数理解析研究所講究録
第 1769 巻 2011 年 124-130

Morphism preserving some kinds of languages

Tetsuo MORIYA

School of Science and Engineering
Kokushikan University
Setagaya 4-28-1, Setagaya-ku, Tokyo 154-8515, Japan
Electric Mail: moriya@kokushikan.ac.jp

Summary

Morphism preserving some kinds of languages are investigated. We study a necessary and sufficient condition for a morphism to preserve the primitivity, the d-primitivity, the square-freeness, the prefix codes, the suffix codes, and the comma-free codes.

Keywords: morphism, primitive word, d-primitive word, prefix code, suffix code, comma-free code

1 Introduction

The notion of primitive words, d-primitive words, square-free words, and codes play an important role in formal language theory, algebraic coding thery, combinatorial theory of words ([2], [3],[5], [6], [7]). Some studies on a problem concerning the behavior of morphims with respect to primitivity has been done. A sufficient condition for a morphism to preserve the primitivity has been presented in [8], and classification of morphisms from the point of view of their primitivity-preserving properties has been done in [4].

On the other hand, few studies have been done on d-primitive preserving and the preserveness of many kinds of codes, except for prefix codes.

We study a necessary and sufficient condition for a morphism to preserve the primitivity, the d-primitivity, the square-freeness, the prefix codes, the suffix codes, and the comma-free codes.

2 Preliminaries

Let Σ be an alphabet consisting of at least two letters. Σ^* denotes the free moniod generated by Σ, that is, the set of all finite words over Σ, including the empty word ϵ, and $\Sigma^+ = \Sigma^* - \{\epsilon\}$. For w in Σ^*, $|w|$ denotes the length of w. A *language* over Σ is a set $L \subseteq \Sigma^*$.

For a word $u \in \Sigma^+$, if $u = vw$ for some $v, w \in \Sigma^*$, then v (w) is called a *prefix (suffix)* of u, denoted by $v \leq_p u$ $(w \leq_s u$, *resp.*). If $v \leq_p u$ $(w \leq_s u)$ and $u \neq v$ $(w \neq u)$, then v (w) is called a *proper prefix (proper suffix)* of u, denoted by $v <_p u$ $(w <_s u$, resp.). If $u = vxw$ for some $v, w \in \Sigma^*$, x is called an infix or a factor of u. For a word w, let $Pref(w)(Suff(w))$ be the set of all prefixes (suffixes, resp.) of w.

A nonempty word u is called a *primitive word* if $u = f^n$, for some $f \in \Sigma^+$, and some $n \geq 1$ always implies that $n = 1$. Let Q be the set of all primitive words over Σ.

A nonempty word u is a *non-overlapping word* if $u = vx = yv$ for $x, y \in \Sigma^+$ always implies that $v = \epsilon$. Let $D(1)$ be the set of all non-overlapping words over Σ. A words in $D(1)$ is also called a *d-primitive word* (See [1] and [7]).

For a word $w \in \Sigma^+$, there exist a unique primitive word x and a uique integer $i \geq 1$ such that $w = x^i$. Let $x = \sqrt{w}$ and call x *the root* of w.

A word $u \in \Sigma^*$ is square-free if $u = vw^2x$ for some $v, w, x \in \Sigma^*$, implies $w = \epsilon$. Let SF be the set of all square-free words.

A language $L \subseteq \Sigma^+$ is a *code* if $x_1x_2...x_n = y_1y_2...y_m$, $x_i, y_j \in L$ implies $n = m$ and $x_i = y_i$, for $i = 1, ..., n$.

A language $L \subseteq \Sigma^+$ is a *prefix code (suffix code)* if the condition $L \cap L\Sigma^+ = \phi$ ($L \cap \Sigma^+L = \phi$) is true. L is a *bifix code* if L is both a prefix code and also a suffix code. L is an infix code if, for all $x, y, u \in \Sigma^*$, $u \in L$ and $xuy \in L$ together imply $x = y = \epsilon$. L is an *intercode* if $L^{m+1} \cap \Sigma^+L^m\Sigma^+ = \phi$ for some $m \geq 1$. The integer m is called the *index* of L. An intercode of index 1 is called a *comma-free code.*

A language L is a *pure code* if it is a code such that, for any $x \in L^*$, $\sqrt{x} \in L^*$

A mapping $h : \Sigma_1^* \to \Sigma_2^*$ such that $h(xy) = h(x)h(y)$ for all $x, y \in \Sigma_1^*$ is a *morphism* of Σ_1^* into Σ_2^*. A morphism h is *primitive preserving* if $h(x) \in Q$ for all $x \in Q$. A morphism h is *d-primitive preserving* if $h(x) \in D(1)$ for all $x \in D(1)$. A morphism h is *square-free preserving* if $h(x) \in SF$ for all $x \in SF$. A morphism h is *prefix(suffix) preserving* if h preserves the prefix(suffix) codes. A morphism h is *comma-free preserving* if h preserves the comma-free codes.

