some border b of x occurs exactly twice in x. We call such a b, the closed border.

Closed



Non-Closed



New Problem

Goals

- Generalise closed strings to k-closed strings, where k is a measure of approximation.
- Choose a natural definition of k-closed such that: closed $\implies$ 1-closed $\implies$ 2-closed $\implies$ 3-closed $\ldots$
- Develop an efficient algorithm to identify whether or not a string is k-closed.

Approximation Method

Hamming Distance

We use Hamming distance (number of mismatched characters) as a measure of approximation between two strings or factors.

e.g. `agtcta` and `agacga` have Hamming distance 2.

Approximating Closed Strings

Closed String: 2 Conditions

There are 2 conditions that must be satisfied for a string x to be closed, both conditions can potentially be approximated individually or simultaneously by a parameter k:

1. *Border Condition:*
   x has a border b.

2. *No Internal occurrence Condition:*
   x has no internal occurrences of border b.

Closed Definitions with Approximation

Closed (Already Defined)

Border Condition: Exact

No Internal occurrence Condition: Exact

k-**Weakly-Closed**

Border Condition: Approximate

No Internal occurrence Condition: Exact

k-**Strongly-Closed**

Border Condition: Exact

No Internal occurrence Condition: Approximate

k-**Pseudo-Closed**

Border Condition: Approximate

No Internal occurrence Condition: Approximate

k-Weakly-Closed Strings: Definition

Definition

A string x of length n is called k-weakly-closed if and only if $n \leq 1$ or the following properties are satisfied:

1. There exists some proper prefix u of x and some proper suffix v of x of length $|u| = |v|$, such that $\delta_H(u, v) \leq k$.
2. Both factors u and v occur only as a prefix and suffix respectively within x, i.e. no internal occurrences of u or v exist in x.

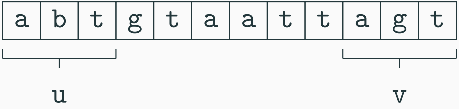
We call such a pair u and v a k-weakly-closed border of x. In the case where $n \leq 1$, we assign ε as the k-weakly-closed border.

k-Weakly-Closed Strings: Example ($k = 1$)

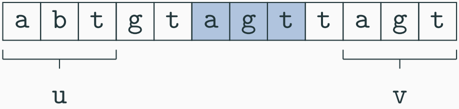
Border Condition: Approximate

No Internal occurrence Condition: Exact

k-Weakly-Closed



Non-*k*-Weakly-Closed



k-Strongly-Closed Strings: Definition

Definition

A string x of length n is called k-strongly-closed if and only if $n \leq 1$ or the following properties are satisfied:

1. There exists some border b of x.
2. There exists no factor w of x of length $|w| = |b|$ such that $\delta_H(b, w) \leq k$, except the prefix and suffix of x.

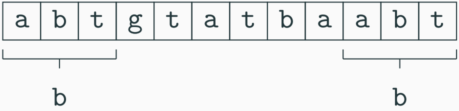
We call b the k-strongly-closed border of x. In the case where $n \leq 1$, we assign ε as the k-strongly-closed border.

k-Strongly-Closed Strings: Example ($k = 1$)

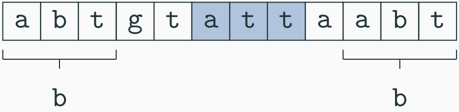
Border Condition: Exact

No Internal occurrence Condition: Approximate

k-Strongly-Closed



Non-*k*-Strongly-Closed



k-Pseudo-Closed Strings: Definition

Definition

A string x of length n is called k-pseudo-closed if and only if $n \leq 1$ or the following properties are satisfied:

1. There exists some proper prefix u of x and some proper suffix v of x of length $|u| = |v|$, such that $\delta_H(u, v) \leq k$.

2. Except for u and v, there exists no factor w of x of length $|w| = |u| = |v|$ such that $\delta_H(u, w) \leq k$ or $\delta_H(v, w) \leq k$.

We call such a pair u and v the k-pseudo-closed border of x. In the case where $n \leq 1$, we assign ε as the k-pseudo-closed border.

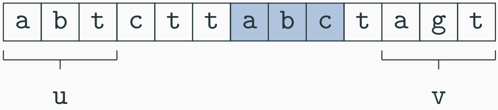
k-Pseudo-Closed Strings: Example ($k = 1$)

Border Condition: Approximate

No Internal occurrence Condition: Approximate

k-Pseudo-Closed

Non-*k*-Pseudo-Closed



k-Closed Strings: Definition

Finally, we define what we mean by a k-closed string:

Definition

A string x of length n is called k-closed if and only if $n \leq 1$ or x is k'-pseudo-closed for some $0 \leq k' \leq k$:

The smallest k' satisfying these conditions, has an associated k'-pseudo-closed border consisting of the pair u and v. We call this pair the k-closed border of x. In the case where $n \leq 1$, we assign ε as the k-pseudo-closed border.

Algorithm

Problem Statement

Problem

Input: A string x of length n and a natural number k, $0 < k < n$
Output: The k-closed border of x or -1 if x is not k-closed

Longest Prefix Match (LPM) and Longest Suffix Match (LSM)

$\mathrm{LPM}_k(x)[j]$ is defined as the length of the longest factor of x starting at index j, which matches the prefix of x of the same length within k errors.

