Complexity of unimodal maps with aperiodic kneading sequences

Nonlinearity **12** (1999) 1151–1176. Printed in the UK PII: S0951-7715(99)88381-9

Complexity of unimodal maps with aperiodic kneading sequences

Yi Wang[†], Ling Yang[†] and Huimin Xie[†][‡][§]

[†] Department of Mathematics, Suzhou University, Suzhou 215006, People's Republic of China
[‡] Institute of Theoretical Physics, Academia Sinica, Beijing 100080, People's Republic of China

E-mail: `hmxian@suda.edu.cn (Huimin Xie)`

Received 13 October 1997, in final form 5 May 1999
Recommended by P Constantin

Abstract. It is well established that a formal language generated from a unimodal map is regular if and only if the map's kneading sequence is either periodic or eventually periodic. A previously proposed conjecture said that if a language generated from a unimodal map is context-free, then it must be regular, i.e. there exists no proper context-free language which can be generated from a unimodal map. This paper is a step forward in answering this conjecture showing that under two situations the conjecture is true. The main results of this paper are: (1) if the kneading map of a unimodal map is unbounded, then the map's language is not context-free, (2) all nonregular languages generated from the Fibonacci substitutions are context-sensitive, but not context-free. These results strongly suggest that the conjecture may be indeed true.

1. Introduction

The task of this paper is to continue the study begun in papers [22, 24], i.e. to explore the complexity of unimodal maps with the tools of symbolic dynamics and formal languages.

Since there exist many books and papers devoted to the study of symbolic dynamics and/or unimodal maps, e.g. [7, 11, 18] and references therein, here we will only briefly introduce the theory of formal languages and its application to the study of dynamical systems, especially to the study of unimodal maps.

The theory of formal languages began with Chomsky's mathematical papers [5, 6], and has developed to become, with the theory of automata and other theories, the foundation of theoretical computer science. There are many good books devoted to this field, and one of the best standard textbook is [14], in which readers can find all general concepts about languages and automata used in this paper. For those readers who would like to know more about formal languages and their applications, the comprehensive reference [19] (published recently in three volumes) is very useful.

An important and interesting point of the theory of formal languages is its wide application in many fields beyond computer science. A few examples are the discrete event systems (DES) in control theory, the analysis of DNA sequences in the science of life, the pattern recognition in image techniques, and the exploration of complexity in dynamical systems.

The applicability of this theory comes firstly from its simple and abstract definition of language that a (formal) language is just a set of finite strings of symbols in which every symbol

[§] Author to whom correspondence should be addressed.

is taken from a finite nonempty set, i.e. the alphabet. Using his generative grammars with different levels of restriction, Chomsky defines four families (or levels) of languages in [6], i.e. regular languages (RGL), context-free languages (CFL), context-sensitive languages (CSL), and recursively enumerable languages (REL). This is the Chomsky hierarchy [14, 19]. This hierarchy, namely, the four families of languages, occupying the central position in the theory of formal languages, is a well-established concept in this field. For instance, a whole chapter in the book [14], i.e. chapter 9, is devoted to this concept with 'The Chomsky Hierarchy' as its title.

The following inclusion relations between the four families (or levels) of Chomsky hierarchy,

$$\mathcal{L}(\mathrm{RGL}) \subsetneq \mathcal{L}(\mathrm{CFL}) \subsetneq \mathcal{L}(\mathrm{CSL}) \subsetneq \mathcal{L}(\mathrm{REL}), \tag{1}$$

are indispensable for understanding this paper. The proofs of (1) can be found in [14]. (In (1) the notation $\mathcal{L}(\mathrm{RGL})$ means the set of all regular languages and so on.)

A very important fact in this theory is the connection of the Chomsky hierarchy with automata. It is rigorously established that the languages of four families mentioned above are recognized (or accepted), respectively, by four classes of automata: finite automata (FA), pushdown automata (PDA), linear bounded automata (LBA), and Turing machines (TM) [14]. We can consider these automata as models of computation, while the Turing machine is the model of modern computers [8].

As we know, Wolfram's paper [23] in 1984 was the first example to use the theory of formal languages to the study of dynamical systems. The topic of [23] is cellular automata, where the appearance of languages is natural. Since then many authors have used this idea to various extents in complexity studies of dynamical systems, including unimodal maps on an interval (e.g. see [1, 2, 9, 10]). Of course, for the case of continuous variable(s), including the unimodal maps, it is necessary to use the method of coarse-graining to obtain infinite symbolic strings, and then to take all finite substrings to obtain the language referred. Hence we can talk about the language of a unimodal map which is the main subject discussed in this paper. Its formal definition will be given in definition 2.1 of section 2.7.

It is natural that the application of formal languages in dynamical systems is often closely associated with the study of symbolic dynamics.

For unimodal maps the first works in this direction are the results contained in [1, 2, 9, 11], among others. Using the method of Markov partition, it was proved or implied in [1, 2, 11] that if the orbit starting from the critical point of a unimodal map consists of only finite points, then the language of this map is regular, i.e. in the lowest level in Chomsky hierarchy. At the same time the Markovian partitions obtained there can be used to provide the (indeterministic) finite automaton required.

It is well known that in the study of symbolic dynamics of unimodal maps the kneading sequence plays a basic role (e.g. see [7]). This is an infinite string reflecting the symbolic behaviour of the orbit starting from the critical point of a unimodal map. A basic fact involved here is that the language of a unimodal map is entirely determined by the map's kneading sequence (see chapter 3 in [27]). (The definition of the kneading sequence for a unimodal map can be found in section 2.6.)

The result mentioned above means that if the kneading sequence of a unimodal map is either periodic or eventually periodic, then the language of this map is regular, providing the orbit of critical point is a finite point set.

Using the methods of formal languages and automata, the last constraint was removed and the reverse proposition was also proved, hence the following theorem holds [22, 24]. (The definition of a unimodal map, the kneading sequence, and the language of a unimodal map are collected in section 2 for the reader's convenience.)

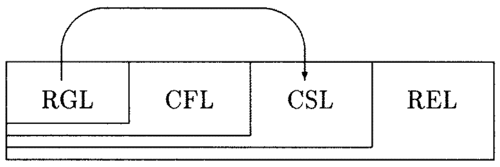

Figure 1. A jump of levels in the Chomsky hierarchy.

Theorem. *The language of a unimodal map is regular if and only if the map's kneading sequence is periodic or eventually periodic.*

For both cases mentioned in this theorem the minimal deterministic finite automata (minDFA) are also constructed in [22, 24]. The number of states in these minDFAs can be used as a measure of complexity for the unimodal maps considered [10, 23].

The next step is to discuss the unimodal maps beyond the family of regular languages. It turns out that there was already a result in this aspect, though it was not well known at that time. This is a result contained in [9]. It is proved that the language generated by a Feigenbaum attractor (e.g. see [21]), i.e. the attractor at the period-doubling accumulation point of a family of unimodal maps, is not context-free, but not beyond the context-sensitive.

Here it should be emphasized that the language defined in [9] is different from those considered in other works mentioned above and in this paper. The language of [9] is obtained from taking all finite substrings from the kneading sequence of a Feigenbaum attractor, hence the discussion of this language can only reflect the symbolic behaviour on the attractor, but not the whole behaviour of the dynamical system, while in other works the languages defined are obtained from all orbits of the systems. This shortcoming was overcome by [4]. A complete proof can be found in section 6.2 of [27]. The conclusion is that the language of a Feigenbaum attractor is not context-free, but a ET0L language, where ET0L is one of Lindenmayer's systems [12], and also a CSL. (Here the relations $\mathcal{L}(\mathrm{CFL}) \subsetneq \mathcal{L}(\mathrm{ET0L}) \subsetneq \mathcal{L}(\mathrm{CSL})$ hold.)

Now we see that every language of a unimodal map with an aperiodic (i.e. neither periodic nor eventually periodic) kneading sequence is not regular, and there is only one language, i.e. the one generated from a Feigenbaum attractor, whose position in the Chomsky hierarchy can be determined at the level of CSLs. We will use figure 1 to indicate that a jump of levels in the Chomsky hierarchy happens at the period-doubling accumulation point of unimodal maps. (In this figure the four nested boxes represent, respectively, the four families (or levels) of the Chomsky hierarchy compatible with the relations (1).) Using the language of paper [8], we can also say that an innovation of computational model classes happens here from the finite automata (FA) jumping to the linear bounded automata (LBA).

Is this phenomenon a rule for languages of unimodal maps? Based on this consideration, a conjecture was proposed in [4, 25, 27] that perhaps there exists no proper CFL that can be generated by unimodal map. It can be stated as follows.

Conjecture. *If the language of a unimodal map is context-free, then it is also regular (or, equivalently, if the language of a unimodal map is not regular, then it is also not context-free).*

Of course, there was only one 'evidence', i.e. the language of a Feigenbaum attractor, which supported this conjecture when it was proposed. This is the situation known about the conjecture before the study of this paper.

This paper is a step forward in answering this conjecture that under two situations the conjecture is true. From the results obtained we believe strongly that, although it is not certain what the final answer is, this conjecture may be indeed true. At least, we can informally say that more than half of this problem has been solved in this paper.

The main results of this paper are:

(1) The language of a unimodal map that has unbounded kneading map is not context-free (theorem 1).

(2) Both odd and even Fibonacci languages are context-sensitive, but not context-free (theorems 2 and 3).

Here some explanation of technical terms used above is needed.

The kneading map is an important tool created by Hofbauer to characterize the intrinsic structure of kneading sequences of unimodal maps. The original reference is [13]. The presentations in [16] (II3b) and [27] (5.1 and 5.2) are also recommended. The kneading map is the main tool to obtain our first main result.

The odd and even Fibonacci languages are two subclasses of languages generated by unimodal maps. They are obtained by Fibonacci substitutions that were proposed by Auerbach and Procaccia [1, 2] and later considered by Hao [11]. A complete mathematical analysis can be found in [26], where it was merely proved that both Fibonacci languages are not regular.

In section 2 we will briefly explain notations, basic notions and tools used in this paper, including the definition of language of a unimodal map, the kneading map and both Fibonacci languages. Some tools from the theory of formal languages are also included.

The main results, i.e., theorems 1–3, are presented in sections 3 and 4 separately. Since the proof of theorem 3 is complicated, some preparatory lemmas which involve study of a special sequence are put into appendices A and B.

2. Preliminaries

2.1. Notations

Here we collect notations used in this paper.

(1) An alphabet is a finite nonempty set of symbols. A string over an alphabet can be either finite or infinite. For convenience sake in this paper we always use the word ‘sequence’ to denote an infinite string, and preserve the word ‘string’ to denote a finite string if no other adjective is put before it. The empty string is denoted by ε.

(2) For the most part of this paper we use the alphabet $S = \{0, 1\}$, where 0 and 1 are symbols, but in the following items the alphabet S is arbitrary, if it is not stated explicitly.

(3) Let S be an alphabet, we use S^* to denote the set of all (finite) strings over S, and S^ω the set of all one-sided infinite strings (sequences) over S.

(4) The length of a string $s = s_1 \ldots s_n$, where each $s_i \in S$, is the number n of the symbols which occur in s and denoted by $|s|$. Of course, the length of the empty string ε is 0.