3 Morphism preserving some kinds of languages

Lemma 1 *([8])*
Let h: $\Sigma^* \to \Sigma^*$ be a morphism. Then h is injective if and only if $h(\Sigma)$ is a code and $|h(\Sigma)| = |\Sigma|$.

□

The "Only if" part of the following proposition is proved in [9]. We give another simpler proof.

Proposition 2 *Let $h : \Sigma^* \to \Sigma^*$ be an injective morphism. Then h is premitive preserving iff $h(\Sigma)$ is a pure code.*

Proof.

[If]([8])

[Only if]By Lemma 1, $h(\Sigma)$ is a code. Let $z = h(w)$ for some $w \in \Sigma^*$, and let $w = p^i$ for some $p \in Q$ and an integer i. We have $h(p) \in Q$ since h is premitive preserving. Hence $\sqrt{z} = h(p) \in h(\Sigma^*) = [h(\Sigma)]^*$. Thus $h(\Sigma)$ is pure. □

Proposition 3 *A morphism $h : \{a,b\}^* \to \{a,b\}^*$ is d-primitive preserving iff $h(u)$ is d-primitive for each d-primitive word u of length ≤ 2.*

Proof.

[Only if] Trivial.

[If] If $h(u)$ is d-primitive for each d-primitive word u of length ≤ 2, then $h(\Sigma)$ is a prefix code and $|h(\Sigma)| = |\Sigma|$. Thus h is injective by Lemma 1. Suppose that that a morphism h is not d-primitive preserving.

There exist $u, v \in \Sigma^+$, and $w \in \Sigma^*$ such that $u \in D(1)$, and $h(u) = vwv$. Let $u = \nu_1\nu_2...\nu_k$ for some $\nu_i \in \Sigma$. If $v = h(\nu_1...\nu_i) = h(\nu_j...\nu_k)$ for some i and j, then $j + i - 1 = k$, and $h(\nu_1) = h(\nu_j), h(\nu_2) = h(\nu_{j+1}), ..., h(\nu_i) = h(\nu_k)$. Since h is injective, $\nu_1 = \nu_j, \nu_2 = \nu_{j+1}, ..., \nu_i = \nu_k$.

This means that $u \notin D(1)$. We have that $v \notin (h(\Sigma))^*$. We can assume that $v = v'w'$ and $v = v"y$ for some $w' \in Pref(h(\Sigma))$ and $y \in h(\Sigma), v', v" \in \Sigma^*$.

If $|w'| \leq |y|$, then either $h(ab) \notin D(1)$ or $h(ba) \notin D(1)$. Thus we can write $w' = xy$ for some $x \in \Sigma^+$. Without loss of generality, we can assume that $h(a) = xyz$ and $h(b) = y$ for some $z \in \Sigma^+$.

(Case 1)$\nu_{k-1} = a$

(Case 1-1) $z = sx$; $|x| < |z|$, $s \in \Sigma^+$.

$h(a) = xyz = xysx$, a contradiction.

(Case 1-2) $x = y"z$; $|z| < |x| < |yz|$, with $y = y'y"$, for $y" \in \Sigma^+$.

$h(a) = y"zyz$, $h(b) = y = y'y"$, contradiction.

(Case 1-3) $x = x"yz$; $|yz| < |x|$, $x = x'x"$, $x" \in \Sigma^+$.

$h(a) = xyz = x"yzyz = x'x"yz$, a contradiction.

(Case 2)$\nu_{k-1} = b$

(Case 2-1) $x = x"yzy^n$ for some $x', x" \in \Sigma^+$ with $x = x'x"$, and $n \geq 1$.

$h(a) = xyz = x"yzy^nyz = x'x"yz$, contradiction.

(Case 2-2) $x = y"y^n$ for some $y", y' \in \Sigma^+$ with $y = y'y"$, and $n \geq 1$. We have that $h(a) = xyz = y"y^nyz$ and $h(b) = y'y"$, a contradiction. □

Proposition 4 *([3])*

Let $h : \Sigma_1^* \to \Sigma_2^*$ be a morphism such that (1)$h(u)$ is squre-free for each square-free word u of length ≤ 3. (2) For a, $b \in \Sigma_1$, $a \neq b$, no $h(a)$ is a proper factor of $h(b)$. Then h is a square-free preserving morphism.

□

The previous Proposition gives a sufficient condition for a morphism to be square-free preserving.

