Fundamenta Informaticae 114 (2012) 1–18
DOI 10.3233/FI-2011-608
IOS Press

Pseudopower Avoidance

Ehsan Chiniforooshan[*], **Lila Kari**[†*], **Zhi Xu**[*]

Department of Computer Science

The University of Western Ontario, London, Ontario, Canada N6A 5B7

chiniforooshan@alumni.uwaterloo.ca

lila@csd.uwo.ca

dr.xu.zhi@gmail.com

Abstract. Repetition avoidance has been intensely studied since Thue's work in the early 1900's. In this paper, we consider another type of repetition, called pseudopower, inspired by the Watson-Crick complementarity property of DNA sequences. A DNA single strand can be viewed as a string over the four-letter alphabet $\{A, C, G, T\}$, wherein A is the complement of T, while C is the complement of G. Such a DNA single strand will bind to a reverse complement DNA single strand, called its Watson-Crick complement, to form a helical double-stranded DNA molecule. The Watson-Crick complement of a DNA strand is deducible from, and thus informationally equivalent to, the original strand. We use this fact to generalize the notion of the power of a word by relaxing the meaning of "sameness" to include the image through an *antimorphic involution*, the model of DNA Watson-Crick complementarity. Given a finite alphabet Σ, an antimorphic involution is a function $\theta : \Sigma^* \longrightarrow \Sigma^*$ which is an involution, i.e., θ^2 equals the identity, and an antimorphism, i.e., $\theta(uv) = \theta(v)\theta(u)$, for all $u \in \Sigma^*$. For a positive integer k, we call a word w a *pseudo-kth-power with respect to θ* if it can be written as $w = u_1 \dots u_k$, where for $1 \leq i, j \leq k$ we have either $u_i = u_j$ or $u_i = \theta(u_j)$. The classical kth-power of a word is a special case of a pseudo-kth-power, where all the repeating units are identical. We first classify the alphabets Σ and the antimorphic involutions θ for which there exist arbitrarily long pseudo-kth-power-free words. Then we present efficient algorithms to test whether a finite word w is pseudo-kth-power-free.

Keywords: pseudopower, pseudosquare, pseudocube, antimorphic involution, pattern avoidance

[*]The authors wish to thank Shinnosuke Seki and Lucian Ilie for comments on the draft of this paper. This research was supported by Natural Sciences and Engineering Research Council of Canada Discovery Grant R2824A01, and Canada Research Chair Award to Lila Kari.

[†]Address for correspondence: Department of Computer Science, The University of Western Ontario, London, Ontario, Canada N6A 5B7

1. Introduction

Let Σ be a finite alphabet. The set of finite words and infinite words over Σ are denoted by Σ^* and Σ^ω, respectively. A word v is called a *factor* of x if $x = uvw$ for some words u and w. A nonempty word w is called a *square* (resp., *cube*) if w can be written as $w = uu$ (resp., $w = uuu$) for some $u \in \Sigma^*$. For example, the English word "murmur" is a square. More generally, for an integer $k \geq 2$, a nonempty word w is called a *kth-power* if $w = u^k$ for some $u \in \Sigma^*$. A word w is called *square-free* (resp., *cube-free*, *kth-power-free*) if w does not contain any square (resp., cube, kth-power) as a factor. In the early 1900's, Thue showed in a series of papers examples of square-free infinite words over 3 letters and 4 letters, respectively, and cube-free infinite words over a binary alphabet [22, 23] (see [3] for the English translation). In 1921, Morse [19] independently discovered Thue's construction. In 1957, Leech [16] showed another construction of square-free infinite words, which are generated by a morphism. There are square-free infinite words that cannot be generated by any morphism [2, 25].

Let $|\, w \,|_a$ be the total number of occurrences of the letter a in the word w, and let $|\, w \,| = \sum_{a \in \Sigma} |\, w \,|_a$ be the length of word w for $w \in \Sigma^*$. A nonempty word w is called an *abelian square* if w is in the form $w = u_1 u_2$ such that $|\, u_1 \,|_a = |\, u_2 \,|_a$ for each letter a. For example, the English word "teammate" is an abelian square. Analogously, a nonempty word w is called an *abelian cube* if $w = u_1 u_2 u_3$, where $|\, u_1 \,|_a = |\, u_2 \,|_a = |\, u_3 \,|_a$ for each $a \in \Sigma$, and called an *abelian kth power* if $w = u_1 u_2 \dots u_k$, where k is an integer and $k \geq 2$, $|\, u_i \,|_a = |\, u_j \,|_a$ for $a \in \Sigma$ and $1 \leq i, j \leq k$. A word w is called *abelian-square-free* (resp., *abelian-cube-free*, *abelian-kth-power-free*) if w contains no abelian square (resp., abelian cube, abelian kth power) as a factor. In 1957, Erdös [8] asked whether there exists an abelian-square-free infinite word. Constructions of such words were given by Evdokimov [9] in 1968 over 25 letters, Pleasants [20] in 1970 over 5 letters, and by Keränen [12] in 1992 over 4 letters. Most recently, Keränen [13] presented many new abelian-square-free infinite words. In 1979, Dekking [7] discussed abelian-kth-power-free infinite words for $k \geq 3$.

The discussion on kth-powers is related to molecular biology, and especially nucleic acids research (DNA, RNA). Indeed, repeats of certain segments in human DNA genomic sequences are sometimes indicative of disease [18]. In the same context, another natural notion arises, that of "informational equivalence" between a DNA sequence and its Watson-Crick complement. A DNA single strand is a polymer consisting of a linear arrangement of monomers called nucleotides. There are four different types of nucleotides, adenine, cytosine, guanine and thymine. Mathematically, a DNA single strand can thus be viewed as a string over the four-letter alphabet $\{A, C, G, T\}$. An essential property of DNA strands is their Watson-Crick complementarity: A is complementary to T, and C to G, and two DNA single strands of opposite orientation and whose nucleotides are respectively complementary, bind together by hydrogen bonds to form a double-stranded helical DNA molecule. The reverse complement of a DNA single-strand is called its Watson-Crick complement. Given a DNA single strand and a supply of individual nucleotides, under certain conditions, the enzyme DNA polymerase proceeds to form a new DNA single strand that is the perfect Watson-Crick complement of the original template. A DNA strand and its Watson-Crick complement can even be experimentally obtained from each other, and can thus be viewed as being informationally equivalent.

We model this fact by relaxing the "measure of sameness" of two words to include an antimorphic involution besides the identity function. Given an alphabet Σ, an antimorphic involution is a function $\theta : \Sigma^* \longrightarrow \Sigma^*$ which is an involution, i.e., θ^2 equals the identity, and an antimorphism, i.e., $\theta(uv) = \theta(v)\theta(u)$, for all $u \in \Sigma^*$. Note that, if $\Sigma = \{A, C, G, T\}$ and δ is the antimorphic involution defined

as $\delta(A) = T$, $\delta(C) = G$, then for any string w representing a DNA strand, $\delta(u)$ will represent its Watson-Crick complement.

Other concepts in combinatorics on words have been generalized by replacing the identity function by more general functions, e.g., morphic or antimorphic involutions. They include pseudoprimitive words [5] and pseudopalindromes [6, 11, 1].

Given an antimorphic involution θ of Σ^* and a positive integer k, we call a word w a *pseudo-kth-power with respect to θ* if it can be written as $w = u_1 \dots u_k$, where for $1 \leq i, j \leq k$ we have either $u_i = u_j$ or $u_i = \theta(u_j)$. In this paper, given integers $k \geq 2$, we investigate words that do not contain any pseudo-kth-power as a factor.

In Section 2, we introduce the concept of pseudopower-free words and discuss the existence of such words over different settings of alphabets Σ and antimorphic involutions θ. For example, we show that, for any Σ with $|\,\Sigma\,| \geq 4$ (resp., $|\,\Sigma\,| \geq 3$), and any antimorphic involution θ over Σ, there exist pseudosquare-free (resp., pseudocube-free) infinite words with respect to θ. In Section 3, we discuss algorithms for deciding, given an alphabet Σ, an antimorphic involution θ of Σ^*, a positive integer k, and a word $w \in \Sigma^*$, whether or not the word w is pseudo-kth-power-free. The pseudosquare-freeness and pseudocube-freeness of a word w can be tested in $O(|\,w\,|)$ and $O(|\,w\,|^2)$ time, respectively. For any integer k, $k \geq 2$, the pseudo-kth-power-freeness can be tested in $O(|\,w\,|^2 \log |\,w\,|)$ time, which does not depend on k. Section 4 summarizes our results and presents open problems.