(5) σ is the shift operator over $S^\omega \cup S^*$: if $s = s_1 s_2 s_3 \ldots$ where each $s_i \in S$, then $\sigma(s) = s_2 s_3 \ldots$.

(6) As a convention, the same notation σ is also used to denote the cyclic shift operator over S^*: if $s = s_1 s_2 \ldots s_n$ for some $n > 0$, then the cyclic shift of s is $\sigma(s) = s_2 \ldots s_n s_1$. Since we mainly use the cyclic shift with the following notation $M(\cdot)$, no confusion will happen.

(7) Let an order relation be introduced between strings and notations $>$, $<$, $=$, $\geqslant$ and $\leqslant$ be used. For a nonempty finite string s its cyclic shift-maximal string is defined by

$$M(s) = \max\{s, \sigma(s), \ldots, \sigma^{|s|-1}(s)\}.$$

If $s = M(s)$ happens, then the string s is called shift-maximal or, simply, maximal.

(8) If y is a nonempty string, then the notation y^n for some positive integer n is the string obtained by concatenating n copies of y, i.e. $y^n = \overbrace{y \dots y}^{n}$. We also use the notation y^∞ to denote the sequence obtained by concatenating copies of y infinitely, i.e. $y^\infty = yy \dots y \dots$.

(9) A nonempty string s is called primitive if $s = y^n$ for some nonempty string y implies $n = 1$ and $s = y$ [20]. Some related facts used in this paper are: if s is primitive, then (1) each cyclic shift of s is also primitive, (2) any two cyclic shifts of s among $s, \sigma(s), \dots, \sigma^{|s|-1}(s)$ are not equal, (3) if s is also maximal, then its cyclic shift $\sigma^i(s) < s$ holds for each $0 < i < |s|$.

(10) If s is a nonempty string, then $s\pi$ is the string obtained from s by removing its last symbol. For instance, if $s = 10110$, then $s\pi = 1011$.

(11) For $S = \{0, 1\}$ two notations are defined as follows. If s is a nonempty string, then $\bar{s}$ is the string obtained from s by changing its last symbol, i.e. from 0 to 1 and vice versa. For instance, if $s = 10110$, then $\bar{s} = 10111$. In a similar way the notation $\underline{s}$ is the string from s by changing its first symbol. For $s = 10110$ we have $\underline{s} = 00110$.

2.2. Unimodal maps

There are several different definitions of unimodal maps used in the related literature. It can be shown that the selection has no influence on the study in this paper. Since this discussion of equivalence is a lengthy and trivial work we will omit it and proceed as follows. Without loss of generality, in this paper a unimodal map is a function f defined on the interval $I = [0, 1]$ and satisfies the following conditions:

(1) f is continuous,

(2) f reaches its maximal value $f(c)$ at some interior point c of the interval I, and is strictly increasing on $[0, c)$, and strictly decreasing on $(c, 1]$,

(3) $f(c) \leqslant 1$ and $f(0) = f(1) = 0$.

The point $c \in (0, 1)$ is called the critical point of the unimodal map f.

The most familiar family of unimodal maps is the logistic maps (or quadratic maps). Its formulation compatible with our definition of unimodal maps is as follows:

$$f(x) = bx(1 - x), \qquad 0 \leqslant x \leqslant 1, \qquad 0 < b \leqslant 4. \tag{2}$$

By iterations of a unimodal map we obtain a real one-dimensional dynamical system. More exactly, starting from a given point $x \in I$ and through iterations, we obtain the orbit of dynamical system as follows:

$$f^*(x) = (x, f(x), f^2(x), \dots, f^n(x), \dots),$$

where we use notation $f^2(x) = f(f(x))$ and so on.

It is obvious that the interesting behaviour of this dynamical system can only happen in the smaller subinterval $[0, f(c)]$ or even $[f^2(c), f(c)]$ (under the condition $f^2(c) < f^3(c)$). The object of our study hereafter will focus on all orbits in $[0, f(c)]$. The extension of results to $[0, 1]$ or $[f^2(c), f(c)]$ are trivial and are omitted.

2.3. Symbolic dynamics of unimodal maps

Using a coarse-grained description we can translate every orbit into a symbolic sequence as follows. First let

$$A(x) = \begin{cases} 0, & \text{when} \quad x \in [0, c), \\ c, & \text{when} \quad x = c, \\ 1, & \text{when} \quad x \in (c, 1], \end{cases}$$

and then define

$$A(f^*(x)) = A(x)A(f(x))A(f^2(x)) \dots A(f^n(x)) \dots .$$

This is an infinite symbolic sequence over the alphabet $S' = \{0, c, 1\}$, which contains 0, c, 1 as symbols, not numbers. Note that we use the same letter c to denote both the critical point of a unimodal map and the symbol appearing in $A(f^*(x))$. We call this sequence the itinerary of point x, and denote it by $I(x)$. The set of all itineraries $\{I(x) | x \in I\}$ for a given unimodal map f is the object of the study of symbolic dynamics.

2.4. A shift operator

As presented in item (5) of section 2.1, a shift operator σ is defined over strings and sequences for any alphabet. Here the alphabet is $S' = \{0, c, 1\}$.

If f is a unimodal map, and s is the itinerary of a point $x \in I$, i.e. $s = I(x)$, then it is easy to verify the following basic property of σ that

$$\sigma(I(x)) = I(f(x)) \qquad \text{for each} \quad x \in I, \tag{3}$$

i.e. $\sigma \circ I = I \circ f$.

2.5. An order relation associated with unimodal maps

The order relation which we will introduce between itineraries is a reflection of the natural order between points of the interval $I = [0, 1]$. The notations $<$, $>$, $\leqslant$, $\geqslant$, and $=$ are used to denote the relation. First we define the order relation between symbols in $S' = \{0, c, 1\}$ by $0 < c < 1$.

Next we need the notion of even string and odd string as follows: a nonempty string $s_1 \dots s_n$ consisting entirely of symbols 0 and 1 is called even (odd) if the number of $1'$s appearing in $s_1 \dots s_n$ is even (odd).

Let the two itineraries be

$$s = s_1 s_2 \dots s_n \dots \qquad \text{and} \quad t = t_1 t_2 \dots t_n \dots .$$

If $s_i = t_i$ for all $i \geqslant 1$, then define $s = t$. Otherwise, compare $s_i = t_i$ starting from $i = 1$. If $s_1 \neq t_1$, then define $s < t$ (or $s > t$) if and only if $s_1 < t_1$ (or $s_1 > t_1$). If $s_i = t_i$ for $i = 1, \dots, n$ for some n but $s_{n+1} \neq t_{n+1}$, then the string $s_1 \dots s_n (= t_1 \dots t_n)$ is referred to as the longest common prefix of s and t. It is easy to know that there is no symbol c in this prefix, otherwise s would equal to t. Now we define $s < t$ if either $s_1 \dots s_n$ is even and $s_{n+1} < t_{n+1}$ or $s_1 \dots s_n$ is odd and $s_{n+1} > t_{n+1}$, and define $s > t$ in similar way.

It is easy to establish that

$$x < y \Longrightarrow I(x) \leqslant I(y), \tag{4}$$

hence this order relation between itineraries preserves the natural order relation in the interval I in a weak sense.

This order relation can be extended beyond itineraries if the sequences compared satisfy one of two conditions: (1) there is no symbol c in these sequences; (2) if there is a symbol c appearing in a sequence, then the suffix after it coincides with the itinerary $I(f(c))$. We also use this order relation to compare two strings, or a string and a sequence, as long as those strings are taken from the sequences mentioned above.

2.6. Kneading sequence

Among all possible itineraries of a unimodal map f, there exists a distinct itinerary called the kneading sequence of f, i.e. the itinerary of point $f(c)$, and the following special notation is used in this paper:

$$KS = I(f(c)) = e_1e_2\ldots e_n\ldots. \tag{5}$$

This notion of the kneading sequence is indispensable in the study of symbolic dynamics of unimodal maps [7].

For a given unimodal map f, the basic property of its kneading sequence KS is

$$\sigma^i(I(x)) \leqslant KS \qquad \text{for every } i \geqslant 0 \text{ and for every point } x \in [0, f(c)]. \tag{6}$$

This can be seen from (4) and (3), as $\sigma^i(I(x)) = I(f^i(x))$ and $0 \leqslant f^i(x) \leqslant f(c)$ hold.

Using this result to $KS = I(f(c))$ itself, we have

$$\sigma^i(KS) = e_{i+1}e_{i+2}\ldots \leqslant KS \qquad \text{for every } i \geqslant 0. \tag{7}$$

By this property we call KS a shift-maximal sequence.

An important fact about the kneading sequence is: if s is either a sequence over $S = \{0, 1\}$ or $s = (tc)^\infty$ for some nonempty $t \in S$, then the necessary and sufficient condition for s being the kneading sequence of a unimodal map is that s is a shift-maximal sequence. We have already shown the necessary part of this fact. The proof of sufficient part can be found in [7]. As a matter of fact, each possible kneading sequence can be realized in the family of logistic maps (2).

2.7. Language of unimodal map

Instead of considering all itineraries of a unimodal map f, we can take all finite substrings from all possible itineraries. Thus we obtain a language which is generated from f through two steps: (1) the coarse-grained description of orbits of a unimodal map by iterations, (2) taking all finite substrings from all possible itineraries. But in doing the step (2) we add two constraints. First we only take those itineraries $I(x)$ in which the point $x \in [0, f(c)]$. Next we discard all those substrings that contain the symbol c.

It can be proved that the language defined verbally above is exactly defined mathematically as follows.

Definition 2.1. *Let KS be the kneading sequence of a unimodal map f, then the language L generated from f, or simply, the language of unimodal f, is defined by*

$$L = \mathcal{L}(KS) = \{s \in S^* | \sigma^i(s) \leqslant KS \quad \textit{for } i = 0, 1, \ldots, |s| - 1\}. \tag{8}$$

By definition this language is a subset of S^*, where $S = \{0, 1\}$.

Is this language an appropriate selection? Since KS is shift-maximal, by (7) each of its substrings belong to L, provided it contains no c. Using (6) it is also clear that each substring of an itinerary $I(x)$ ($x \in [0, f(c)]$) belongs to L as long as this substring do not contain c. For instance, every string 0^n ($n > 0$) belongs to L for any unimodal map. This fact coincides with our definition of unimodal map f, as 0 is always a fixed point of f.

On the other hand, it can be proved that for each string $t \in L$, there exists a point $x \in [0, f(c)]$ such that the string t is a substring of the itinerary $I(x)$. The proof of this basic fact can be found in [27] (section 3.3.3: Each Word is 'Real').

From the definition of the language we see that although this language is generated from the unimodal map f, it is completely determined by the kneading sequence KS of f, hence in the notation $L = \mathcal{L}(KS)$ the unimodal map f does not appear explicitly.

It should be noted that each language L of a unimodal map has two distinctive properties: (1) L is factorial that if $x \in L$ then every substring of x also belongs to L, (2) L is extensible that if $x \in L$, then there exist $a, b \in S = \{0, 1\}$ such that $axb \in L$. (From definition 2.1 it is clear that the selection of $a = 0$ is always possible.) The factorial property is frequently used in sections 3 and 4. There exists an important one-to-one relation between this kind of language and symbolic flows (see, e.g., section 2.3 of [27]).