$\mathrm{LSM}_k(x)[j]$ is defined as the length of the longest factor of x ending at index j, which matches the suffix of x of the same length within k errors.

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1

Example for $k = 2$

Longest Common Extension (LCE)

The Longest Common Extension $\mathrm{LCE}(i, j)$ of a string X is defined as the length of the longest factor of X starting at both i and j, i.e. the longest L such that $X[i \mathinner{..} i + L - 1] = X[j \mathinner{..} j + L - 1]$.

If no valid L exists, the LCE equals 0.

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	b	b	a	a	b	a	a	b	a	b	a	b	b	b	a

$$LCE(3, 8) = 3$$

Recursively Generating LPM and LSM

We may compute the $\mathrm{LPM}_{k'+1}$ and $\mathrm{LSM}_{k'+1}$ arrays from the $\mathrm{LPM}_{k'}$ and $\mathrm{LSM}_{k'}$ arrays, such that the arrays are progressively constructed:

$$\mathrm{LPM}_{k'+1}(x)[j] = p + 1 + \mathrm{LCE}(p+1, j+p+1) \text{ of } x$$
$$\mathrm{LSM}_{k'+1}(x)[j] = s + 1 + \mathrm{LCE}(s+1, n-j+s) \text{ of } x^R$$

$$\text{where } p = \mathrm{LPM}_{k'}(x)[j] \text{ and } s = \mathrm{LSM}_{k'}(x)[n-1-j].$$

One iteration of the recursive formula requires $\mathcal{O}(1)$ time for a single index (via standard operations on suffix trees) and thus $\mathcal{O}(n)$ time for the whole array. Therefore, determining $\mathrm{LPM}_{k'}$ and $\mathrm{LSM}_{k'}$ for all $0 \leq k' \leq k$ requires $\mathcal{O}(kn)$ time.

Identifying k-Closed Strings

Once the k LPM's and LSM's are known we can determine if x is k-closed. This is done by finding some j and k' with $1 \le j \le n-1$ and $0 \le k' \le k$ such that all the following 3 conditions are satisifed:

1. $j + \mathrm{LPM}_{k'}(x)[j] = n$
2. $\forall i < j,\ \mathrm{LPM}_{k'}(x)[i] < \mathrm{LPM}_{k'}(x)[j]$
3. $\forall i > n-1-j,\ \mathrm{LSM}_{k'}(x)[i] < \mathrm{LSM}_{k'}(x)[n-1-j]$.

The length of the k-closed border is then $n-j$ for the smallest k' for which there exists a j satisfying the conditions.

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complete Example ($k = 2$)

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$x[j]$	a	b	b	a	b	a	a	b	a	b	a	a	b	a	b
$\mathrm{LPM}_2[j]$	-1	3	4	7	2	10	4	4	7	2	5	4	3	2	1
$\mathrm{LSM}_2[j]$	1	2	3	4	5	2	7	6	2	10	2	5	7	2	-1
Cond 1.	F	F	F	F	F	T	F	F	T	F	T	T	T	T	T
Cond 2.	T	T	T	T	F	T	F	F	F	F	F	F	F	F	F
Cond 3.	T	T	T	F	F	T	F	F	F	F	F	F	F	F	F
2-Closed Border	F	F	F	F	F	T	F	F	F	F	F	F	F	F	F

Complexity Analysis

1. Preprocess x (via a suffix tree)
   to allow for constant time LCE queries.
   $\mathcal{O}(n)$ **time and** $\mathcal{O}(n)$ **space.**
2. Recursively generate $\mathrm{LPM}_{k'}$ and $\mathrm{LSM}_{k'}$ for $0 \leq k' \leq k$.
   k **steps each requiring** $\mathcal{O}(n)$ **time. Total of** $\mathcal{O}(n)$ **space.**
3. During each of the k steps, determine the "peaks" of the LPM and LSM arrays, then verify if the 3 conditions are satisfied for some j where $1 \leq j \leq n - 1$.
   Requires additional $\mathcal{O}(n)$ **time for each of the** k **steps.**

Summary

Summary

- We have generalised closed strings to k-closed strings.
- We have an algorithm that identifies whether a string x is k-closed, and determines the k-closed border, in $\mathcal{O}(kn)$ time and $\mathcal{O}(n)$ space.
- **Further Work:** Improvement in the construction of the LPM and LSM arrays, currently requiring $\mathcal{O}(kn)$ time. Decreasing this time complexity appears to be a reasonable, however non-trivial, goal for any future work on this problem.

Appendix

References

Gabriele Fici
A Classification of Trapezoidal Words
Words 2011, 63:129–137, 2011.

Golnaz Badkobeh and Gabriele Fici and Zsuzsanna Lipták
A Note on Words With the Smallest Number of Closed Factors
CoRR, 1305.6395, 2013.

Golnaz Badkobeh and Hideo Bannai and Keisuke Goto and Tomohiro I and Costas S. Iliopoulos and Shunsuke Inenaga and Simon J. Puglisi and Shiho Sugimoto
Closed factorization
Discrete Applied Mathematics, 212:23–29, 2016.

Thank you for listening ☺