2. Pseudopower-Free Infinite Words

Without loss of generality, unless mentioned explicitly, we always assume the letters are $\mathtt{0}, \mathtt{1}, \mathtt{2}, \dots$. The empty word is denoted by ϵ and the lexicographical order of words is denoted by $<$.

A function $\theta : \Sigma^* \to \Sigma^*$ is called an *involution* if $\theta(\theta(w)) = w$ for all $w \in \Sigma^*$, and called an *antimorphism* (resp., *morphism*) if $\theta(uv) = \theta(v)\theta(u)$ (resp., $\theta(uv) = \theta(u)\theta(v)$). We call θ an *antimorphic involution*, if θ is both an involution and an antimorphism. An antimorphic involution is also called an involutory antimorphism in the literature. For example, the classic Watson-Crick complementarity of DNA strands can be viewed as an antimorphic involution δ over the four-letter alphabet of DNA nucleotides $\{\mathtt{A}, \mathtt{T}, \mathtt{C}, \mathtt{G}\}$. *Reversal* (also called *mirror image*), defined by $(a_1 a_2 \dots a_n)^R = a_n \dots a_2 a_1$, is an antimorphic involution over any given alphabet. We also write $(a_1 a_2 \dots a_n)^R$ as $\mathrm{R}(a_1 a_2 \dots a_n)$ for the convenience of composition of functions. A *transposition* (a, b), $a < b$, is a morphism defined by $(a, b)(a) = b, (a, b)(b) = a$, and $(a, b)(c) = c$ for $c \neq a, c \neq b, a, b, c \in \Sigma$. One can verify that reversal and any transposition commute, i.e., $\mathrm{R} \circ (a, b) = (a, b) \circ \mathrm{R}$, and two disjoint transpositions (a_1, a_2) and (b_1, b_2) commute, i.e., $(a_1, a_2) \circ (b_1, b_2) = (b_1, b_2) \circ (a_1, a_2)$ for $a_i \neq b_j, i, j \in \{1, 2\}$. By definition, every antimorphic involution on an alphabet is a permutation of the letters. Furthermore, we have the following proposition.

Proposition 2.1. Let θ be an antimorphic involution over the alphabet Σ. Then θ can be uniquely written as the composition of transpositions with reversal

$$\theta = (a_0, a_1) \circ (a_2, a_3) \dots (a_{2m-2}, a_{2m-1}) \circ \mathrm{R} \tag{1}$$

up to changing the composition order, where $a_i \neq a_j$ for $i \neq j$ and $m \geq 0$.

Proof:
First we show that $\theta(a)$ is a single letter for any letter a. By definitions, $\theta(w) = \theta(\epsilon w) = \theta(\epsilon)\theta(w)$, so $\theta(\epsilon) = \epsilon$. For any $a \in \Sigma$, since $\theta(\theta(a)) = a$, we have $|\,\theta(a)\,| > 0$ and $|\,\theta(a)\,| < 2$. Hence $|\,\theta(a)\,| = 1$ for any $a \in \Sigma$.

Now we prove the existence of the decomposition in (1) for θ by induction on the size of the alphabet. If $|\,\Sigma\,| = 1$, then $\theta(\mathtt{0}) = \mathtt{0}$ and $\theta(\mathtt{0}^m) = \mathtt{0}^m = \mathrm{R}(\mathtt{0}^m)$. So $\theta = \mathrm{R}$ and (1) holds. If $|\,\Sigma\,| = 2$, then either $\theta(\mathtt{0}) = \mathtt{0}$ or $\theta(\mathtt{0}) = \mathtt{1}$. One can verify that $\theta = \mathrm{R}$ when $\theta(\mathtt{0}) = \mathtt{0}$ and $\theta = (\mathtt{0},\mathtt{1}) \circ \mathrm{R}$ when $\theta(\mathtt{0}) = \mathtt{1}$. Suppose (1) holds for any $|\,\Sigma\,| < n$. For $|\,\Sigma\,| = n \geq 3$, either $\theta(\mathtt{0}) = \mathtt{0}$ or $\theta(\mathtt{0}) = a$ for some $a \neq \mathtt{0}, a \in \Sigma$. If $\theta(\mathtt{0}) = \mathtt{0}$, then by induction hypothesis the restriction of θ on alphabet $\Sigma \setminus \{\mathtt{0}\}$ can be written as $(a_0, a_1) \ldots (a_{2m-2}, a_{2m-1}) \circ \mathrm{R}$. So $\theta = (a_0, a_1) \ldots (a_{2m-2}, a_{2m-1}) \circ \mathrm{R}$ in this case. If $\theta(\mathtt{0}) = a$, then $\theta(a) = \mathtt{0}$. By induction hypothesis, the restriction of θ on alphabet $\Sigma \setminus \{\mathtt{0}, a\}$ can be written as $(a_0, a_1) \ldots (a_{2m-2}, a_{2m-1}) \circ \mathrm{R}$. So $\theta = (\mathtt{0}, a) \circ (a_0, a_1) \ldots (a_{2m-2}, a_{2m-1}) \circ \mathrm{R}$. Therefore, (1) holds.

To show the uniqueness of the form in (1), we first notice that every transposition is the inverse of itself. Assume there is another decomposition $\theta = (b_0, b_1) \circ (b_2, b_3) \ldots (b_{2n-2}, b_{2n-1}) \circ \mathrm{R}$, where $b_i \neq b_j$ for $i \neq j$ and $n \geq 0$. Then $\theta(b_0) = b_1$ and thus there is some $(a_p, a_{p+1}) = (b_0, b_1)$. Since $a_i \neq a_j$ for $i \neq j$, the order of composition in (1) can be arbitrarily changed. Hence we have

$$\begin{aligned}
&(b_2, b_3) \ldots (b_{2n-2}, b_{2n-1}) \circ \mathrm{R} \\
=&(b_0, b_1) \circ \theta \\
=&(a_0, a_1) \ldots (a_{p-2}, a_{p-1}) \circ (a_{p+2}, a_{p+3}) \ldots (a_{2m-2}, a_{2m-1}) \circ \mathrm{R} \ .
\end{aligned}$$

Continuing this procedure, it follows that $\mathrm{R} = (a'_0, a'_1) \ldots (a'_{2m'-2}, a'_{2m'-1}) \circ \mathrm{R}$. By the construction of $\{a_i\}$, however, we know $a'_i \neq a'_j$ for $i \neq j$. So $m' = 0$, since otherwise $\mathrm{R}(a'_0) = a'_1 \neq a'_0$. Therefore, $m = n$ and $\{(b_0, b_1), \ldots, (b_{2n-2}, b_{2n-1})\} = \{(a_0, a_1), \ldots, (a_{2m-2}, a_{2m-1})\}$. □

For any antimorphic involution θ over Σ, we define $\mathrm{Idt}(\theta) = \{a \in \Sigma : a = \theta(a)\}$ to be the set of all letters that remain identical under θ, and define $\mathrm{Trn}(\theta) = \{a \in \Sigma : a < \theta(a)\}$ to be the set consisting of the "smaller" letter, of each pair of letters that are "transposed" into each other by θ. Then θ is fully characterized by $|\,\mathrm{Idt}(\theta)\,|$ and $|\,\mathrm{Trn}(\theta)\,|$ up to an isomorphism. The following proposition follows directly from Proposition 2.1.

Proposition 2.2. Let θ be an antimorphic involution over Σ. Then θ can be written as the composition of $|\,\mathrm{Trn}(\theta)\,|$ distinct transpositions with reversal, and

$$|\,\mathrm{Idt}(\theta)\,| + 2|\,\mathrm{Trn}(\theta)\,| = |\,\Sigma\,| \ . \tag{2}$$

Proof:
By the proof of Proposition 2.1, θ can be written as $\theta = (a_0, a_1) \circ (a_2, a_3) \ldots (a_{2m-2}, a_{2m-1}) \circ \mathrm{R}$, where $a_i \neq a_j$ for $i \neq j$. Then $\mathrm{Idt}(\theta) = \Sigma \setminus \{a_0, a_1, \ldots, a_{2m-1}\}$ and $\mathrm{Trn}(\theta) = \{a_0, a_2, \ldots, a_{2m-2}\}$. So $m = |\,\mathrm{Trn}(\theta)\,|$ and (2) holds. □

For integers k, $k \geq 2$, and antimorphic involution θ, we call a word w a *pseudo-kth-power* (with respect to θ) if w can be written as $w = u_1 u_2 \ldots u_k$, where either $u_i = u_j$ or $u_i = \theta(u_j)$ for $1 \leq i, j \leq k$. In particular, we call a pseudo-2nd-power a *pseudosquare*, and a pseudo-3rd-power a *pseudocube*. For

example, over the alphabet $\{\mathtt{A}, \mathtt{T}, \mathtt{C}, \mathtt{G}\}$ (here we use the conventional symbols instead of $\{0, 1, 2, 3\}$) with respect to the Watson-Crick complementarity (δ: $\mathtt{A} \mapsto \mathtt{T}, \mathtt{T} \mapsto \mathtt{A}, \mathtt{C} \mapsto \mathtt{G}, \mathtt{G} \mapsto \mathtt{C}$), $\mathtt{ACGCGT} = \mathtt{ACG}\theta(\mathtt{ACG})$ is a pseudosquare and $\mathtt{ACGTAC} = \mathtt{AC}\theta(\mathtt{AC})\mathtt{AC}$ is a pseudocube. By definition, a palindrome is a pseudosquare with respect to reversal. A word w is called *pseudo-kth-power-free* (resp., *pseudosquare-free*, *pseudocube-free*), if w cannot be written as $w = uvx$ where v is a pseudo-kth-power (resp., pseudosquare, pseudocube). The first proposition is rather straightforward.