2.8. Fibonacci languages

As stated in the introduction the language of a unimodal map is regular if and only if the map's kneading sequence is either periodic or eventually periodic. Since the regular language is in the lowest level of Chomsky hierarchy (see (1)), the problem of finding more complicated languages of unimodal maps is a problem which should be be studied. One method found in [1, 2, 11] is to obtain new kneading sequence through the Fibonacci substitution. The example in [2] (p 6607) is as follows.

Example 1. *Using the substitution rule*

$$h = (101 \to 10110111, 11 \to 10111),$$

we can obtain a sequence

$$s = h^{\infty}(101) = \lim_{n\to\infty} h^n(101) = 10110111, 10110111, 10111, \ldots,$$

which is the kneading sequence of the logistic map (2) at the parameter value $b = 3.800\,284\,948\ldots.$

Remark. *The above example is obtained by rewriting the original one in [2]. There are several reasons to do so: (1) the definition of kneading sequence in [2] is different from ours, and equals $\sigma(KS)$ in our notation, (2) the formulation of the logistic maps in [2] is $f(x) = 1 - ax^2$, not the one of (2), hence we have to recalculate the parameter value b from the value $a = 1.710\,3989\ldots$ of [2]. (3) The original substitution rule can be simplified.*

In [11] it was pointed out that the same result can be obtained through two operations as follows: first starting from

$$t_0 = 0, \qquad t_1 = 11, \qquad \text{and} \qquad t_n = t_{n-1}t_{n-2} \qquad \text{for} \quad n \geqslant 2$$

to obtain all t_n, then rotate t_n cyclically to obtain the cyclic shift-maximal string $m_n = M(t_n)$ for all n. The limit of $\{m_n\}_{n\geqslant 0}$ is the kneading sequence required. Four examples are studied there, and for these four kneading sequences, the parameter values for the logistic maps were calculated by the method of word-lifting technique (section 2.3 of [11]).

The name 'Fibonacci' comes from the lengths of $\{t_n\}$:

$$|t_0| = 1, |t_1| = 2, \ldots, |t_n| = |t_{n-1}| + |t_{n-2}| \qquad \text{for} \quad n \geqslant 2,$$

hence the Fibonacci numbers 1, 2, 3, 5, 8, ...

In [26] this approach is analysed completely. It was found that there are exactly two nontrivial kinds of languages that can be obtained this way. The notions of odd and even

Fibonacci languages are defined there, and it was proved that both of them are not regular. In this paper we will discuss their positions in Chomsky hierarchy further. The formal definitions of two Fibonacci languages are as follows.

Definition 2.2. *Let the substitution rule h be defined by*

$$h = (\alpha \to \alpha\beta\alpha\alpha\beta, \beta \to \alpha\alpha\beta), \tag{9}$$

where $\alpha, \beta \in S^*$ satisfy the following conditions: (1) α is odd and β is even, (2) α, $\alpha\beta$ and $\alpha\beta\alpha$ are primitive maximal, (3) $\alpha > \beta$.

The odd Fibonacci kneading sequence KS is obtained by

$$KS = h^\infty(\alpha) = \lim_{n\to\infty} h^n(\alpha),$$

and the odd Fibonacci language is $\mathcal{L}(KS)$.

Definition 2.3. *Let the substitution rule g be defined by*

$$g = (\alpha \to \alpha\alpha\beta, \beta \to \alpha\beta), \tag{10}$$

where $\alpha, \beta \in S^*$ satisfy the following conditions: (1) both α, β are even, (2) $\alpha, \alpha\beta$ and $\alpha\alpha\beta$ are primitive maximal, (3) $\alpha > \beta$.

The even Fibonacci kneading sequence KS is obtained by

$$KS = g^\infty(\alpha) = \lim_{n\to\infty} g^n(\alpha),$$

and the even Fibonacci language is $\mathcal{L}(KS)$.

Out of the four examples in [11], three examples are of odd Fibonacci languages, and one example, i.e. example 1, is of even Fibonacci language. The substitution rule (1) is obtained from (10) by taking $\alpha = 101$ and $\beta = 11$.

2.9. Kneading map

Hofbauer's kneading map characterizes the intrinsic structure of every kneading sequence of unimodal maps [13]. It is our basic tool in section 3. Here we merely present what the kneading map is, list some properties of this map as needed in this paper, and give some examples related to our study.

Let the kneading sequence of a given unimodal map f be

$$KS = e_1 e_2 \ldots e_n \ldots,$$

in which each $e_i \in S = \{0, 1\}$, and N be the set of all positive integers. It is proved in [13] that if KS is nonperiodic, then there exists a map Q from N to $\{0\} \cup N$, such that the kneading sequence KS can be rewritten (or decomposed) in the form

$$KS = B_0 B_1 \ldots B_n \ldots, \tag{11}$$

where each B_i is a nonempty string and recursively defined by

$$B_0 = 1, \qquad \text{and} \qquad B_n = B_0 B_1 \ldots \bar{B}_{Q(n)} \qquad \text{for} \quad n \geqslant 1, \tag{12}$$

through the map Q. We call Q the kneading map associated with the kneading sequence KS (or the unimodal map f).

Some basic properties of Q are as follows:

(1) Since every nonperiodic kneading sequence must begin from ten, it can be seen from (12) that $B_1 = \bar{B}_0 = 0$, and hence $Q(1) = 0$ is always true.
(2) $Q(n) < n$ for each $n > 0$.

(3) Q is bounded if and only if $\{|B_n|\}_{n\geqslant 0}$ is bounded. This can be seen from the following inequalities

$$1 + Q(n) \leqslant |B_n| = |B_0| + |B_1| + \cdots |B_{Q(n)}| \qquad \text{for} \quad n > 1,$$

which is a direct consequence of (12).

(4) A prefix of KS, say s, is odd primitive and maximal if and only if $s = B_0 \ldots B_n$ for some $n \geqslant 0$ [27].

(5) If Q is unbounded, then the kneading sequence is aperiodic, but the reverse is not true. (This fact is a consequence of lemma 5.1.8 of [27].)

For a given kneading sequence KS the rule to calculate Q is implied in (11) and (12). Roughly speaking, if we already have $B_0, B_1, \ldots, B_{n-1}$, and write $KS = B_0B_1 \cdots B_{n-1}t$, where t is an infinite suffix of KS, then compare KS and t until they are different, i.e. obtain a nonempty prefix u of KS such that

$$KS = u \ldots \qquad \text{and} \qquad t = \bar{u} \ldots,$$

and we obtain $B_n = \bar{u}$. Again, compare B_n with $KS = B_0B_1 \ldots B_{n-1} \ldots$ to determine the value of $Q(n)$. The property $Q(n) < n$ ensures the success of this process.

Some examples about Q are as follows.

Example 2. *For $KS = 101^\infty$, the kneading sequence of $2 \to 1$ band-merging point of unimodal maps, we can decompose this sequence by*

$$101^\infty = 1, 0, 11, 11, \ldots$$

where for clarity we insert commas to separate B_n. Here we have $B_0 = 1$, $B_1 = 0$, and $B_n = 11 = \overline{10} = B_0\bar{B}_1$ for each $n > 1$, hence

$$Q(1) = 0, \qquad Q(n) = 1 \qquad \textit{for each } n > 1.$$

This is an example of a bounded kneading map. It is easy to prove that for each eventually periodic (but not periodic) kneading sequence its associated kneading map is bounded.

Example 3. *A bounded kneading map can associate a much more complicated kneading sequence than those in example 2. As an example, we can decompose the kneading sequence of example 1 as follows:*

$$KS = 1, 0, 11, 0, 11, 11, 0, 11, 0, 11, 11, 0, 11, 1 \ldots,$$

and hence obtain

$$Q(1) = Q(3) = Q(6) = Q(8) = Q(11) = 0,$$
$$Q(2) = Q(4) = Q(5) = Q(7) = Q(9) = Q(10) = Q(12) = 1, \ldots.$$

The proof of the fact that this kneading map Q really takes only two values, 0 and 1, is simple and omitted.

Example 4. *The kneading sequence of a Feigenbaum attractor can be obtained from the substitution rule $h = (1 \to 10, 0 \to 11)$ as follows [18]:*

$$KS = h^\infty(1) = \lim_{n\to\infty} h^n(1) = 1, 0, 11, 1010, 10111011, \ldots,$$

and here $Q(n) = n - 1$ holds for each $n \geqslant 1$. This is an example of unbounded kneading map.

Example 5. *The kneading sequence of Fibonacci map discussed in [15,16] can be defined by its associated kneading map*

$$Q(1) = 0, \qquad Q(n) = n - 2 \qquad \textit{for each} \quad n \geqslant 2,$$

thus we obtain

$$KS = 1, 0, 0, 11, 101, 10010, 10011100, 1001110110011, \ldots.$$

This is also an example of unbounded kneading map.

2.10. Tools from formal languages

Two necessary conditions for a language being context-free are basic tools for our purpose (see [3, 17]).

Lemma 2.4 (pumping lemma for CFL). *For each context-free language L there exists a positive integer $k(L)$ such that if z is a word in L and $|z| > k(L)$, then we can write $z = uvwxy$ such that:*

(i) $uv^iwx^iy \in L$ *for all* $i \geqslant 0$,
(ii) $|vx| > 0$, *and*
(iii) $|vwx| \leqslant k(L)$.

Lemma 2.5 (Ogden lemma). *For each CFL L there exists a positive integer $k(L)$ such that if z is a word in L, and if any k or more distinct positions in z are designated as distinguished, then we can write $z = uvwxy$ such that:*

(i) $uv^iwx^iy \in L$ *for each* $i \geqslant 0$,
(ii) *w contains at least one of the distinguished positions,*
(iii) *either u and v both contain distinguished positions, or x and y both contain distinguished positions, and*
(iv) *vwx contains at most k distinguished positions.*

Another useful result is the closure property of the class of CFLs [14].

Proposition 2.6. *The class of CFLs is closed under union, concatenation, closure, homomorphism, inverse homomorphism and intersection with a regular language.*

3. When the kneading map is unbounded

In this section we consider languages of unimodal maps whose kneading maps are unbounded. This class includes the languages of Feigenbaum attractor [4, 9, 21], Fibonacci map [15, 16], and all odd Fibonacci languages [26, 27]. (See examples 4 and 5, and definition 2.2 for their kneading sequences.)

The main result of this section is as follows.

Theorem 1. *Let KS be a given kneading sequence and Q be its kneading map. If Q is unbounded, then the language $L = \mathcal{L}(KS)$ is not context-free.*

In order to prove theorem 1 we need to establish some lemmas as follows.

Lemma 3.1. *Let KS and $L = \mathcal{L}(KS)$ be given as in theorem 1. If a string $u \in L$ is of the form*

$$u = B_0B_1 \ldots B_{r-1}u' \tag{13}$$

for some $r > 0$, and the length of u' satisfies the condition $|u'| \geqslant |B_r| - 1 = |B_r\pi|$, then u' must have $B_r\pi$ as its prefix.

Proof. Firstly, from definition 2.1 of L and (11) we have

$$u = B_0B_1 \ldots B_{r-1}u' \leqslant KS = B_0B_1 \ldots B_{r-1}B_r \ldots .$$

Since the string $B_0B_1 \ldots B_{r-1}$ ($r > 0$) is odd (see the fourth property of kneading map in section 2.9), we obtain $u' \geqslant B_r$.