Example 1 *Define a morphism $h : \{a, b, c\}^* \to \{a, b, c, d\}^*$ by $h(a) = abcd, h(b) = b, h(c) = c$. Then h is square-free preserving. Both $h(b)$ and $h(c)$ are proper factor of $h(a)$.*

Proposition 5 *([8]) $h : \Sigma^* \to \Sigma^*$ be a morphism such that $|h(\Sigma)| = |\Sigma|$. Then $h(\Sigma)$ is prefix code iff h preserves the prefix codes.*

Corollary 6 *$h : \Sigma^* \to \Sigma^*$ be a morphism such that $|h(\Sigma)| = |\Sigma|$. Then $h(\Sigma)$ is bifix code iff h preserves the bifix codes*

Proposition 7 *Let $h : \Sigma^* \to \Sigma^*$ be an injective morphism. Then $h(\Sigma)$ is a comma-free code iff h preserves the comma-free codes.*

Proof.
[If] Trivial.
[Only if] Suppose h does not preserve the comma-free codes. For some comma-free code L, $h(L)$ is not a comma-free code. There exist $x, y \in \Sigma^+$, $w, u, v \in L$ such that $xh(w)y = h(u)h(v)$. Let $u = a_1...a_n; v = b_1...b_m; w = c_1...c_k, a_1, ..., a_n; b_1, ..., b_m; c_1, ..., c_k \in \Sigma$. Let $d_1 = a_1, ..., d_n = a_n, d_{n+1} = b_1, ..., d_{n+m} = b_m$. Note that $h(u) = h(a_1)...h(a_n); h(v) = h(b_1)...h(b_m); h(w) = h(c_1)...h(c_k)$.
(Case 1) There exists an integer i such that $h(d_i) = h(c_1), ..., h(d_{i+k-1}) = h(c_k)$. By injectivity of h, L is not comma-free.

(Case 2) For some $i, j \geq 1$, $h(c_1) = h(d_i), ..., h(c_j) = h(d_{i+j-1})$, $h(c_{j+1}) \neq h(d_{i+j})$. Then either $h(c_{j+1}) <_p h(d_{i+j})$ or $h(c_{j+1}) <_s h(d_{i+j})$. Thus $h(\Sigma)$ is not comma-free since it is not a prefix code.

(Case 3) For some integer $i \geq 2$, $h(d_1)...h(d_{i-1}) <_p x <_p h(d_1)...h(d_i)$

(3-1)$h(d_1)...h(d_i) <_p xh(c_1)$

If $h(d_1)...h(d_{i+1}) <_p xh(c_1)$, then $h(c_1)$ has a proper infix $h(d_{i+1})$. If $xh(c_1) <_p h(d_1)...h(d_{i+1})$, then $h(d_i)h(d_{i+1})$ has a proper infix $h(c_1)$. In either case, $h(\Sigma)$ is not comma-free. (3-2) $xh(c_1) <_p h(d_1)...h(d_i)$

$h(d_i)$ has a proper infix $h(c_1)$. Thus $h(\Sigma)$ is not an infix code.

(Case 4) $x <_p h(d_1)$

(4-1)$xh(c_1) <_p h(d_1)$

$h(d_1)$ has a proper infix $h(c_1)$. Thus $h(\Sigma)$ is not an infix code.

(4-2)$h(d_1) <_p xh(c_1)$

If $xh(c_1) <_p h(d_1)h(d_2)$, then $h(d_1)h(d_2)$ has a proper infix $h(c_1)$.

If $h(d_1)h(d_2) <_p xh(c_1)$, then $h(c_1)$ has a has a proper infix $h(d_2)$.

In either case, $h(\Sigma)$ is not comma-free. □

References

[1] Chen-Ming Fan, H.J.Shyr, and S.S.Yu, "d-words and d-languages", Acta Informatica, vol.35, pp.709-727, 1998.

[2] M. Ito and M. Katsura, "Context-free languages consisting of non-primitive words", Int. Journ. of Comp. Math. vol.40, 157-167, 1991.

[3] M. Lothaire, "Combinatorics on words", Addison-Wesley, Reading MA, 1983.

[4] G. Paun and G. Thierrin, "Morphisms and primitivity", Bulletine of EATCS vol.61, pp.85-88,1997.

[5] H. J. Shyr, and G. Thierrin, "Disjunctive languages and codes", in Proc. FCT77, Lecture Notes in Computer Science, vol.56, pp.171-176, Springer, Berlin, 1977.

[6] H. J. Shyr, "Disjunctive languages on a free monoid", Information and Control, vol.34, pp.123-129, 1977.

[7] H. J. Shyr, "Free monoids and languages", Hon Min Book Company, Taichung, Taiwan, 2001

[8] H. J. Shyr, and G. Thierrin, "Codes, languages and MOL schemes", RAIRO. Theor. Computer Sci., vol.1, No.4, pp.293-301, 1977.

[9] V. Mitrana, "Primitive morphisms", Information Processing Letters vol. 64, pp.277-281, 1997