Proposition 2.3. If a word w is pseudo-kth-power-free, then w is kth-power-free. If a word w is abelian-kth-power-free, then w is pseudo-kth-power-free with respect to reversal.

In the remaining part of this section, we will discuss the following problem: Is there a pseudo-kth-power-free infinite word over Σ with respect to θ? The discussion on pseudo-kth-power-free words is related to ordinary kth-power-free words and abelian-kth-power-free words in some cases. By definition, every kth-power is a pseudo-kth-power (with respect to any antimorphic involution on the same alphabet). Every pseudo-kth-power is an abelian kth power with respect to reversal, but a pseudo-kth-power may not be an abelian kth power in general. For example, $\mathtt{ACGT}$ is a pseudosquare with respect to Watson-Crick complementarity δ and is not an abelian square, while 012021 is an abelian square and is not a pseudosquare with respect to any antimorphic involution of 3 letters. We, however, have the following lemmas that reveal part of the relation between the avoidance of different types of repetitions.

Lemma 2.1. Let l be the minimal size of the alphabet over which kth-power-free infinite words exist, let Σ be an alphabet, and let θ be an antimorphic involution over Σ.

(1) If $|\,\Sigma\,| < l$, then no pseudo-kth-power-free infinite word over Σ exists with respect to θ; otherwise

(2) When $|\,\mathrm{Trn}(\theta)\,| \geq l$, then there is a pseudo-kth-power-free infinite word over Σ with respect to θ.

Proof:
(1) Since l is the minimal size of the alphabet over which there is a kth-power-free infinite word and $|\,\Sigma\,| < l$, there is an integer N such that any word of length greater than N over Σ contains a kth-power. Notice that a kth-power is a pseudo-kth-power (with respect to any antimorphic involution). So any word of length greater than N over Σ contains a pseudo-kth-power with respect to θ.

(2) We choose $\Sigma' \subseteq \mathrm{Trn}(\theta)$ such that $|\,\Sigma'\,| = l$. Then there is an infinite word w over Σ' such that w is kth-power-free. Now we claim that w is also pseudo-kth-power-free over Σ with respect to θ. Suppose w contains a pseudo-kth-power. Then $w = xu_1u_2 \dots u_k y$, where either $u_i = u_j$ or $u_i = \theta(u_j)$ for $1 \leq i, j \leq k$. For any $a \in \Sigma'$, by definition, $a < \theta(a)$ and $\theta(\theta(a)) = a < \theta(a)$. So $\theta(a) \notin \Sigma'$ and thus $\theta(u_i)$ is not a word over Σ' for every $1 \leq i \leq k$. Hence $u_i = u_j$ for $1 \leq i, j \leq k$ and thus $u_1u_2 \dots u_k$ is a normal kth-power, which contradicts the fact that w is kth-power-free. Therefore, w is pseudo-kth-power-free with respect to θ. □

Lemma 2.2. Let Σ be a $(2l-1)$-letter alphabet and let θ be an antimorphic involution over Σ. If there is an abelian-kth-power-free infinite word over l letters, then there is a pseudo-kth-power-free infinite word over Σ with respect to θ.

Proof:
By Eq. (2) in Proposition 2.2, it follows that $|\operatorname{Trn}(\theta)| < l$ and thus $|\operatorname{Idt}(\theta)| + |\operatorname{Trn}(\theta)| \geq l$. We choose $\Sigma' \subseteq \operatorname{Idt}(\theta) \cup \operatorname{Trn}(\theta)$ such that $|\Sigma'| = l$. Then there is an infinite word w over Σ' such that w is abelian-kth-power-free. Now we claim that w is also pseudo-kth-power-free over Σ with respect to θ.

Suppose $w = xu_1u_2 \dots u_k y$ contains a pseudo-kth-power, where either $u_i = u_1$ or $u_i = \theta(u_1)$ for $1 \leq i \leq k$. Then either u_1 is a word over $\operatorname{Idt}(\theta)$ or u_1 contains at least one letter from $\operatorname{Trn}(\theta)$. If $u_1 \in \operatorname{Idt}(\theta)^*$, then $\theta(u_1) = \mathrm{R}(u_1)$. So $u_1u_2 \dots u_k$ is an abelian kth power, which contradicts the fact that w is abelian-kth-power-free. Otherwise, u_1 contains at least one letter from $\operatorname{Trn}(\theta)$, say a. Then since $a < \theta(a)$ and $\theta(\theta(a)) = a < \theta(a)$, we have $\theta(a) \notin \Sigma' \subseteq \operatorname{Idt}(\theta) \cup \operatorname{Trn}(\theta)$. So $u_i = u_1$ for $1 \leq i \leq k$, and thus w contains a kth-power, which again contradicts the fact that w is abelian-kth-power-free.

Therefore, w is pseudo-kth-power-free with respect to θ. □

Lemma 2.3. Let θ be an antimorphic involution over Σ and let θ' be an antimorphic involution over Σ' such that $|\operatorname{Trn}(\theta')| \geq |\operatorname{Trn}(\theta)|$ and $|\operatorname{Idt}(\theta')| + |\operatorname{Trn}(\theta')| \geq |\operatorname{Idt}(\theta)| + |\operatorname{Trn}(\theta)|$. If pseudo-$k$th-power-free infinite words exist over Σ with respect to θ, then such words also exist over Σ' with respect to θ'.

Proof:
We choose $\Sigma_1 \subseteq \operatorname{Trn}(\theta')$ such that $|\Sigma_1| = |\operatorname{Trn}(\theta)|$. Since $|\operatorname{Idt}(\theta')| + |\operatorname{Trn}(\theta')| - |\operatorname{Trn}(\theta)| \geq |\operatorname{Idt}(\theta)|$, we can choose $\Sigma_2 \subseteq \operatorname{Idt}(\theta') \cup \operatorname{Trn}(\theta') \setminus \Sigma_1$ such that $|\Sigma_2| = |\operatorname{Idt}(\theta)|$. Define $\Sigma'' = \Sigma_1 \cup \{\theta'(a) : a \in \Sigma_1\} \cup \Sigma_2$, and define antimorphic involution θ'' by $\theta''(a) = a, \theta''(b) = \theta'(b)$ for $a \in \Sigma_2, b \in \Sigma_1 \cup \{\theta'(a) : a \in \Sigma_1\}$. Then $|\Sigma''| = |\Sigma|$, $|\operatorname{Idt}(\theta'')| = |\Sigma_2| = |\operatorname{Idt}(\theta)|$ and $|\operatorname{Trn}(\theta'')| = |\Sigma_1| = |\operatorname{Trn}(\theta)|$. So θ and θ'' are identical up to renaming of the letters. There is a word w over Σ'' such that w is pseudo-kth-power-free with respect to θ''. We claim w is also pseudo-kth-power-free over Σ' with respect to θ'.

Suppose $w = xu_1u_2 \dots u_k y$ contains a pseudo-kth-power, where either $u_i = u_1$ or $u_i = \theta'(u_1)$ for $1 \leq i \leq k$. Then either u_1 is a word over $\Sigma_1 \cup \{\theta'(a) : a \in \Sigma_1\} \cup (\Sigma_2 \cap \operatorname{Idt}(\theta'))$ or u_1 contains at least one letter from $\Sigma_2 \cap (\operatorname{Trn}(\theta') \setminus \Sigma_1)$. In the former case, $\theta''(u_1) = \theta'(u_1)$ and thus w contains a pseudo-kth-power with respect to θ'', which contradicts the fact that w is pseudo-kth-power-free. In the latter case, we assume u_1 contains $a \in \Sigma_2 \cap (\operatorname{Trn}(\theta') \setminus \Sigma_1)$. One can verify that $\theta'(a) \notin \Sigma''$, so $u_i \neq \theta'(u_1)$ for every $1 \leq i \leq k$. Hence w contains a kth-power $u_1u_2 \dots u_k$, which again contradicts the fact that w is pseudo-kth-power-free.