On the other hand, u', as a substring of u, is also a string of L, thus $u' \leqslant KS$ is true. Since $B_r\pi$ ($r > 0$) is a prefix of KS (see (12) and (11)), we also have $u' \leqslant B_r\pi$. Combining these results finishes our proof. □

Lemma 3.2. *Let KS and $L = \mathcal{L}(KS)$ be given as in theorem 1. If a string $v \in L$ is nonempty, and $M(v)$ is its cyclic shift maximal string, then $v^i \in L$ for all $i > 0$ if and only if $M(v^2) = [M(v)]^2 < KS$.*

Proof. Since the string v^2 is always even, it suffices to prove that if $v \in L$ is an even string, i.e. v contains even number of $1'$s, then

$$v^i \in L \qquad \text{for all} \quad i > 0 \Longleftrightarrow M(v) < KS.$$

It is also clear that $v^i \in L$ for all $i > 0$ is equivalent to $[M(v)]^i \in L$ for all $i > 0$. Without loss of generality, we can assume that $v = M(v)$ holds in our discussion hereafter.

The 'if' part. From the definition of $M(v)$ and the condition $v = M(v) < KS$, it is clear that $\sigma^k(v^i) \leqslant KS$ holds for every $i > 0$ and $0 \leqslant k < |v^i|$. This implies that $v^i \in L = \mathcal{L}(KS)$ for all $i > 0$.

The 'only if' part. Assume the contrary that $v = M(v) \geqslant KS$ is true. Since the condition $v \in L = \mathcal{L}(KS)$ means that $v \leqslant KS$, we find that v must be a prefix of KS.

Writing $KS = vK_1$, then from $K_1 \leqslant KS$ and v being even, we also have

$$KS = vK_1 \leqslant vKS \qquad \text{and} \qquad KS \leqslant v^\infty.$$

Moreover, since $v^i \in L$ for all $i > 0$, we have $v^i \leqslant KS$ for all $i > 0$, and hence $v^\infty \leqslant KS$ holds. Therefore we obtain $KS = v^\infty$, a contradiction with the nonperiodicity of KS, which is implied by the unboundedness of the kneading map Q (see the fifth property of the kneading map in section 2.9). □

Remark. *In section 2.10 we present two lemmas, the pumping lemma 2.4 and the Ogden lemma 2.5, both of them necessary conditions for a language to be context-free. It can be shown that the first is not strong enough to prove that a language $L = \mathcal{L}(KS)$ is nonCFL, if it is known already that L is nonregular. Hence we have to rely on lemma 2.5 in the following proof.*

It should be noted that in the rest of the paper we often use the factorial property of the language L of a unimodal map (see the last paragraph of section 2.7):

$$\text{If } x \in L, \text{ and } y \text{ is a substring of } x, \qquad \text{then } y \in L. \tag{14}$$

Proof of theorem 1. Assume the contrary that $L = \mathcal{L}(KS)$ is context-free. By applying the Ogden lemma 2.5 to L we obtain the integer $k(L)$ claimed in the lemma.

Taking an integer r such that

$$|B_0 \dots B_r| \geqslant k(L), \tag{15}$$

and then using the unboundedness of both the kneading map Q and $\{|B_n|\}_{n\geqslant 0}$ (the third property of the kneading map in section 2.9), we can find integers m, q, l, p consecutively that satisfy requirements as follows:

$$|B_m| > 2|B_0 \dots B_r|, \tag{16}$$

$$r < m < Q(q), \tag{17}$$

$$|B_l| > 2|B_0 \dots B_q|, \tag{18}$$

$$q < l < Q(p). \tag{19}$$

By these selections we can write $B_0 \ldots B_p$, a prefix of KS, into the form of

$$B_0 \ldots B_p = B_0 \ldots B_{p-1} \underbrace{B_0 \ldots B_{q-1} \overbrace{B_0 \ldots B_r \ldots B_m \ldots \bar{B}_{Q(q)}}^{B_q} \ldots B_l \ldots \bar{B}_{Q(p)}}_{B_p}, \tag{20}$$

in which the substrings B_m and B_l will play distinctive role later.

Denote the prefix $B_0 \ldots B_{p-1} B_0 \ldots B_{q-1} B_0 \ldots B_r$ of (20) by z and designate its last $|B_0 \ldots B_r|$ positions as distinguished:

$$z = B_0 \cdots B_{p-1} B_0 \cdots B_{q-1} \overbrace{B_0 \cdots B_r}^{\text{distinguished}}. \tag{21}$$

From the statement of lemma 2.5 about the integer $k(L)$, and using the fact that $z \in L$ and the inequality (15), there exists a decomposition of z in the form of

$$z = uvwxy, \tag{22}$$

which satisfies all conditions (i)–(iv) in lemma 2.5.

There exist two cases to be treated separately.

Case (1). This case contains two subcases that (1.1) $|u| \leqslant |B_0 \ldots B_{p-1}|$, and (1.2) $|u| > |B_0 \ldots B_{p-1}|$ but $v = \varepsilon$ in (22).

From the condition (ii) of lemma 2.5, the substring w in (22) must contain some distinguished positions. Comparing (21) and (22), there exists a string t satisfying requirements:

$$uvw = B_0 \ldots B_{p-1} B_0 \ldots B_{q-1} t \tag{23}$$

and

$$txy = B_0 \ldots B_r. \tag{24}$$

Since in subcase (1.1) u cannot contain distinguished positions, and in subcase (1.2) $v = \varepsilon$, using the condition (iii) of lemma 2.5 we see that in case (1) the substring x and y in (22) must contain distinguished positions, and hence $x \neq \varepsilon$ holds.

From (i) of lemma 2.5, (23) and (14), we have in subcase (1.1) that

$$uv^i wx^i y \in L \Longrightarrow vwx^i \in L \Longrightarrow B_0 \ldots B_{q-1} t x^i \in L \qquad \text{for all} \quad i > 0$$

and in subcase (1.2)

$$uwx^i y \in L \Longrightarrow uwx^i \in L \Longrightarrow B_0 \ldots B_{q-1} t x^i \in L \qquad \text{for all} \quad i > 0.$$

Because of $|x| > 0$ we can find an integer i_0 such that $|tx^{i_0}| \geqslant |B_q|$. Using lemma 3.1 to the string $B_0 \ldots B_{q-1} t x^{i_0} \in L$, we find that (see (20))

$$tx^{i_0} = (B_q \pi) \ldots = B_0 \ldots B_r \ldots B_m \ldots (B_{Q(q)} \pi) \ldots. \tag{25}$$

Since (24) and $|x| > 0$ imply that $|t| < |B_0 \ldots B_r|$, the substring B_m in the right-hand side of (25) (see (20)) must be embedded into the string x^{i_0}.

On the other hand, the inequality (16) guarantees now that $i_0 \geqslant 3$. Therefore, a square of some cyclic shift of x, say $[\sigma^j(x)]^2$, is a proper prefix of B_m, and also a prefix of KS. Since $M(x^2) = [M(x)]^2 \geqslant [\sigma^j(x)]^2$, we have $M(x^2) \geqslant KS$. This contradicts the fact that $x^i \in L$ for all $i > 0$ and the statement of lemma 3.2, i.e. $M(x^2) < KS$.

Case (2). Now we have $|u| > |B_0 \ldots B_{p-1}|$ and $v \neq \varepsilon$ in (22). Here the discussion is simpler than that of case (1) since we only need to use

$$uv^i \in L \qquad \text{for all} \quad i > 0,$$

which is a consequence of the condition (i) of lemma 2.5 and (14).

Since $|v| > 0$ we can take an integer i_1 such that

$$|uv^{i_1}| > |B_0 \ldots B_{p-1} \underbrace{B_0 \ldots B_q \ldots B_l \ldots (B_{Q(p)}\pi)}_{B_p\pi}|$$

holds. Since $|u| > |B_0 \ldots B_{p-1}|$, uv^{i_1} has the form of $B_0 \ldots B_{p-1} \ldots$. Using lemma 3.1 to $uv^{i_1} \in L$, we find that

$$uv^{i_1} = B_0 \ldots B_{p-1} B_0 \ldots B_q \ldots B_l \ldots . \tag{26}$$

Considering that

$$|u| < |z| < |B_0 \ldots B_{p-1} B_0 \ldots B_q|,$$

the substring B_l in the right-hand side of (26) must be embedded into v^{i_1}. The remaining argument is nearly the same as in case (1) only by using (18) and $v^i \in L$ instead of (16) and $x^i \in L$. □

From the discussion in examples 4 and 5 we have the following.

Corollary 3.3. *The languages of the Feigenbaum attractor and the Fibonacci map are not context-free.*

It is easy to prove that both odd and even Fibonacci languages (defined in section 2.8) are context-sensitive. Moreover, we can establish a more general conclusion as follows (see [14] for concepts involved).

Proposition 3.4. *If a kneading sequence KS is generated from some substitution rule h as $KS = h^\infty(\alpha) = \lim_{n\to\infty} h^n(\alpha)$ starting from some string α, then the language $L = \mathcal{L}(KS)$ is context-sensitive.*

Proof. It suffices to describe how to construct a Turing machine to recognize L [14]. Using the finite control of it, this machine should easily generate $h^n(\alpha)$, the prefixes of KS, for any n. For a given string s, this machine can compute the prefix of KS which is of the same length with s. After that it can check the conditions

$$\sigma^i(s) \leqslant KS \qquad \text{for all} \quad i = 0, 1, \ldots, |s| - 1,$$

to decide to accept s or not. It is clear that this Turing machine can be realized by a linear bounded automaton (LBA), and hence $\mathcal{L}(KS)$ is a context-sensitive language as desired. □

Remark. *Using the notion of a D0L system, one of Lindenmayer's systems, the sequence of strings $\{h^n(\alpha)\}_{n\geqslant 0}$ is called a D0L language. The limit of this sequence generates the kneading sequence in proposition 3.4. This result can be restated as: if the kneading sequence KS is generated from a D0L system, then the language $\mathcal{L}(KS)$ is not beyond the level of context-sensitive.*

In [26] it was proved that the odd Fibonacci languages are nonregular. Now we can say more about their grammatical position in Chomsky hierarchy (see (1) in the introduction).

Theorem 2. *Each odd Fibonacci language is context-sensitive, but not context-free.*

Proof. Here we need only to prove that the kneading map of odd Fibonacci language is unbounded, and then to use theorem 1 and proposition 3.4.

Let $a_n = h^n(\alpha)$ and $b_n = h^n(\beta)$ be denoted for all $n \geqslant 0$ (see definition 2.2). It is easy to prove inductively that for each n, a_n is an odd primitive maximal prefix of KS and $a_{n+1}a_n$ is also a prefix of KS [26]. Therefore, using the fourth property of the kneading maps in section 2.9, a_{n+1} can be written as

$$a_{n+1} = B_0 B_1 \ldots B_m$$

for some integer m depending on n. Observing that

$$KS = a_{n+1}a_n \ldots = B_0 \ldots B_m a_n \ldots = B_0 \ldots B_m B_{m+1} \ldots,$$

and using the facts that both a_n and $\bar{B}_{m+1}$ are prefixes of KS, we find that $|a_n| < |B_{m+1}|$. Since $\{|a_n|\}$ is unbounded, it is clear that both $\{|B_n|\}$ and the kneading sequence Q are unbounded (the third property of the kneading map in section 2.9). □

4. Grammatical level of even Fibonacci languages

After obtaining theorem 1, the next step should be to discuss the languages of unimodal maps associated with bounded kneading maps, provided that their kneading sequences are aperiodic (as mentioned in example 2 that all eventually kneading sequences are associated with bounded kneading maps and regular languages).