Therefore, w is pseudo-kth-power-free over Σ' with respect to θ'. □

2.1. Pseudosquare-Free Infinite Words

From the general case of pseudo-kth-power-free infinite words, we now focus our attention on the particular case of pseudosquare-free infinite words. Since every binary word of length greater than 3 contains squares, there is no square-free infinite word over 2 letters. By Keränen's construction of abelian-square-free infinite words, there exist pseudosquare-free infinite words over 4 letters with respect to reversal. Furthermore, we have the following result.

Proposition 2.4. For a three-letter alphabet, a pseudosquare-free infinite word exists with respect to reversal, and does not exist with respect to any other antimorphic involution.

Proof:
There are two kinds of antimorphic involutions over 3 letters: θ is either reversal or a transposition composed with reversal.

Suppose θ is reversal. Then a word is pseudosquare-free if and only if it is square-free. Every pseudosquare-free word is square-free. To see the other direction, assume w is square-free. If w contains a pseudosquare of the form u_1u_2 such that $u_1 = \theta(u_2)$, then $u_1u_2 = u_2^R u_2$ contains a square of length 2 in the middle, which is impossible since w is square-free. So w is also pseudosquare-free. Since there are square-free infinite words over a three-letter alphabet, for example the word $w = l^\omega(0)$ where l is a morphism given by Leech [16] as $l(0) = 0121021201210$, $l(1) = 1202102012021$, $l(2) = 2010210120102$, there are pseudosquare-free infinite words with respect to reversal.

Now suppose θ is a transposition composed with reversal. Without loss of generality, we assume $\theta = (0, 1) \circ \mathrm{R}$. We prove that no pseudosquare-free word of length greater than 7 exists in this setting. Suppose w is a pseudosquare-free word of length greater than 7. Since a pseudosquare-free word (with respect to θ) cannot contain $00, 11, 22, 01, 10$, the letter 2's in this word either all appear at odd positions or all appear at even positions. If we omit all symbols 2 in w, the new word w' is over $\{0, 1\}$ and is of length greater than 3. Then w' must contain a square and so does w. So there is no pseudosquare-free infinite word over 3 letters with respect to a transposition composed with reversal. □

We wrote a computer program to find the longest pseudosquare-free word with respect to $\theta = (0, 1) \circ \mathrm{R}$, if any. Starting from the empty word ϵ, if a word is pseudosquare-free, then we extend the word by adding a new letter 0 at the end; otherwise, we do back-tracking and try the next letter. In other words, we do a depth-first-search in a labeled tree (called a *trie*), where each node presents a finite word. Application of similar techniques have been used in the literature to show the non-existence of words of certain type (for example, see [21]). In the case of 3 letters and $\theta = (0, 1) \circ \mathrm{R}$, the tree has 91 nodes, including 61 leaves, and all pseudosquare-free words are enumerated. The tree is of depth 8 and one of the longest pseudosquare-free words is 0212021 of length 7.

Now we discuss the existence of pseudosquare-free infinite words in the "DNA setting", that is over 4 letters with respect to an antimorphic involution θ such that $|\operatorname{Trn}(\theta)| = 2$.

Proposition 2.5. Let $\Sigma = \{1, 2, 3, 4\}$ and θ be an antimorphic involution over Σ such that $\theta(0) = 1, \theta(2) = 3$. Then an infinite word $w \in \Sigma^\omega$ is pseudosquare-free with respect to θ, if and only if $w \in ((0 + 1)(2 + 3))^\omega + ((2 + 3)(0 + 1))^\omega$ and w is square-free.

Proof:
"⇒". Since w is pseudosquare-free, w is square-free. Furthermore, any word in $(0 + 1)^2 + (2 + 3)^2$ is a pseudosquare, so the letters in w must appear alternatively from $\{0, 1\}$ and $\{2, 3\}$. Hence $w \in ((0 + 1)(2 + 3))^\omega + ((2 + 3)(0 + 1))^\omega$.

"⇐". Suppose w contains a pseudosquare. Since w is square-free, it must be the case that $w = ux\theta(x)u$. By the definition of antimorphic involution, the last letter of x and the first letter of $\theta(x)$ are either both from $\{0, 1\}$ or both from $\{2, 3\}$, which contradicts the fact that $w \in ((0 + 1)(2 + 3))^\omega + ((2 + 3)(0 + 1))^\omega$. □

Lemma 2.4. Let θ be an antimorphism over 4 letters with $|\operatorname{Trn}(\theta)| = 2$. A pseudosquare-free infinite word exists with respect to θ.

Table 1. The existence of pseudosquare-free infinite words

$\lvert\Sigma\rvert$	1	2	3	4	5	6	7	8
$\lvert\mathrm{Trn}(\theta)\rvert = 0$	No	No	Yes	Yes	Yes	Yes	Yes	Yes
$\lvert\mathrm{Trn}(\theta)\rvert = 1$	–	No	No	Yes	Yes	Yes	Yes	Yes
$\lvert\mathrm{Trn}(\theta)\rvert = 2$	–	–	–	Yes	Yes	Yes	Yes	Yes
$\lvert\mathrm{Trn}(\theta)\rvert = 3$	–	–	–	–	–	Yes	Yes	Yes
$\lvert\mathrm{Trn}(\theta)\rvert = 4$	–	–	–	–	–	–	–	Yes

Proof:
Without loss of generality, suppose $\theta = (0,1) \circ (2,3)$. Let the morphism f be $f(0) = 02$, $f(1) = 12$, $f(2) = 03$, $f(3) = 13$. We prove that $w = f^\omega(0)$ satisfies the conditions in Proposition 2.5 and thus is pseudosquare-free. First, by noticing $f(a) \in (0+1)(2+3)$ for every letter a, we have $w \in ((0+1)(2+3))^\omega$. Now we show that f preserves square-freeness for some particular type of words. Let w be the shortest square-free word such that w does not contain $01, 10, 23, 32$ and $f(w)$ contains a square. Then there are four cases: (i) $w = uu'$, $f(u) = f(u')$; (ii) $w = uau'$, $a \in \Sigma$, $f(a) = bc$, $f(u)b = cf(u')$; (iii) $w = uu'$, $f(u) = av$, $f(u') = va'$; (iv) $w = uau'$, $a \in \Sigma$, $f(a) = bc$, $f(u) = dcv$, $f(u') = vbd'$. Case (i) is impossible since f is a length-uniform morphism and w is square-free. Both cases (ii) and (iii) are impossible due to the interlacing of the letters $0, 1$ with $1, 2$ in $f(w)$. For case (iv), we have either $w = 1x0x2$ or $w = 0x1x3$ or $w = 2x3x1$ or $w = 3x2x0$, none of which is possible due to the interlacing of the letters in w. So, for a square-free word w that does not contain $01, 10, 23, 32$, the word $f(w)$ is also square-free. Therefore $w = f^\omega(0)$ is pseudosquare-free. □

Theorem 2.1. Let θ be an antimorphic involution over the alphabet Σ. The existence of pseudosquare-free infinite words is as specified in Table 1.

Proof:
(i) Since 3 is the minimal size of alphabet over which there is a square-free infinite word, by Lemma 2.1, there is no pseudosquare-free infinite word over k letters for $k \leq 2$ and there is a pseudosquare-free infinite word with respect to θ with $\lvert\mathrm{Trn}(\theta)\rvert \geq 3$.

(ii) Since there exists an abelian-square-free infinite word over 4 letters, by Lemma 2.2, there is a pseudosquare-free infinite word over 7 or more letters.

(iii) By Proposition 2.4, over 3 letters, there is a pseudosquare-free infinite word with respect to reversal, where $\lvert\mathrm{Idt}(\mathrm{R})\rvert = 3, \lvert\mathrm{Trn}(\mathrm{R})\rvert = 0$, and there is no pseudosquare-free infinite word with respect to other antimorphic involution. So by Lemma 2.3, there is a pseudosquare-free infinite word with respect to θ such that $\lvert\mathrm{Idt}(\theta)\rvert + \lvert\mathrm{Trn}(\theta)\rvert \geq 3$.