However we still do not know of any powerful tool to attack this problem in the general case. It is clear that the proof of theorem 1 completely relies on the condition of unbounded kneading maps (see (16)–(20)), hence we cannot use the method of section 3 again.

On the other hand, in contrast with the odd Fibonacci languages which are associated with unbounded kneading maps (see the proof of theorem 2), we have the following.

Proposition 4.1. *The kneading map of each even Fibonacci kneading sequence is bounded.*

The proof of this result is simple and omitted. An example has been already shown in section 2.9 (example 3).

In this section we will prove that each even Fibonacci language is not context-free. Hence at least for a subclass of unimodal languages whose kneading map is bounded the conjecture proposed in [4, 25, 27] is still true. The method used here is quite different from those in section 3. Instead of Ogden lemma 2.5 the pumping lemma 2.4 is used, and a detailed study for a special sequence K (defined below) is the key factor to success.

The main result of this section is as follows.

Theorem 3. *Every even Fibonacci language is context-sensitive, but not context-free.*

The first statement is a consequence of proposition 3.4. In order to prove the second statement, we will show that it suffices to discuss a special language which is not generated from unimodal maps.

Let h be the substitution rule given by

$$h = (1 \to 110, 0 \to 10), \qquad (27)$$

and K be the infinite sequence defined by

$$K = h^{\infty}(1) = \lim_{n\to\infty} h^n(1) = 11011010, 11011010, 11010, \ldots. \qquad (28)$$

Readers who are familiar with circle maps will have no difficulty in recognizing that K is exactly the kneading sequence of circle homeomorphism with golden mean ratio $(\sqrt{5}-1)/2$ as its rotation number (see [8, 18, 27]).

By imitating definition 2.1 for the languages of unimodal maps, a concrete language generated by K can be defined formally as follows:

$$\mathcal{L}(K) = \{x \in S^* | \sigma^i(x) \leqslant K \text{ for } i = 0, 1, \ldots, |x| - 1\}, \tag{29}$$

in which the order relation used is the lexicographical one, defined simply by $0 < 1$ and $u0\ldots < u1\ldots$ for any prefix $u \in S^*$ with $S = \{0, 1\}$. It can be shown that K is a shift-maximal sequence in the sense of a lexicographical order relation. Hence this definition of $\mathcal{L}(K)$ is quite different with definition 2.1. Some properties, however, are preserved: the factorial property (14), every substring of K belongs to $\mathcal{L}(K)$, every string 0^n ($n > 0$) belongs to $\mathcal{L}(K)$, etc.

It is easy to show that the sequence K cannot be a kneading sequence for any unimodal map, and the language $\mathcal{L}(K)$ is neither a language for any unimodal map nor a language for any circle homeomorphism. As a matter of fact, it is not clear whether the language $\mathcal{L}(K)$ has any proper dynamical meaning. The role of this language is entirely technical and reflected in the following result.

Proposition 4.2. *Let K be the sequence defined by (28). If the language $\mathcal{L}(K)$ defined by (29) is not context-free, then every even Fibonacci language is also not context-free.*

Proof. Let $KS = g^\infty(\alpha)$ be an even Fibonacci kneading sequence generated from the substitution rule $g = (\alpha \to \alpha\alpha\beta, \beta \to \alpha\beta)$ where the strings α, β satisfy the conditions required in definition 2.3.

Take a homomorphism h_1 as follows:

$$h_1 = (1 \to \alpha\alpha\beta, 0 \to \alpha\beta).$$

It is clear that this h_1 can be extended to become a mapping from $(0+1)^*$ to $(\alpha\alpha\beta + \alpha\beta)^*$, and $KS = h_1(K)$ holds. (It seems that taking $(1 \to \alpha, 0 \to \beta)$ is more natural, but then the proof cannot be completed as follows.)

The key step of this proof is to establish the following relation between the languages $\mathcal{L}(K)$ and $\mathcal{L}(KS)$, i.e., the even Fibonacci language:

$$\mathcal{L}(K) = h_1^{-1}[\mathcal{L}(KS) \cap (\alpha\alpha\beta + \alpha\beta)^*], \tag{30}$$

where the notation h_1^{-1} is the inverse homomorphism of h_1 [14]. Using this relation and proposition 2.6 our proof is finished.

The reason for taking the intersection in the right-hand side of (30) is that although KS is a concatenation of $\alpha\alpha\beta$ and $\alpha\beta$, the strings in $\mathcal{L}(KS)$ are not. For example, every string 0^n ($n > 0$) belongs to $\mathcal{L}(KS)$.

If $x \in h_1^{-1}[\mathcal{L}(KS) \cap (\alpha\alpha\beta + \alpha\beta)^*]$, then we have

$$h_1(x) = z \in \mathcal{L}(KS) \cap (\alpha\alpha\beta + \alpha\beta)^*.$$

Denote $x = x_1 \ldots x_n$, where each $x_i \in S = \{0, 1\}$, and $z_i = h_1(x_i)$ for $i = 1, \ldots, n$, hence $z = z_1 \ldots z_n$. Since $z \in \mathcal{L}(KS)$, for each $i = 1, \ldots, n$ the relation

$$z_i \ldots z_n \leqslant KS$$

is true. This implies directly that the relation

$$x_i \ldots x_n \leqslant K$$

is also true for every $i = 1, \ldots, n$. Therefore, we know that $x \in \mathcal{L}(K)$. This means that

$$\mathcal{L}(K) \supseteq h_1^{-1}[\mathcal{L}(KS) \cap (\alpha\alpha\beta + \alpha\beta)^*].$$

Now suppose that $x = x_1 \ldots x_n \in \mathcal{L}(K)$ where each $x_i \in S = \{0, 1\}$ and construct

$$z = z_1 \ldots z_n, \qquad \text{where } z_i = h_1(x_i) \qquad \text{for } i = 1, \ldots, n.$$

We should prove that this string z belongs to $\mathcal{L}(KS) \cap (\alpha\alpha\beta + \alpha\beta)^*$. It is obvious that $z \in (\alpha\alpha\beta + \alpha\beta)^*$ and

$$z_i \ldots z_n \leqslant KS \qquad \text{for} \quad i = 1, \ldots, n$$

(from $x_i \ldots x_n \leqslant K$ for $i = 1, \ldots, n$). In order to prove further that $\sigma^k(z) \leqslant KS$ holds for all $0 \leqslant k < |z| - 1$, we need only treat cases that for some i, $1 \leqslant i < n$, $z_i = uv$, where $0 < |u| < |z_i|$, and

$$\sigma^k(z) = vz_{i+1} \ldots z_n. \tag{31}$$

(There is no difficulty for the case of $|\sigma^k(z)| < |z_n|$.) There are four cases to be treated separately, in which the main tool is the notion of primitive maximal strings presented in items (7) and (9) of section 2.1.

Case 1 is $z_i z_{i+1} = \alpha\alpha\beta\alpha\alpha\beta$. Since then string (31) has $\sigma^{|u|}(\alpha\alpha\beta)$ as its prefix, and $\alpha\alpha\beta$ is primitive maximal, the inequality $\sigma^k(z) < KS$ is true.

The next two cases are either $z_i z_{i+1} = \alpha\beta\alpha\beta$ or $z_i z_{i+1} = \alpha\beta\alpha\alpha\beta$, and the discussion is nearly the same as for case 1.

The last case is $z_i z_{i+1} = \alpha\alpha\beta\alpha\beta$. If $0 < |u| < |\alpha|$, then we can use the fact that α is primitive maximal. If $|u| = |\alpha|$, then $vz_{i+1} = [\alpha\beta]^2$ and we can use the fact that $\beta < \alpha$ to obtain $\sigma^k(z) < KS$. If $|u| > |\alpha|$, then vz_{i+1} has $\sigma^{|u|-|\alpha|}(\alpha\beta)$ as its prefix. From $vz_{i+1} < \alpha\beta < KS$ we obtain the required result.

Hence we have

$$\mathcal{L}(K) \subseteq h_1^{-1}[\mathcal{L}(KS) \cap (\alpha\alpha\beta + \alpha\beta)^*]$$

and complete our proof. □

Now the problem remaining is to prove the following proposition.

Proposition 4.3. *The language $\mathcal{L}(K)$ is not context-free.*

If this is true then combining it with proposition 4.2 the proof of theorem 3 is completed.

Several lemmas are indispensable for the proof of proposition 4.3. The first one is quite similar to lemma 3.2 and hence its proof is omitted.

Lemma 4.4. *If x is a nonempty substring of the sequence K, then $x^i \in \mathcal{L}(K)$ for all $i > 0$ if and only if $M(x) < K$.*

Lemma 4.5. *Let the string uvw be a prefix of K and $v \neq \varepsilon$. If $uv^i w \in \mathcal{L}(K)$ for all $i \geqslant 0$, then w is a prefix of v^∞.*

Proof. For $i = 0$ we have $uw \leqslant K = uvw \ldots$, and hence $w \leqslant vw$. This simply leads to that $w \leqslant v^\infty$ is true. On the other hand, from $uv^i w \leqslant uvw \ldots$ for i big enough we have $v^{i-1} \leqslant w$, and hence $v^\infty \leqslant w$. Combining these results completes our proof. □

Lemma 4.6. *Let the string $uvwxy$ be a prefix of K and $x \neq \varepsilon$. If $uv^i wx^i y \in \mathcal{L}(K)$ for all $i \geqslant 0$, then $y \leqslant x^\infty$ is true.*

Proof. If $v = \varepsilon$ then we can use lemma 4.5 to know that y is a prefix of x^∞. Otherwise, from using lemma 4.5 to uvw we can write

$$w = v^j v_1 \qquad \text{for some } j \geqslant 0 \quad \text{and} \quad v = v_1 v_2 \quad \text{with} |v_2| > 0.$$

Thus using the condition for $i = 0$, from $uwy \leqslant K = uvwxy \ldots$ we obtain

$$y \leqslant v_2 v_1 xy. \tag{32}$$

On the other hand, using the condition $uv^i w \leqslant uvwxy \ldots$ for i big enough we obtain

$$xy \geqslant (v_2 v_1)^\infty. \tag{33}$$

We claim that the inequality $y \leqslant xy$ is true and, consequently, $y \leqslant x^\infty$ holds.

Assume the contrary that $y > xy$ holds. Combining it with (32) leads to $y \leqslant v_2 v_1 xy < v_2 v_1 y$ and hence $y \leqslant (v_2 v_1)^\infty$. Combining this with (33) we obtain $y \leqslant (v_2 v_1)^\infty \leqslant xy$, and hence $y \leqslant xy$ that contradicts our hypothesis. □

Remark. *The next two propositions are unlike previous lemmas in that their proofs completely depend on a study about substrings of K. Since this discussion is long and technical we put it into appendices A and B.*

Proposition 4.7. *If $K = pxs$ where p is a prefix, s is an infinite suffix of K, and $M(x) < K$ holds, then $s \geqslant x^\infty$.*

Proposition 4.8. *The string $z = h^n(1)h^{n-1}(1)\ldots h(1)11$ ($n > 0$) is a prefix of K, and for every suffix s of z (including $s = z$) the inequality $M(s) \geqslant K$ is true.*

Proof of proposition 4.3. Assume the contrary that the language $\mathcal{L}(K)$ is context-free. By applying lemma 2.4 (pumping lemma for CFL) to $\mathcal{L}(K)$ we obtain the integer $k(L)$ claimed in it.