(iv) The only case remaining is over 4 letters with $\lvert\mathrm{Trn}(\theta)\rvert = 2$, and by Lemma 2.4 pseudosquare-free infinite words exist.

The results are as summarized in Table 1 and thus the theorem is proved. □

2.2. Other Pseudopower-Free Infinite Words

We now focus our attention on other pseudopower-free infinite words. First, we consider pseudocube-free infinite words. By Dekking's construction of abelian-cube-free infinite words, there exist pseudocube-free infinite words over 3 letters with respect to reversal. The case over the unary alphabet is trivial. For the binary alphabet, we have the following result.

Proposition 2.6. No pseudocube-free infinite word exists over a binary alphabet with respect to any antimorphic involution.

Proof:
There are two kinds of antimorphic involutions over a binary alphabet: we have either $\theta = \mathrm{R}$ or $\theta = (\mathtt{0},\mathtt{1}) \circ \mathrm{R}$.

Suppose $\theta = \mathrm{R}$. We use the computer to find the longest pseudocube-free word, if any, as we did with pseudosquare-free words. Starting from the empty word ϵ, if a word is pseudocube-free, then we extend the word by appending $\mathtt{0}$; otherwise, we do back-tracking and try the next letter. The resulting depth-first-search tree is finite. There are in total 171 nodes, including 86 leaves. The tree is of depth 10 and one of the longest pseudocube-free words is $\mathtt{001101100}$. So there is no pseudocube-free infinite word on this setting.

Suppose $\theta = (\mathtt{0},\mathtt{1}) \circ \mathrm{R}$. Then any word in this setting is a pseudopower and thus one of the longest pseudocube-free words is $\mathtt{00}$. □

Proposition 2.7. There is a pseudocube-free infinite word over 3 letters with respect to each antimorphic involution.

Proof:
There are two kinds of antimorphic involutions over 3 letters: θ is either reversal or a transposition composed with reversal.

Suppose θ is reversal. The following morphism d_3 given by Dekking [7] presents an abelian-cube-free infinite word $d_3^\omega(\mathtt{0})$ over 3 letters, which is therefore also pseudocube-free: $d_3(\mathtt{0}) = \mathtt{0012}$, $d_3(\mathtt{1}) = \mathtt{112}$, $d_3(\mathtt{2}) = \mathtt{022}$. So there is a pseudocube-free infinite word $d_3^\omega(\mathtt{0})$ over 3 letters with respect to reversal.

Now suppose θ is a transposition composed with reversal. Without loss of generality, we assume $\theta = (\mathtt{0},\mathtt{1}) \circ \mathrm{R}$. Consider the following morphism: $t(\mathtt{0}) = \mathtt{021}$, $t(\mathtt{1}) = \mathtt{120}$, $t(\mathtt{2}) = \mathtt{2}$. One can verify that the word $z = t^\omega(0) = \mathtt{021212021202021212020210}\ldots$ is the Thue-Morse sequence [23] with the letter $\mathtt{2}$ inserted between every two consecutive letters. Now we prove that z is pseudocube-free. Suppose $z = xw_1w_2w_3y$ contains a pseudocube $w_1w_2w_3$ with $|\,w_1\,| = |\,w_2\,| = |\,w_3\,|$. Either the last letter of w_1 or the first letter of w_2 is $\mathtt{2}$, but not both. Since $\theta(\mathtt{2}) = \mathtt{2}$, we have $w_1 \neq \theta(w_2)$. So $w_1 = w_2$. By the same reasoning, $w_2 = w_3$. Then the length of $w_1 = w_2 = w_3$ must be even. Otherwise, either the first letter of w_1 or the first letter of w_2 is $\mathtt{2}$, but not both, and thus we have $w_1 \neq w_2$. Now since $|\,w_1\,| = |\,w_2\,| = |\,w_3\,|$ is even, we can omit the letter $\mathtt{2}$ from each word and get new words w_1', w_2', w_3' such that $w_1' = w_2' = w_3'$ and $w_1'w_2'w_3'$ is a factor of the Thue-Morse sequence, which contradicts the fact that the Thue-Morse sequence is cube-free. Therefore, $z = t^\omega(0)$ is pseudocube-free with respect to $(\mathtt{0},\mathtt{1}) \circ \mathrm{R}$ over 3 letters. □

Table 2. The existence of pseudocube-free infinite words

$\lvert\Sigma\rvert$	1	2	3	4	5	6	7	8
$\lvert\mathrm{Trn}(\theta)\rvert = 0$	No	No	Yes	Yes	Yes	Yes	Yes	Yes
$\lvert\mathrm{Trn}(\theta)\rvert = 1$	–	No	Yes	Yes	Yes	Yes	Yes	Yes
$\lvert\mathrm{Trn}(\theta)\rvert = 2$	–	–	–	Yes	Yes	Yes	Yes	Yes
$\lvert\mathrm{Trn}(\theta)\rvert = 3$	–	–	–	–	–	Yes	Yes	Yes
$\lvert\mathrm{Trn}(\theta)\rvert = 4$	–	–	–	–	–	–	–	Yes

Theorem 2.2. Let θ be an antimorphic involution over the alphabet Σ. Then pseudocube-free infinite words over Σ do not exist for $\lvert\Sigma\rvert \leq 2$ and exist for $\lvert\Sigma\rvert \geq 3$, with respect to θ.

Proof:
(i) Since 2 is the minimal size of alphabet over which there is a cube-free infinite word, by Lemma 2.1, there is no pseudocube-free infinite word over k letters for $k \leq 1$ and there is a pseudocube-free infinite word with respect to any θ such that $\lvert\mathrm{Trn}(\theta)\rvert \geq 2$.

(ii) There is an abelian-cube-free infinite word $d_3^\omega(0)$ over 3 letters. By Lemma 2.2, there is a pseudocube-free infinite word over 5 or more letters.

(iii) By Proposition 2.6, there is no pseudocube-free infinite word over a binary alphabet. By Proposition 2.7, there is a pseudocube-free infinite word over 3 letters. In particular, there is a pseudocube-free infinite word over 3 letters with respect to reversal, where $\lvert\mathrm{Idt}(\mathrm{R})\rvert = 3, \lvert\mathrm{Trn}(\mathrm{R})\rvert = 0$. So by Lemma 2.3, there is a pseudocube-free infinite word with respect to θ such that $\lvert\mathrm{Idt}(\theta)\rvert + \lvert\mathrm{Trn}(\theta)\rvert \geq 3$.

The results are summarized in Table 2. □

We now discuss pseudo-kth-power-free infinite words for $k \geq 4$. Every word over a single letter is a power. So the unary case is trivial and no X-free infinite word exists for X either a kth-power, or an abelian kth power, or a pseudo-kth-power. One can verify that the word $d_4^\omega(0)$ is pseudo-kth-power-free for $k \geq 4$ with respect to reversal, where the morphism d_4 is given by Dekking [7] as $d_4(0) = 011$, $d_4(1) = 0001$. By Lemma 2.3, the following theorem holds.

Theorem 2.3. Let θ be an antimorphic involution over the alphabet Σ and let $k \geq 4$ be an integer. Then pseudo-kth-power-free infinite words exist either when $\lvert\Sigma\rvert > 2$ or when $\lvert\Sigma\rvert = 2$ and $\lvert\mathrm{Trn}(\theta)\rvert = 0$.

Proof:
(1) There are no pseudopower-free infinite words over the unary alphabet. Since there is a kth-power-free infinite word over the binary alphabet, by Lemma 2.1, there is a pseudo-kth-power-free infinite word with respect to any θ such that $\lvert\mathrm{Trn}(\theta)\rvert \geq 2$.

(2) There exists an pseudo-4th-power-free infinite word over the binary alphabet, such as the following construction by Dekking [7] $w = d_4^\omega(0)$ where $d_4(0) = 011$, $d_4(1) = 0001$. So there exists an abelian-kth-power-free infinite word w over the binary alphabet for any integer $k \geq 4$. By Lemma 2.2, there is a pseudo-kth-power-free infinite word over 3 or more letters.