Taking an integer n big enough such that the string

$$z = h^n(1)h^{n-1}(1)\ldots h(1)11$$

in proposition 4.8 satisfies the requirement $|z| > k(L)$. Since z is a prefix of K, hence $z \in \mathcal{L}(K)$ holds. From the statement of lemma 2.4 we can decompose z into $z = uvwxy$ such that $uv^i wx^i y \in \mathcal{L}(K)$ for all $i \geqslant 0$ and $|vx| > 0$.

If $x = \varepsilon$, then $v \neq \varepsilon$. Writing $z = uv(wy)$ and using lemma 4.4 to $v^i \in \mathcal{L}(K)$, we obtain $M(v) < K$. Using lemma 4.5 we know that the suffix wy of z is a prefix of v^∞. Thus z will have a cyclic shift of v as its suffix, say s, and $M(s) = M(v) < K$ holds.

If $x \neq \varepsilon$, then from lemmas 4.4 and 4.6 we have $M(x) < K$ and $y \leqslant x^\infty$. On the other hand, writing $K = (uvw)xy\ldots$ and using proposition 4.7 (as $M(x) < K$ holds) we also have $y \geqslant x^\infty$. Therefore, the suffix y of z must be a prefix of x^∞. Thus z also has a cyclic shift of x as its suffix, say s, and again by lemma 4.4 $M(s) = M(x) < K$ holds.

Thus for both cases we find a suffix s of z such that $M(s) < K$. This contradicts the conclusion of proposition 4.8 and our proof is complete. □

Acknowledgments

The authors would like to thank the referees for their critical review and valuable suggestions to improve the clarity of this paper. This study was supported by National Basic Research Project ‘Nonlinear Science’.

Appendix A. A discussion about substrings in K

Recalling the definition of K in section 4 (see (27) and (28)):

$$K = h^{\infty}(1) = 11011010, 11011010, 11010, \ldots,$$

where $h = (1 \to 110, 0 \to 10)$, some simple features of K can be established directly or inductively, e.g. there are no strings 111 and 00 in K. But much more properties have to be found out before we can prove propositions 4.7 and 4.8 and theorem 3.

At first some simple facts are presented without proof.

Lemma A.1. The following facts hold for every integer $n \geqslant 0$:

(1) $h^{n+1}(1) = h^n(1)h^n(1)h^n(0)$=$h^n(1)h^{n+1}(0)$,
(2) $h^{n+1}(0) = h^n(1)h^n(0)$,
(3) $h^n(1) > h^n(0)$,
(4) both $h^n(1)$ and $h^n(0)$ are primitive maximal strings,
(5) $h^{n+1}(1) = \overline{h^{n+1}(0)}\underline{h^n(1)}$, $h^{n+1}(0) = \overline{h^n(0)}\underline{h^n(1)}$ (see the notations $\bar{x}$ and $\underline{x}$ in section 2.1).

Lemma A.2. *There is no $[h^n(1)]^3$ appearing in K for each integer $n \geqslant 0$.*

Proof. Using $[h^n(1)]^3 > K = h^n(1)h^n(1)h^n(0)\ldots$ and the shift-maximal property of K is enough to obtain the statement. □

Lemma A.3. *Every appearance of the string $[h^n(0)]^2$ ($n > 0$) in K must be a suffix of some $h^{n+1}(0) = h^n(1)h^n(0)$ in K, and there is no $[h^n(0)]^3$ in K.*

Proof. From the definition of $K = h^{\infty}(1)$ we know that K is a fixed point of h: $K = h(K) = h^n(K)$ for each integer $n > 0$. Thus for each $n > 0$ we can express K as an infinite product, i.e. concatenation, as follows:

$$K = \prod_{1}^{\infty} h_i, \qquad \text{where each } h_i \text{ is either } h^n(1) \text{ or } h^n(0). \tag{34}$$

Since there is no 00 appearing in K, it is impossible to have $h_i = h_{i+1} = h^n(0)$ in (34). But since $h^n(1) = h^{n-1}(1)h^n(0)$ there does exist $[h^n(0)]^2$ appearing in K for each $n > 0$.

Here we claim a strong fact which implies the statements in the lemma directly. This fact is that for each substring s of length $|h^n(0)|$ in K there are only three cases: (1) $s = h^n(0)$, and s is either a factor $h_i = h^n(0)$ in (34) or a suffix of factor $h_j = h^n(1) = h^{n-1}(1)h^n(0)$ in (34), (2) $s = \overline{h^n(0)}$ if s is a prefix of $h^n(1)$, (3) s is a cyclic shift of $h^n(0)$ that $s = \sigma^i(h^n(0))$ for some $0 < i < |h^n(0)|$.

At first we consider s of length $|h^n(0)|$ appearing in $h^n(1)$. From lemma A.1 we have decompositions

$$\begin{aligned} h^n(1) &= h^{n-1}(1)h^n(0) \\ &= \overline{h^n(0)}\underline{h^{n-1}}(0) \\ &= h^{n-1}(1)\overline{h^{n-1}(0)}\underline{h^{n-1}(1)}, \end{aligned}$$

such that each substring of the length $|h^n(0)|$ is visible. There are three cases: (1) $s = h^n(0)$ if s is the suffix of $h^n(1)$, (2) $s = \overline{h^n(0)}$ if s is the prefix of $h^n(1)$, (3) $s = \sigma^i(h^n(0))$ for $0 < i < |h^n(0)|$. This can be seen from the last decomposition of $h^n(1)$ listed above, $h^n(0) = \overline{h^{n-1}(0)}\underline{h^{n-1}(1)}$ and the difference between the prefix $h^{n-1}(1)$ and the suffix $\underline{h^{n-1}(1)}$ being at their first symbols. Since $h^n(0)$ is primitive, hence $h^n(0)$ can appear in $h^n(1)$ only in case (1).

Next considering two successive factors in (34) which can be one of the three cases:

$$
\begin{aligned}
&h^n(1)h^n(1) = h^{n-1}(1)h^n(0)\overline{h^n(0)}\underline{h^{n-1}(1)},\\
&h^n(1)h^n(0) = h^{n-1}(1)h^n(0)h^n(0),\\
&h^n(0)h^n(1) = h^n(0)\overline{h^n(0)}\underline{h^{n-1}(1)}.
\end{aligned}
$$

Again by the primitivity of $h^n(0)$, our claim is true and the lemma is proved. □

Remark. *The fact claimed in lemma A.3 also implies that the appearance of $h^n(1)$ can only happen when it is a factor in (34) and the appearance of $h^n(0)$ can only happen when it is either a factor or a suffix of $h^n(1)$ in (11).*

The next two facts are consequences from lemmas A.2 and A.3.

Lemma A.4. *If $h^m(0)h^n(0)$ appears in K, then $m \leqslant n$ holds.*

Proof. If $m > n$ then $h^m(0)h^n(0)$ will have a suffix $h^{n+1}(0)h^n(0) = h^n(1)h^n(0)h^n(0)$ which contradicts lemma A.3. □

Lemma A.5. *If $h^m(1)h^n(0)$ appears in K, then $n - 1 \leqslant m \leqslant n$ holds.*

Proof. If $m > n$ then $h^m(1)h^n(0) = h^{m-1}(1)h^m(0)h^n(0)$ that contradicts lemma A.4. If $m < n - 1$, then $h^m(1)h^n(0) = h^m(1)h^{n-1}(1)\ldots = h^m(1)h^{m+1}(1)\ldots = [h^m(1)]^3\ldots$ which contradicts lemma A.2. □

Lemma A.6. *Let p be a nonempty prefix of K. (1) If $K = ph^n(1)\ldots = ups$ for some $n \geqslant 0$ and some prefix u, then the infinite suffix s of K must begin from either $h^n(1)$ or $h^n(0)$. (2) If $K = ph^n(0)\ldots = ups$ for some $n > 0$ and some prefix u, then s must begin from either $h^n(0)$ or $h^{n-1}(0)$, but not $h^n(1)$.*

Proof. (1) Using remark after lemma A.3 the prefix p in $K = ph^n(1)\ldots$ must be a product of h_i in (34). Since $p \neq \varepsilon$, p contains at least a factor $h^n(1)$. The same argument and comparing $K = ups$ with (34) leads to the statement required.

(2) From the fact proved in lemma A.3 we know that the $h^n(0)$ in $K = ph^n(0)\ldots$ appears only in two cases. If this $h^n(0)$ is a suffix of $h^n(1)$ in (34), then p has $h^{n-1}(1)$ as its suffix. Hence the s in $K = ups$ begins from either $h^{n-1}(0)$ or $h^{n-1}(1)$. By lemma A.2 in the latter case s must begins from $h^{n-1}(1)h^{n-1}(0) = h^n(0)$. If $h^n(0)$ in $K = ph^n(0)\ldots$ is itself a factor in (34), then p is a product of h_i in (34). By the same argument as in (1) the s begins from either $h^n(0)$ or $h^n(1)$. But for the latter case, then we would have

$$\sigma^{|u|}(K) = ph^n(1)\ldots > ph^n(0)\ldots = K,$$

a contradiction to the shift-maximal property of K. □

Lemma A.7. *There is no string of the form*

$$
\begin{aligned}
s &= h^n(0)\left(\prod_{i=n}^{n+m} h^i(0)\right)h^{n+m}(0)\\
&= [h^n(0)]^2 h^{n+1}(0)\ldots h^{n+m-1}(0)[h^{n+m}(0)]^2
\end{aligned}
$$

in K for any $n > 0$ and $m > 0$.

Proof. Assume the contrary that such string s appears in K. From lemma A.3 s cannot be a prefix of K, and we have $K = ps\ldots$ for some $p \neq \varepsilon$. Using lemma A.3 repeatedly the prefix p must have $h^{n+m-2}(1)\cdots h^{n-1}(1)$ as its suffix, and then we obtain

$$h^{n+m-2}(1)\ldots h^{n-1}(1)s = [h^{n+m}(0)]^3$$

appearing in K, a contradiction to lemma A.2. □

Lemma A.8. *Every nonempty prefix p of K can be written in the form of*

$$h^{m_1}(1)\ldots h^{m_k}(1),$$

where the sequence of integers $(m_1,\ldots,m_k)$ is nonincreasing.

Proof. If $p = h^n(1)$ then simply let $m_1 = n$ and $k = 1$. Otherwise, we can find an integer $n \geqslant 0$ such that $|h^n(1)| < |p| < |h^{n+1}(1)|$ holds, and then take $m_1 = n$. From $h^{n+1}(1) = h^n(1)h^n(1)h^n(0)$ and $h^n(0) = h^{n-1}(1)h^{n-1}(0)$ we can continue this procedure to find out the sequence $(m_1,\ldots,m_k)$ as required. □

The next fact is important.

Lemma A.9. *If a string v appears in K, and $\bar{v}$ is a prefix of K, then there exists an integer $n \geqslant 0$ such that $v = h^n(0)$.*

Proof. By lemma A.8 we can write $\bar{v} = h^{m_1}(1)\ldots h^{m_k}(1)$ for some $m_1 \geqslant \cdots \geqslant m_k$.

We claim that $m_k = 0$ and, moreover, the sequence $(m_1,\ldots,m_{k-1})$ must be an arithmetic progression with common difference -1 and the last term $m_{k-1} = 0$. If this is true, then $m_i = k-1-i$ for $1 \leqslant i \leqslant k-1$, and it follows that

$$v = h^{k-2}(1)\ldots h(1)10 = h^{k-1}(0)$$

and the statement of lemma holds with $n = k-1$.

Let us begin from m_k. If $m_k > 0$, then the last symbol of $\bar{v}$ is 0, and $v > \bar{v}$ holds. This would lead to $v > K$, which contradicts the condition that v is a substring of K and the kneading sequence K is shift-maximal. Hence $m_k = 0$ is true.

If $m_{k-1} > 0$, then $\bar{v}$ has 101 as a suffix, and v has 100 as a suffix that contradicts the fact that there is no 00 in K. Hence $m_{k-1} = 0$ is also proved.

Assume that we already have established that for some l, $1 < l \leqslant k-1$, the formula $m_i = k-1-i$ holds for $l \leqslant i \leqslant k-1$ and consider m_{l-1} inductively. Since in v the substring $h^{m_{l-1}}(1)$ is followed by

$$h^{k-1-l}(1)\ldots h(1)10 = h^{k-l}(0),$$

using lemma A.5 we know that

$$k-l-1 \leqslant m_{l-1} \leqslant k-l.$$

If $m_{l-1} = k-l$, then the induction is complete. Otherwise, when $m_{l-1} = k-l-1$ happens, then $\bar{v}$ would have a suffix

$$h^{k-l-1}(1)\overline{h^{k-l}(0)} = h^{k-l-1}(1)h^{k-l-1}(1)\overline{h^{k-l-1}(0)} > K,$$

which contradicts the condition that $\bar{v}$ is a prefix of K. □

By lemma 4.4 $x^i \in \mathcal{L}(K)$ for all $i > 0$ is equivalent to $M(x) < K$. The following lemma is a structural study of such x.

Lemma A.10. *If x be a substring of K and $M(x) < K$ holds, then the string x can be uniquely decomposed as $x = Xx'$ where x' is its longest suffix that is also a prefix of K ($x' = \varepsilon$ is also allowed), and*

$$X = h^{n_1}(0) \ldots h^{n_l}(0) \tag{35}$$

for some sequence $(n_1, \ldots, n_l)$ $(l > 0)$, and at the same time, for some k, $0 \leqslant k < l$, K is of the form:

$$K = x'h^{n_1}(0) \ldots h^{n_k}(0)h^{n_{k+1}}(1) \ldots. \tag{36}$$

Proof. Since $x \leqslant M(x) < K$, comparing x with K and using lemma A.9 (repeatedly) it is easy to find out expression (35). It is also easy to show that no nonempty suffix of any $h^n(0)$ can be a prefix of K, and hence x' has the property as required.

Now we only need to establish (36). Writing $K = x's \ldots$, where $|s| = |h^{n_1}(0)|$, from the fact in proof of lemma A.3 there are three cases: (1) $s = \overline{h^{n_1}(0)}$, (2) $s = h^{n_1}(0)$, (3) $s = \sigma^i(h^{n_1}(0))$ for some $0 < i < |s|$. For case (1) $\sigma^{|x'|}(K)$ must begin from $h^{n_1}(1)$ and $k = 0$. If case (2) happens then $l > 1$ must hold. Otherwise we would have $x = h^{n_1}(0)x'$ and $M(x) \geqslant x'h^{n_1}(0)$ leads to a contradiction with $M(x) < K = x'h^{n_1}(0) \ldots$. Hence we can proceed on to discuss $h^{n_2}(0)$. For case (3) we have $M(x) > K$, a contradiction again.

If $l > 1$ and case (2) happens, then we can repeat our argument further to obtain (36). We need only to show that $k = l$ is impossible. Otherwise, we have $K = x'X \ldots$. This leads to $M(x) \geqslant \sigma^{|X|}(x) = x'X$ and contradicts $M(x) < K$. □

Appendix B. Proofs of propositions 4.7 and 4.8

B.1. A critical lemma

The following lemma is called critical because (1) it leads directly to the proof of proposition 4.7, the most difficult step in the proof of theorem 3, and (2) in its proof we actually need all lemmas established in appendix A.

Lemma B.1. *If $K = pxs$ where p is a prefix, s is an infinite suffix of K, and $M(x) < K$ holds, then there exists a decomposition of $x = C0Dx'$ where x' is x's longest suffix that is also a prefix of K, and $K = x'C1 \ldots$ such that s has C as its prefix.*

Proof. By using lemma A.10 to x, we have $x = Xx'$ where x' is x's longest suffix that is also a prefix of K as required. From (35) and (36) we obtain

$$C = h^{n_1}(0) \ldots h^{n_k}(0)h^{n_{k+1}}(0)\pi,$$

such that $X = C0D$ and $K = x'C1 \ldots$. The problem which remains to be proven is that s (in $K = pxs$) must have C as its prefix.

The sequence K can be written in two expressions as follows:

$$K = x'h^{n_1}(0) \ldots h^{n_k}(0)h^{n_{k+1}}(1) \ldots \tag{37}$$

$$= ph^{n_1}(0) \ldots h^{n_l}(0)x's, \tag{38}$$

where $n_1, \ldots, n_l$ and k $(0 \leqslant k < l)$ are determined in lemma A.10.

From lemma A.4 we know that $n_1 \leqslant \cdots \leqslant n_l$.

If $k = 0$, then $C = h^{n_1}(0)\pi$, and expressions of K are reduced to

$$K = x'h^{n_1}(1) \ldots = ph^{n_1}(0) \ldots h^{n_l}(0)x's.$$

If $x' \neq \varepsilon$, then by using lemma A.6 (1) we know that s will have either $h^{n_1}(0)$ or $h^{n_1}(1)$ as its prefix. Hence both of them have C as their prefix. If $x' = \varepsilon$, then since s follows $h^{n_l}(0)$, s

will begin from either $h^{n_l}(0)$ or $h^{n_l}(1)$ (by lemma A.3). Since $n_1 \leqslant n_l$, s will also have C as prefix.

For $k > 0$ it suffices to prove that s will have $h^{n_1}(0)$ as its prefix. As a fact, if $s = h^{n_1}(0)s'$ holds already, then we can rewrite (37) and (38) to obtain

$$\begin{aligned} K &= (x'h^{n_1}(0))h^{n_2}(0)\ldots h^{n_k}(0)h^{n_{k+1}}(1)\ldots \\ &= (ph^{n_1}(0))h^{n_2}(0)\ldots h^{n_l}(0)(x'h^{n_1}(0))s', \end{aligned}$$

replacing x' by $x'h^{n_1}(0)$, and then repeat our arguments again.

By using lemma A.6 (2) to expressions of K, s will have either $h^{n_1}(0)$ or $h^{n_1-1}(0)$ as its prefix. Hence we need only to prove that the latter one is impossible. This is the content of the proof hereafter.

Using lemma A.3 we know that if in (38) s begins from $h^{n_1-1}(0)$ then x' will have either $h^{n_1-1}(1)$ or $h^{n_1-2}(1)$ as its suffix. But for the latter case expression (37) of K would contain $[h^{n_1-2}(1)]^3$, a contradiction to lemma A.2. Hence we can assume that

$$s = h^{n_1-1}(0)s' \qquad \text{and} \qquad x' = x''h^{n_1-1}(1)$$

hold in the rest of our proof.

We can also claim that, without loss of generality, our discussion can be limited to cases specified by the following condition:

$$0 \leqslant n_{i+1} - n_i \leqslant 1 \qquad \text{for all} \quad 0 \leqslant i \leqslant k. \tag{39}$$

As a matter of fact, if there is i, $0 \leqslant i \leqslant k$, such that $n_{i+1} - n_i \geqslant 2$, then we can suppose i is the minimal value for this case. If $i < k$ we have

$$K = x'h^{n_1}(0)\ldots h^{n_i}(0)h^{n_{i+1}}(0)\ldots = x'h^{n_1}(0)\ldots h^{n_i}(0)h^{n_i+1}(1)\ldots,$$

and can replace i with k. For $i = k$ we have

$$K = x'h^{n_1}(0)\ldots h^{n_k}(0)h^{n_k+1}(1)\ldots.$$

Thus we can always suppose that condition (39) is true.

Here there are four cases to be treated separately.

Case (1). In this case all differences $n_{i+1} - n_i = 1$ in (39), i.e. the sequence of integers $(n_1, n_2, \ldots, n_{k+1})$ is an arithmetic progression with common difference 1, then we can simply write

$$n_1 = n, n_2 = n + 1, \ldots, n_{k+1} = n + k.$$

By calculating

$$\begin{aligned} K &= x'h^n(0)\ldots h^{n+k-1}(0)h^{n+k}(1)\ldots \\ &= x''h^{n-1}(1)h^n(0)\ldots h^{n+k-1}(0)h^{n+k}(1)\ldots \\ &= x''h^{n+k-1}(1)h^{n+k}(1)\ldots, \end{aligned}$$

and $h^{n+k-1}(1)h^{n+k}(1) = [h^{n+k-1}(1)]^3\ldots > K$, a contradiction happens.

Case (2). In this case we assume $n_1 = n_2$, but the sequence $(n_2, \ldots, n_{k+1})$ is an arithmetic progression with common difference 1. Thus we can write

$$n_1 = n_2 = n, n_3 = n + 1, \ldots, n_{k+1} = n + k - 1. \tag{40}$$

There are two subcases to be treated separately.

(2.1) $k = 1$. Now the expressions of K are

$$\begin{aligned} K &= x''h^{n-1}(1)h^n(0)h^n(1)\ldots \\ &= x''h^n(1)h^n(1)\ldots \qquad (41) \\ &= ph^n(0)h^n(0)\ldots x''h^{n-1}(1)h^{n-1}(0)s' \\ &= ph^n(0)h^n(0)\ldots x''h^n(0)s'. \qquad (42) \end{aligned}$$

By using the remark after lemma A.3, from (41) we can see that x'' must have $h^n(1)h^n(0)$ as its suffix, but this will generate in (42) the substring

$$h^n(1)h^n(0)h^n(0) = h^{n-1}(1)[h^n(0)]^3,$$

a contradiction with lemma A.3.

(2.2) $k > 1$. Repeatedly using lemma A.3, i.e. if $[h^n(0)]^2$ can appears in K only as suffix of $h^{n+1}(1)$, we can proceed on until

$$K = x'h^n(0)h^n(0)\ldots h^{n+k-1}(1)\ldots = x'''h^{n+k-1}(0)h^{n+k-1}(1)\ldots,$$

where $x' = x'''h^{n+k-3}(1)\ldots h^n(1)h^{n-1}(1)$. Then

$$K = pXx'h^{n-1}(0)s' = pXx'''h^{n+k-2}(0)s'.$$

Since x''' is a prefix of K, by lemma A.8 we can assume that $x''' = x^{(4)}h^m(1)$ for some m. Using lemma A.5, we have $n + k - 3 \leqslant m \leqslant n + k - 2$. But by the same lemma from $K = x^{(4)}h^m(1)h^{n+k-1}(0)h^{n+k-1}(1)\ldots$ we also have $n + k - 2 \leqslant m \leqslant n + k - 1$. Thus $m = n + k - 2$ is uniquely determined.