(3) That infinite word $w = d_4^\omega(0)$ is also a pseudo-kth-power-free infinite word for $k \geq 4$ over binary alphabet with respect to reversal, where $\lvert\mathrm{Idt}(\mathrm{R})\rvert = 2, \lvert\mathrm{Trn}(\mathrm{R})\rvert = 0$. So by Lemma 2.3, there

Table 3. The existence of pseudo-kth-power-free infinite words for $k \geq 4$

$\lvert \Sigma \rvert$	1	2	3	4	5	6	7	8
$\lvert \mathrm{Trn}(\theta) \rvert = 0$	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
$\lvert \mathrm{Trn}(\theta) \rvert = 1$	–	No	Yes	Yes	Yes	Yes	Yes	Yes
$\lvert \mathrm{Trn}(\theta) \rvert = 2$	–	–	–	Yes	Yes	Yes	Yes	Yes
$\lvert \mathrm{Trn}(\theta) \rvert = 3$	–	–	–	–	–	Yes	Yes	Yes
$\lvert \mathrm{Trn}(\theta) \rvert = 4$	–	–	–	–	–	–	–	Yes

is a pseudo-kth-power-free infinite word for $k \geq 4$ with respect to θ such that $\lvert \mathrm{Idt}(\theta) \rvert + \lvert \mathrm{Trn}(\theta) \rvert \geq 2$. If $\theta = (\mathtt{0}, \mathtt{1}) \circ \mathrm{R}$ over binary alphabet, then any word is a pseudopower.

The results are summarized in Table 3. □

3. Testing Pseudopower-Freeness of Words

Let Σ be an alphabet and let θ be an antimorphic involution over Σ. In this section, we will discuss the following problem: Given a finite word w over Σ and an integer $k \geq 2$, does w contain a pseudo-kth-power with respect to θ as a factor? Section 3.1 discusses the general algorithm for an arbitrary k, and Section 3.2 provides more efficient algorithms for the particular cases of the existence of pseudosquares and pseudocubes. We assume $\lvert w \rvert = N$ in the following discussion.

3.1. Testing Pseudo-kth-Power-Freeness for Arbitrary k

The naïve algorithm to decide whether w contains any pseudo-kth-power as a factor runs in $O(N^3)$ time. The idea is that we check each possible candidate factor u of w to see whether u is a pseudo-kth-power. There are $O(N^2)$ factors, and checking whether a word is a pseudo-kth-power can be done with $O(N)$ comparisons of letters.

Here we consider a more general problem and present an $O(N^2 \log N)$-time algorithm. Let $d : \Sigma^* \times \Sigma^* \to \mathbb{R}$ be a function. Since d will serve as a tool to compare words and we only care about comparing equal-length words, we assume that $d(u, v) = \infty$ for all words u and v with unequal lengths. The FIND-d-POWERS problem is as follows:

Input: A finite word w and two integers $k \geq 2$ and $g \geq 0$.

Output: *All* pairs (s, t) such that $w[s \dots t] = u_1 v_1 u_2 v_2 \dots u_{k-1} v_{k-1} u_k$, $\lvert u_i \rvert = \lvert u_{i+1} \rvert$, $d(u_i, u_{i+1}) = 0$, and $\lvert v_i \rvert = g$, for all $1 \leq i < k$.

In Fig. 1, an output pair (s, t) is shown ($g = 2$). Our original problem of finding all pseudo-kth-power factors is equivalent to FIND-d-POWERS if we set $d(u, v) = \min\{H(u, v), H(u, \theta(v))\}$, where H is the Hamming distance between u and v, and $g = 0$. We show that a simple dynamic programming approach can be used to solve FIND-d-POWERS in $O(N^2 t(N) \lg N)$ time for a large class of functions, called t-breakable.

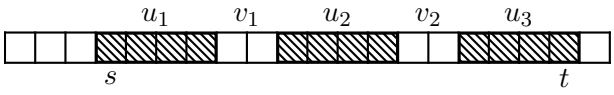

Figure 1. A third-power with gaps of length two ($d(u_1, u_2) = d(u_2, u_3) = 0$).

We first explain our algorithm for $d(u, v) = H(u, v)$; then, we show how it can be expanded to more general distance functions.

Lemma 3.1. Let $g \geq 0$ be an integer, and $w \in \Sigma^*$ be a word. Then there is an algorithm that constructs an $N \times N$ matrix D in time $O(N^2 \lg N)$ such that

$$D_{i,j} = \begin{cases} \infty, & \text{if } i + 2j + g - 1 > N; \\ d(w[i \dots i + j - 1], w[i + j + g \dots i + 2j + g - 1]), & \text{otherwise.} \end{cases}$$

Note that an equivalent description of D is that $D_{i,j}$ is the distance between two subwords of w, both of length j, such that there is a gap of exactly g letters in between them and the first subword starts at position i.

Proof:
In order to compute D, we first construct a number of auxiliary $N \times N$ matrices $D^{(0)}, D^{(1)}, \dots, D^{(\lg N)}$ such that

$$D^{(p)}_{i,j} = \begin{cases} \infty, & \text{if } i + 2^p - 1 > N \text{ or } j + 2^p - 1 > N; \\ d(w[i \dots i + 2^p - 1], w[j \dots j + 2^p - 1]), & \text{otherwise.} \end{cases}$$

Since $D^{(0)}_{i,j} = d(w[i], w[j])$, $D^{(0)}$ can be computed in time $O(N^2)$. Moreover, $D^{(p)}_{i,j}$ can be computed in constant time using values in $D^{(p-1)}$:

$$D^{(p)}_{i,j} \quad = \quad D^{(p-1)}_{i,j} + D^{(p-1)}_{i+2^{p-1}, j+2^{p-1}}. \tag{3}$$

Therefore, constructing all the auxiliary matrices can be done in time $O(N^2 \lg N)$. Since each entry of D can be computed as the sum of at most $\lceil \lg N \rceil$ entries from the auxilary matrices, D can be computed in time $O(N^2 \lg N)$. □

We observe that a pair (s, t) is in the output of FIND-d-POWERS if and only if the column number

$$x = \frac{t - s + 1 - (k - 1)g}{k}$$

in D contains $k - 1$ zero's at rows $s, s + x + g, s + 2x + 2g, \dots, s + (k - 2)x + (k - 2)g$. So, having D computed, it is possible to compute the output of FIND-d-POWERS with only one scan of all entries of D. Note that this does not depend on k; we do an $O(N^2 \lg N)$ time preprocessing, and, then, we can solve FIND-d-POWERS for any given value of k in time $O(N^2)$.

Now, in an exactly similar manner, we can solve FIND-d-POWERS, where d is t-breakable for a non-decreasing function $t : \mathbb{N} \to \mathbb{N}$: we call d *t-breakable* if there are two algorithms BREAK and COMBINE such that for all words $u_1, u_2, v \in \Sigma^*$,

1. BREAK(u_1, u_2, v) is an integer in $[0, |\, v\,|]$.
2. $d(u_1u_2, v_1v_2) = \text{COMBINE}\left(u_1, u_2, v_1, v_2, \begin{bmatrix} d(u_1, v_1), & d(u_1, v_2), \\ d(u_2, v_2), & d(u_2, v_2) \end{bmatrix}\right)$, where $v = v_1v_2$ and $|\, v_1\,| = \text{BREAK}(u_1, u_2, v)$.
3. BREAK and COMBINE run in time $O(t(|\, u_1\,| + |\, u_2\,| + |\, v\,|))$[1].

The only modification is that in (3), we use BREAK and COMBINE to compute $D^{(p)}_{i,j}$ in time $O(t(N))$ from appropriate entries of $D^{(p-1)}$.

Lemma 3.2. For every t-breakable function $d : \Sigma^* \times \Sigma^* \to \mathbb{R}$, there is an $O(N^2t(N) \lg N)$-time algorithm for FIND-d-POWERS. □

Example 3.1. The following functions are **1**-breakable, where **1** is a function mapping every number to 1:

1. the Hamming distance $H(u, v)$;
2. $d(u, v) = H(u, \theta(v))$;
3. $d(u, v) = \begin{cases} 0 & \text{if } u = v, \\ \frac{1}{|\,\mathrm{lcp}(u,v)\,|+1} & \text{otherwise,} \end{cases}$ where $\mathrm{lcp}(u, v)$ is the longest common prefix of u and v;
4. $d(u, v) = \sum_{i=1}^{|\,u\,|} s(u[i], v[i]) + s(u[i], v[i-1])/10 + s(u[i], v[i+1])/10$, where $s : \Sigma^2 \to \mathbb{R}$ is any function assigning penalties to mismatches depending on the letters that mismatch. Roughly speaking, in this case, the distance function d is a "weighted-Hamming distance" function which also takes into account neighbouring positions.

Hamming distance H is a **1**-breakable function, because BREAK$(u_1, u_2, v) = |\, u_1\,|$ and

$$\text{COMBINE}\left(u_1, u_2, v_1, v_2, \begin{bmatrix} H(u_1, v_1) & H(u_1, v_2) \\ H(u_2, v_2) & H(u_2, v_2) \end{bmatrix}\right) = H(u_1, v_1) + H(u_2, v_2)$$

satisfy the above-mentioned three conditions. The discussion of other examples is similar and omitted here.