Now we have

$$K = x^{(4)}[h^{n+k-1}(1)]^2\ldots \tag{43}$$

$$= ph^{n_1}(0)\ldots h^{n_l}(0)x^{(4)}h^{n+k-1}(0)s'. \tag{44}$$

If $x^{(4)} \neq \varepsilon$, then $x^{(4)}$ should have $h^{m'}(1)$ as its suffix for some m'. But from (43) and lemma A.2 $m' > n + k - 1$ must be true. However from (44) and lemma A.5 we have $m' \leqslant n + k - 1$, a contradiction.

If $x^{(4)} = \varepsilon$, then from (44) and lemma A.4 we have $n_1 \leqslant \cdots \leqslant n_l \leqslant n+k-1 = n_{k+1}$ and hence $n_l = n_{k+1}$. If $k + 1 < l$, then $[h^{n_{k+1}}(0)]^3$ appears in (44), a contradiction to lemma A.3. If $k+1 = l$, then in (44) we have $[h^n(0)]^2\ldots[h^{n+k-1}(0)]^2$, which leads to a contradiction with lemma A.7.

Case (3). Suppose that the sequence $(n_1, \ldots, n_k)$ is an arithmetic progression with common difference 1 and $n_k = n_{k+1}$. Writing

$$n_1 = n, n_2 = n + 1, \ldots, n_k = n_{k+1} = n + k - 1,$$

and calculating

$$h^{n-1}(1)h^n(0)h^{n+1}(0)\ldots h^{n+k-1}(0)h^{n+k-1}(1) = [h^{n+k-1}(1)]^2,$$

we find that x'' must have $h^{n+k-1}(1)h^{n+k-1}(0)$ as its suffix (again by the remark after lemma A.3). On the other hand, in

$$K = pXx''h^{n-1}(1)h^{n-1}(0)s'$$

we would have $h^{n+k}(0)h^n(0)$ which contradicts lemma A.4.

Case (4). Suppose there is an integer i, $1 < i < k$, such that $n_i = n_{i+1}$, and both sequences $(n_1, \ldots, n_i)$ and $(n_{i+1}, \ldots, n_{k+1})$ are arithmetic progressions with common difference 1. Then we have

$$K = x''h^{n-1}(1)h^n(0)h^{n+1}(0)\ldots[h^{n+i-1}(0)]^2\ldots h^{n+k-2}(0)h^{n+k-1}(1)\ldots.$$

From the calculation

$$h^{n-1}(1)h^n(0)\ldots[h^{n+i-1}(0)]^2\ldots h^{n+k-2}(0)h^{n+k-1}(1)$$
$$= [h^{n+i}(0)]^2\ldots h^{n+k-2}(0)h^{n+k-1}(1)$$

we can find that (like in subcase (2.2))

$$x' = x'''h^{n+k-3}(1)\ldots h^{n+i-1}(1)h^{n-1}(1).$$

and

$$K = pXx'h^{n-1}(0)s' = pXx'''h^{n+k-3}(1)\ldots h^{n+i-1}(1)h^n(0)s',$$

a contradiction with lemma A.5 happens.

Since lemma A.7 means that the equality $n_{i+1} = n_i$ can happen only once, if at all, the discussion of cases (1)–(4) includes all possibilities that $n_{i+1} - n_i \leqslant 1$ for all $i = 1, \ldots, k$. □

B.2. Proof of proposition 4.7

Proof. By using lemma B.1, we have $x = Xx'$, $X = C0D$, $K = pxs = x'C1\ldots$, and s has C as its prefix. Write s in the form as follows:

$$s = Cas'' \qquad \text{where} \quad a \in \{0, 1\}.$$

If $a = 1$ then $s > x$ holds, and $s \geqslant x^\infty$ is true. ($a = 1$ can happen in reality.) Hence assume $a = 0$ in the following discussion.

From $s = C0s''$ we have $K = x'C1\ldots = pXx'C0s''$. By using lemma A.9 there exists an integer N such that $x'C0 = h^N(0)$. Therefore, we have

$$K = pxs = pXx's = pC0Dx'C0s'' = pC0Dh^N(0)s''.$$

Since s'' follows $h^N(0)$ in K, s'' must begin from either $h^N(1)$ or $h^N(0)h^N(1)$ (see lemma A.3 and its remark). Hence if $D = \varepsilon$, then $x'X = h^N(0)$, and we have the required result

$$x^\infty = X(x'X)^\infty = X[h^N(0)]^\infty = C0[h^N(0)]^\infty < s = C0s''.$$

If $D \neq \varepsilon$, then D begins from some $h^i(0)$ (by lemma A.10). By $K = pC0Dh^N(0)s''$ and lemma A.4, we have $i \leqslant N$. If $i < N$, then from $h^i(0) < h^N(0) < h^N(1)$ we obtain $s'' > D$, and $s > x = C0Dx'$. If $i = N$, then using lemma A.3, i.e. there is no $[h^N(0)]^3$ in K, we can only have $D = h^N(0)$ and $s'' = h^N(1)\ldots$. Hence $s'' > D$, and $s > x = C0Dx'$ as required. □

B.3. Proof of proposition 4.8

Proof. Since each $h^{n+1}(1)$ ($n \geqslant 0$) is a prefix of K, from the identity

$$h^{n+1}(1) = h^n(1)h^{n-1}(1)\ldots h(1)110h(0)\ldots h^n(0)$$

it is clear that the string $z = h^n(1)h^{n-1}(1)\ldots h(1)11$ ($n \geqslant 0$) is a prefix of K.

Assume the contrary that there is a suffix s of z such that $M(s) < K$. It is easy to see that $|s| > 2$ is necessary. From $M(s) < K$ we know that s itself cannot be a prefix of K, and hence from the structure of z, there exists some j, $2 \leqslant j \leqslant n$, such that $s = Xx'$, where $x' = h^{j-1}(1)\ldots h(1)11$ is a suffix of z and also a prefix of K, while X is a nonempty proper suffix of $h^j(1)$. Moreover, since it can be shown inductively that every proper suffix of any $h^j(1)$ cannot be a prefix of K, the string x' is s's longest suffix that is also a prefix of K.

By the hypothesis of $M(s) < K$, we can also use lemma A.10 to s. By the property of x' just obtained, the result of lemma A.10 gives

$$X = h^{n_1}(0)\ldots h^{n_l}(0)$$

for some $n_1, \ldots, n_l$, and for some k, $0 \leqslant k < l$,

$$K = x'h^{n_1}(0)\ldots h^{n_k}(0)h^{n_{k+1}}(1)\ldots.$$

On the other hand, we also have

$$K = \underbrace{\overbrace{h^{j-1}(1)\ldots h(1)11}^{x'}\,0h(0)\ldots h^{j-1}(0)}_{h^j(1)}\,h^j(1)h^j(0)\ldots.$$

Comparing these two expressions of K reveals that $k = j$,

$$n_1 = 0, \qquad n_2 = 1, \ldots, n_k = j-1, \qquad n_{k+1} = j,$$

and hence $X = 0h(0)\ldots h^j(0)\ldots$. Estimating the length of X, i.e.

$$|X| \geqslant |0h(0)h^2(0)\ldots h^j(0)| = |1h(0)h^2(0)\ldots h^j(0)| = |h^j(1)|,$$

contradicts the fact that X is a proper suffix of $h^j(1)$. □

References

[1] Auerbach D 1990 Dynamical complexity of strange sets *Measures of Complexity and Chaos* ed N B Abraham *et al* (New York: Plenum) pp 203–7

[2] Auerbach D and Procaccia I 1990 Grammatical complexity of strange sets *Phys. Rev.* A **41** 6602–14

[3] Bar-Hillel Y, Perles M and Shamir E 1961 On formal properties of simple phrase structure grammars *Z. Phonetik. Sprachwiss. Komm.* **14** 143–72

[4] Chen X, Lu Q-H and Xie H-M 1993 Grammatical complexity of Feigenbaum attractor *Preprint* (an English abstract is in *Advances in Mathematics (China)* **22** 185–6)

[5] Chomsky N 1956 Three models for the description of languages *IRE Trans. Information Theory* **2** 113–24

[6] Chomsky N 1959 On certain formal properties of grammars *Inf. Control* **2** 137–67

[7] Collet P and Eckmann J-P 1980 *Iterated Maps on the Interval as Dynamical Systems, Progress in Physics* vol 1 (Boston, MA: Birkhäuser)

[8] Crutchfield J P 1994 The calculi of emergence: computation, dynamics and induction *Physica* D **75** 11–54

[9] Crutchfield J P and Young K 1990 Computation at the onset of chaos *Complexity, Entropy, and Physics of Information* ed W Zurek (New York: Addison-Wesley) pp 223–69

[10] Grassberger P 1986 Toward a quantitative theory of self-generated complexity *Int. J. Theor. Phys.* **25** 907–38

[11] Hao B-L 1991 Symbolic dynamics and characterization of complexity *Physica* D **51** 161–76

[12] Herman G T and Rozenberg G 1975 *Developmental Systems and Languages* (Amsterdam: North-Holland)

[13] Hofbauer F 1980 The topological entropy of the transformations $x \mapsto ax(1-x)$ *Monatsh. Math.* **90** 117–41

[14] Hopcroft J E and Ullman J D 1979 *Introduction to Automata Theory, Languages and Computation* (Reading MA: Addison-Wesley)

[15] Lyubich M and Milnor J 1993 The Fibonacci unimodal map *J. Am. Math. Soc.* **6** 425–57

[16] de Melo W and van Strien S 1993 *One-Dimensional Dynamics* (Berlin: Springer)

[17] Ogden W 1968 A helpful result for proving inherent ambiguity *Math. Syst. Theory* **2** 191–4

[18] Procaccia I, Thomae S and Tresser C 1987 First-return maps as a unified renormalisation scheme for dynamical systems *Phys. Rev.* A **35** 1884–900

[19] Rozenberg G and Salomaa A eds 1997 *Handbook of Formal Languages* vol 1–3 (New York: Springer)

[20] Shyr H-J 1991 *Free Monoids and Languages* (Taichung, Taiwan: Hon Min Book Co.)

[21] Vul E B, Sinai Y G and Khanin K M 1984 Feigenbaum universality and thermodynamic formation *Usp. Mat. Nauk* **39** 3–37

[22] Wang Y and Xie H-M 1994 Grammatical complexity of unimodal maps with eventually periodic kneading sequences *Nonlinearity* **7** 1419–36

[23] Wolfram S 1984 Computation theory of cellular automata *Commun. Math. Phys.* **96** 15–57

[24] Xie H-M 1993 On formal languages of one-dimensional dynamical systems *Nonlinearity* **6** 997–1007

[25] Xie H-M 1995 Distinct excluded blocks and grammatical complexity of dynamical systems *Complex Syst.* **9** 73–90

[26] Xie H-M 1996 Fibonacci sequences and homomorphisms of free submonoid for unimodal maps *Nonlinearity* **9** 1469–87

[27] Xie H-M 1996 *Grammatical Complexity and One-Dimensional Dynamical Syst.* (Singapore: World Scientific)