We call a function d, *pseudo-t-breakable* if there is a constant number of t-breakable functions $d_1, d_2, \ldots, d_m$ and an $O(t(|\, u\,| + |\, v\,|))$-time algorithm A that computes $d(u, v)$ if u, v, $d_1(u, v)$, $d_2(u, v)$, …, $d_m(u, v)$ are given as the input. Clearly, if we compute the matrix D of Lemma 3.1 for functions $d_1, d_2, \ldots, d_m$, we can combine these matrices using A and compute D for d in time $O(N^2t(N))$. So, we have the following lemma.

Lemma 3.3. For every pseudo-t-breakable function $d : \Sigma^* \times \Sigma^* \to \mathbb{R}$, there is an $O(N^2t(N) \lg N)$-time algorithm for FIND-d-POWERS. □

[1]Here we are assuming that BREAK and COMBINE have oracle access to the input words, i.e., they can read the ith letter of an input word and the length of an input word in constant time.

Example 3.2. The following are pseudo-**1**-breakable:

1. any c-breakable function;
2. $d(u,v) = \min\{H(u,v), H(u,\theta(v))\}$;
3. $d(u,v) = 0$, if $\min\{H(u,v), H(u,\theta(v))\} \leq |\,u\,|/10$, and 1 otherwise.

In the particular case $d(u,v) = \min\{H(u,v), H(u,\theta(v))\}$, which is a pseudo-**1**-breakable, we have the following result.

Corollary 3.1. There is an $O(N^2 \lg N)$-time algorithm to enumerate all pseudo-kth-power factors in w.

3.2. Testing Pseudosquare- and Pseudocube-Freeness

In this subsection, we show how to efficiently decide whether or not a word w is pseudosquare-free (resp., pseudocube-free).

A word w is pseudosquare-free with respect to θ if and only if w is square-free (which can be tested in $O(N)$ time [4, 17]) and w does not have factors of the form $a\theta(a)$ for $a \in \Sigma$. Thus, we have the following theorem.

Theorem 3.1. There is a linear-time algorithm to decide whether a word w is pseudosquare-free.

Proof:
A word w contains a pseudosquare if and only if w contains a square or a word of the form $w\theta(w)$.

To check whether w contains a square can be done in linear time. There are a few papers in the literature on testing square-freeness in linear time [4, 17].

To check whether w contains a word of the form $u\theta(u)$, it is enough to check whether w contains a word $a\theta(a)$ for a letter a. To see this, if w contains $u\theta(u)$, then let a be the right-most letter of u and w contains $a\theta(a)$; for the other direction, the word $a\theta(a)$ itself is a pseudosquare.

Algorithm 1: Decide whether w is pseudosquare-free in linear time

Input: a word $w = w[1\,..\,N]$.
Output: "YES" if w is pseudosquare-free; "NO" otherwise.

```
if w contains a square then return "NO";
for i from 1 to N − 1 do
    if θ(w[i]) = w[i + 1] then return "NO";
return "YES";
```

The algorithm is illustrated in Algorithm 1. It is obvious that the algorithm is linear. □

Before we show a quadratic-time algorithm for testing whether a word is pseudocube-free, we first introduce some concepts. Let $w = w[1\,..\,N]$ be a finite word over Σ and let θ be an antimorphic involution over the same alphabet Σ. For a fixed integer k, a *right minimal period array* $\mathrm{rmp}^k_w[1\,..\,N]$ of w is a vector and is defined by

$$\mathrm{rmp}^k_w[i] = \min\left\{\{n : w[i\,..\,i+kn-1] = x^k \text{ for some } x \neq \epsilon\} \cup \{+\infty\}\right\},$$

and similarly a *left minimal period array* $\mathrm{lmp}_w^k[1\,..\,N]$ of w is defined by

$$\mathrm{lmp}_w^k[i] = \min\left\{\{n : w[i-kn+1\,..\,i] = x^k \text{ for some } x \neq \epsilon\} \cup \{+\infty\}\right\} \quad .$$

A *centralized maximal pseudopalindrome array* $\mathrm{cmp}_w[0\,..\,N]$ of w (with respect to θ) is defined by

$$\mathrm{cmp}_w[i] = \max\{\{m : \theta(w[i-m\,..\,i-1]) = w[i\,..\,i+m-1]\} \cup \{0\}\} \quad .$$

For example, when $w = \texttt{01001010}$ and $\theta = \mathrm{R}$, we have $\mathrm{rmp}_w^2 = [3, +\infty, 1, 2, 2, +\infty, +\infty, +\infty]$, $\mathrm{lmp}_w^2 = [+\infty, +\infty, +\infty, 1, +\infty, 3, 2, +\infty]$, and $\mathrm{cmp}_w = [0, 0, 0, 3, 0, 0, 0, 0, 0]$.

Lemma 3.4. [15] For any fixed integer k, the right minimal period array rmp_w^2 of word w can be computed in $O(N)$ time.

There is an algorithm to compute rmp_w^2, the shortest square starting at each position, in linear time [15] by using suffix trees. Since lmp_w^2 can be obtained by first computing $V = \mathrm{rmp}_{\mathrm{R}(w)}^2$ and then reversing V, lmp_w^2 can also be computed in $O(N)$ time.

Lemma 3.5. The array cmp_w of word w can be computed in $O(N)$ time.

Lemma 3.5 in the case $\theta = \mathrm{R}$ has been proved in the book [10, pp. 197–198], and can be generalized to arbitrary antimorphic involutions θ.

Now we are ready to give a quadratic-time algorithm to test whether a word w is pseudocube-free. By definition, a pseudocube is in one of the forms xxx, $xx\theta(x)$, $\theta(x)xx$, and $x\theta(x)x$. We check each of the four cases. A word w has any factor of the form xxx if and only if any maximal repetition [14] in w has exponent ≥ 3, which can be tested in linear time by finding all maximal repetitions. To check whether w contains any word of the form $xx\theta(x)$, we check whether there is a pair of factors $w[i-2n+1\,..\,i] = yy$ and $w[i-m+1\,..\,i+m] = z\theta(z)$ that overlap in the sense that $n \leq m$. By the definitions of lmp_w^2 and cmp_w, we only need to check for each position i whether $\mathrm{lmp}_w^2[i] \leq \mathrm{cmp}_w[i]$. This can be done in $O(N)$ time when all $\mathrm{lmp}_w^2, \mathrm{cmp}_w$ are already computed. The case for $\theta(x)xx$ is similar. To check whether w contains any word of the form $x\theta(x)x$, we check whether there is a pair of factors $w[i-n+1\,..\,i+n] = y\theta(y)$ and $w[j-m+1\,..\,j+m] = z\theta(z)$ that overlap in the sense that $|\,i-j\,| \leq n$ and $|\,i-j\,| \leq m$. By the definition of cmp_w, we check for each pair of indices i, j with $i < j$ whether $j - i \leq \mathrm{cmp}_w[i]$ and $j - i \leq \mathrm{cmp}_w[j]$. This can be done in $O(N^2)$ time when cmp_w is already known. The completed algorithm is given in Algorithm 2. Thus, the following theorem holds.

Theorem 3.2. There is a quadratic-time algorithm to decide whether a word w is pseudocube-free.

Proof:
Algorithm 2 checks the pseudocube-freeness of w in quadratic time. By Lemma 3.4 and Lemma 3.5, the computation of $\mathrm{rmp}_w^2, \mathrm{lmp}_w^2, \mathrm{cmp}_w$ in line 1 can be done in $O(N)$ times. Line 2 can be done in $O(N)$ time. Line 3–8 can be done in $O(N^2)$ time. So the algorithm runs in $O(N^2)$ time.

Now we prove the correctness of the algorithm. First, we prove that if the algorithm returns "NO", then w contains a pseudocube. If the algorithm stops at line 2, then w contains a cube of the form xxx, which is also a pseudocube. Suppose the algorithm stops at line 4. Let $n = \mathrm{rmp}_w[i]^2$ and $m = \mathrm{cmp}_w[i-1]$. Then $n \leq m$ and the word $w[i-n\,..\,i+2n-1]$ is of the form $\theta(x)xx$, which is a

Algorithm 2: Decide whether w is pseudocube-free in $O(N^2)$ time

Input: a word $w = w[1 \,..\, N]$.
Output: "YES" if w is pseudocube-free; "NO" otherwise.

```
1 compute rmp^2_w, lmp^2_w, cmp_w;
2 if w contains a cube then return "NO"; // The case xxx
3 for i from 1 to N do
4     if rmp^2_w[i] ≤ cmp_w[i − 1] then return "NO"; // The case θ(x)xx
5     if lmp^2_w[i] ≤ cmp_w[i] then return "NO"; // The case xxθ(x)
6     for d from 1 to cmp_w[i] do
7         if d ≤ cmp_w[i + d] then return "NO"; // The case xθ(x)x
8 return "YES";
```

pseudocube. Suppose the algorithm stops at line 5. Let $n = \mathrm{lmp}^2_w[i]$ and $m = \mathrm{cmp}_w[i]$. Then $n \le m$ and the word $w[i-2n+1\,..\,i+n]$ is of the form $xx\theta(x)$, which is a pseudocube. Suppose the algorithm stops at line 7. Then the word $w[i-d+1\,..\,i+2d]$ is of the form $x\theta(x)x$, which is a pseudocube.

Now, we prove that if w contains a pseudocube, then the algorithm returns "NO". If w contains a pseudocube of the form xxx, then the algorithm stops at line 2. Suppose $w[s\,..\,s+3p-1] = xx\theta(x)$. Then $q = \mathrm{lmp}^2_w[s+2p-1] \le |\,x\,|$ and $\mathrm{cmp}_w[s+2p-1] \ge |\,x\,| \ge q$. So the algorithm stops at line 5 for $i = s+2p-1$, (although the detected pseudocube $w[s+2p-2q\,..\,s+2p+q-1]$ may be different from $w[s\,..\,s+3p-1]$.) The case $w[s\,..\,s+3p-1] = \theta(x)xx$ is similar. Suppose $w[s\,..\,s+3p-1] = x\theta(x)x$. Then $\mathrm{cmp}_w[s+p-1] \ge |\,x\,|$ and $\mathrm{cmp}_w[s+2p-1] \ge |\,x\,|$. So the algorithm stops at line 7 for $i = s+p-1$ and $d = p$. □

4. Conclusion

In this paper, we discussed the existence of infinite words that do not contain pseudo-kth-powers, in all possible settings for the alphabet Σ and the antimorphic involution θ. No pseudosquare-free infinite word exists for $|\,\Sigma\,| \le 2$, and pseudosquare-free infinite words exist for $|\,\Sigma\,| \ge 4$. For $|\,\Sigma\,| = 3$, the existence of pseudosquare-free infinite words depends on θ. No pseudocube-free infinite word exists for $|\,\Sigma\,| \le 2$ and pseudocube-free infinite words exist for $|\,\Sigma\,| \ge 3$. For any integer $k \ge 4$, pseudo-kth-power-free infinite words exist except when either $|\,\Sigma\,| = 1$, or $|\,\Sigma\,| = 2$ and $|\,\mathrm{Trn}(\theta)\,| = 1$. In the particular DNA setting, $|\,\delta\,| = 4$ and $|\,\mathrm{Trn}(\delta)\,| = 2$, pseudo-$k$th-power-free infinite words exist for any exponent k.

We also proposed algorithms for testing whether or not a word w of length N is pseudo-kth-power-free. For an arbitrary integer $k \ge 2$, we provide an $O(N^2 \lg N)$-time algorithm to find all pseudo-kth-powers in w, where $N = |\,w\,|$. In addition, we provide an $O(N)$-time algorithm and an $O(N^2)$-time algorithm for testing whether w is pseudosquare-free and pseudocube-free, respectively. It is still unknown whether there is faster algorithm for testing whether w is pseudo-kth-power-free.

Lemma 3.4 can be generalized to arbitrary (fractional) powers with exponents $k > 1$ [24] such that the right (resp., left) minimal period array rmp^k_w (resp., lmp^k_w) can be computed in linear time. Thus, by the same techniques in testing pseudocube-freeness, it can be tested in linear time whether or not a word

w contains any pseudo-kth-power of the particular form $xx \dots x\theta(x)$ and $\theta(x)xx \dots x$. In addition, it can be tested in quadratic time whether or not w contains any pseudo-kth-power of the form $x\theta(x)x\theta(x) \dots$. It is also possible that some other particular cases of pseudopowers could be detected faster.

References

[1] Anne, V., Zamboni, L., Zorca, I.: Palindromes and pseudo-palindromes in Episturmian and pseudo-palindromic infinite words, *Publications du LACIM, Proc. of Words 2005* (S. Brlek, C. Reutenauer, Eds.), 36, 2005.

[2] Aršon, S. E.: Démonstration de l'existence des suites asymétriques infinies, *Mat. Sb.*, **(N. S.) 2**(3), 1937, 769–779.

[3] Berstel, J.: *Axel Thue's papers on repetitions in words: a translation*, Number 20 in Publications du Laboratoire de Combinatoire et d'Informatique Mathématique, Université du Québec à Montréal, 1995.

[4] Crochemore, M.: Recherche linéaire d'un carré dans un mot, *Comptes Rendus Acad. Sci. Paris Sér. I*, **296**, 1983, 781–784.

[5] Czeizler, E., Kari, L., Seki, S.: On a special class of primitive words, *Theoret. Comput. Sci.*, **411**(3), 2010, 617–630.

[6] de Luca, A., De Luca, A.: Pseudopalindrome closure operators in free monoids, *Theoret. Comput. Sci.*, **362**, 2006, 282–300.

[7] Dekking, F.: Strongly non-repetitive sequences and progression-free sets, *J. Combin. Theory Ser. A*, **27**(2), 1979, 181–185.

[8] Erdös, P.: Some unsolved problems, *Michigan Math. J.*, **4**(3), 1957, 291–300.

[9] Evdokimov, A. A.: Strongly asymmetric sequences generated by a finite number of symbols, *Dokl. Akad. Nauk. SSSR*, **179**, 1968, 1268–1271.

[10] Gusfield, D.: *Algorithms on strings, trees, and sequences: computer science and computational biology*, Cambridge University Press, 1997.

[11] Kari, L., Mahalingam, K.: Watson-Crick palindromes in DNA computing, *Nat. Comput.*, **9**(2), 2010, 297–316.

[12] Keränen, V.: Abelian squares are avoidable on 4 letters, *Proc. 19th Int'l Conf. on Automata, Lang., and Progr. (ICALP '92)* (W. Kuich, Ed.), 1992.

[13] Keränen, V.: A powerful abelian square-free substitution over 4 letters, *Theoret. Comput. Sci.*, **410**, 2009, 38–40.

[14] Kolpakov, R., Kucherov, G.: Finding maximal repetitions in a word in linear time, *Proc. 40th Ann. Symp. Found. Comput. Sci. (FOCS '99)*, 1999.

[15] Kosaraju, S. R.: Computation of squares in a string, *Proc. 5th Ann. Symp. Combinat. Pattern Matching (CPM 1994)* (M. Crochemore, D. Gusfield, Eds.), 1994.

[16] Leech, J.: A problem on strings of beads, *Math. Gaz.*, **41**(338), 1957, 277–278.

[17] Main, M., Lorentz, R.: Linear time recognition of square free strings, in: *Combinat. Algorithms on Words* (A. Apostolico, Z. Galil, Eds.), Springer, 1985, 272–278.

[18] Mirkin, S.: Expandable DNA repeats and human disease, *Nature*, **447**(7147), 2007, 932–940.

[19] Morse, H.: Recurrent geodesics on a surface of negative curvature, *Trans. Amer. Math. Soc.*, **22**(1), 1921, 84–100.

[20] Pleasants, P.: Non-repetitive sequences, *Math. Proc. Cambridge Philos. Soc.*, **68**(2), 1970, 267–274.

[21] Shallit, J.: Simultaneous avoidance of large squares and fractional powers in infinite binary words, *Int'l. J. Found. Comput. Sci.*, **15**, 2004, 317–327.

[22] Thue, A.: Über unendliche Zeichenreihen, *Norske Vid. Selsk. Skr. I. Mat.-Nat. Kl.*, (7), 1906, 1–22.

[23] Thue, A.: Über die gegenseitige Lage gleicher Teile gewisser Zeichenreihen, *Norske Vid. Selsk. Skr. I. Mat.-Nat. Kl.*, (1), 1912, 1–67.

[24] Xu, Z.: A minimal periods algorithm with applications, *Proc. 21st Ann. Symp. Combinat. Pattern Matching (CPM 2010)* (A. Amir, L. Parida, Eds.), 2010.

[25] Yu, X.: A new solution for Thue's problem, *Inform. Process. Lett.*, **54**, 1995, 187–191.