Complex Systems **9** (1995) 73–90

Distinct Excluded Blocks and Grammatical Complexity of Dynamical Systems

Huimin Xie
Department of Mathematics,
Suzhou University, Suzhou 215006, China

Abstract. The concept of distinct excluded block (DEB) is defined for languages generated in studies of dynamical systems. The relationship of languages L and L'', which is the set of all DEBs of L, is analyzed in the context of Chomsky's hierarchy. Some further results of unimodal maps on interval are obtained.

1. Introduction

The concept of a distinct excluded block (DEB) first appeared in [21] and is used there as well as other means to characterize the complexity of languages (or sets of strings) generated by cellular automata (CA). The concept also appears in other places, such as [1, 2], under the different names of "forbidden blocks" and "irreducible forbidden blocks." From results in [21, 1, 2] and others, it seems that using this concept in the study of grammatical complexity in dynamical systems may become a new and prosperous approach.

For a given language, that is, a set of strings, there are two ways to logically analyze its structure. One way is to study the blocks that appear in it, the other way is to study the blocks that do not appear. In this paper we show that for languages of a certain class D the second approach is very useful. The main tool used is the concept of DEBs.

The purpose of this paper is to establish a theoretical framework to answer the following questions.

1. Where does the concept of a DEB come from?
2. How is this concept characterized and used to find these blocks for a given language?
3. If the set of all DEBs for a language L is denoted by L'', then what is the connection between L and L''?
4. If Chomsky's hierarchy is used when studying the complexity of languages, then what is the relationship of the levels of L and L'' in that hierarchy?

We discuss these questions in sections 2–4.

In section 5 we discuss these questions for unimodal maps on interval and obtain further results. Examples are given to show that this new method is useful in the study of dynamical systems.

2. Languages of class D and distinct excluded blocks

Now we will use some terminology (or notions) from computation theory, that is, formal languages and automata, which can be found in [10].

Let S be an alphabet (i.e., a finite set of symbols), and let S^* be the set of all finite strings over S. Any subset of S^* is called a (formal) language over S.

A useful operation π on strings will be used as follows: if x is a nonempty string, then $x\pi$ is the string formed from x by deleting its last letter.

Since it is obvious that not every formal language can be used to describe dynamical systems, the first question we consider is: what are the features of those languages that appear in studies of dynamical systems? The answer is summarized in Definition 1, where the letter "D" is used to remind us of dynamical systems.

Definition 1. A Language $L \subset S^*$ is called of class D, if it satisfies the conditions:

D1 a string $z \in L$ if and only if every substring of z belongs to L;

D2 if $z \in L$, then there exists a symbol $a \in S$ such that $za \in L$.

Remark 1. From D1 we can see that ε, the empty string, must be a string of every language of class D.

It seems that the conditions D1 and D2 are ubiquitous in studies of dynamical systems. Similar conditions are mentioned in [15, 14]. In [15] we read that "In the applications to dynamical theory the sequences admitted are not formed freely from the generating symbols, but are subject to certain limitations." Four conditions of admissibility were listed there for the problem presented. In [14] the names "factorial language" and "prolongable language" are used with respect to the conditions D1 and D2 in Definition 1.

If we denote the complement of L in S^* by

$$L' = S^* - L$$

then an equivalent description of condition D1 is

D1′ if $z \in L'$, then $xzy \in L'$ for any $x, y \in S^*$.

This can also be written as

$$L' = S^* L' S^*.$$

This is the essential content of Lemma 1.1 in [14].

The main concept of this paper is stated in Definition 2.

Definition 2. A string x is a DEB of language L, if $x \in L'$ but every proper substring of x belongs to L.

Let L'' be the set of all DEBs of language L. Following [21], a language L is called a finite (infinite) complement language if L'' is finite (infinite).

Showing the existence of a DEB is simple, for example, the shortest string of L' is a DEB if $L' \neq \emptyset$. But the DEBs found may be useless in the study of L. For languages of class D, however, the situation is quite different as Proposition 1 shows.

Proposition 1. If a language L satisfies condition D1, then

$$L' = S^* L'' S^*,$$

that is, for every $x \in L'$ there exists a substring y of x such that y is a DEB of L.

Proof. As every string of L' must contain a DEB as its substring, we have

$$L' \subset S^* L'' S^*.$$

This is true for any L. On the other hand, from the property D1′ we also have

$$L' \supset S^* L'' S^*,$$

which completes the proof. ■

Remark 2. From Proposition 1 it is clear that L is uniquely determined by L''. Although we can have DEBs and L'' for any language L, in a general sense they cannot characterize L completely.

The next question is: which set (of strings) can be the set of DEBs of a formal language?

Proposition 2. There exists a language L of class D for a given set $U \in S^*$ such that $L'' = U$ if and only if the set U satisfies the following conditions.

1. No string in U is a proper substring of another string in U.
2. If $\#S = n$, and $x_1, \ldots, x_n$ are n strings in U with n different last symbols, and x_n is the longest string among them, then at least one string among $x_1\pi, \ldots, x_{n-1}\pi$ is not a suffix of $x_n\pi$.

Proof. Since the "only if" part is obviously true, we just give the proof of the "if" part. Using the subset U of S^*, we construct a set L by

$$L = S^* - S^* U S^*.$$

We will show that this L is a language of class D and $L'' = U$.

First assume the contrary, that the condition D1 is not satisfied by L, then there is a string $x \in L$ and $x = uvw$ with $v \notin L$. But then $v \in L'$ and $v = abc$ with $b \in U$ by the condition $L' = S^*US^*$. Writing $x = (ua)b(cw)$ shows that $x \in L'$, which contradicts $x \in L$.

Now assume that L does not satisfy D2, that is, there exists a string $x \in L$, but $xa \notin L$ for any $a \in S$. Using Proposition 1, we obtain a suffix of xa that is a DEB of L for every $a \in L$. These DEBs will provide the strings $x_1, \ldots, x_n$ $(\#S = n)$ required in the second condition and leads to contradiction.

The proof of $L'' = U$ is easy and omitted. ■

After we introduce the following operators on languages (e.g., [10]):

$$\begin{aligned} MIN(L) &= \{x \in L \mid \text{no proper prefix of } x \text{ is in } L\};\\ MIN'(L) &= \{x \in L \mid \text{no proper suffix of } x \text{ is in } L\};\\ R(L) &= \{x \mid x^R, x \text{ is written backward, belongs to } L\}; \end{aligned}$$

it is easy to establish that

$$MIN'(L) = R \circ MIN \circ R(L),$$

and can have Proposition 3.

Proposition 3. If L is a language of class D, then

$$\begin{aligned} L'' &= MIN \circ MIN'(S^* - L)\\ &= MIN' \circ MIN(S^* - L)\\ &= MIN'(S^* - L) \cap MIN(S^* - L). \end{aligned}$$

Since these results are technical we only sketch the procedure of finding a DEB from a given string x of L'. First, using Remark 1, we know that $x \neq \varepsilon$, the empty string. Then we delete the last symbol of x and denote the string thus formed by $x\pi$. If $x\pi \in L'$ we repeat this operation, otherwise we do the same thing from the first symbol of this string. Finally a substring y of x is obtained such that deleting either the first symbol or the last symbol of y will generate a string belonging to L, and $y \in L'$. Using D1 again, every proper substring of y belongs to L. This proves that y is a DEB. The operations above can be expressed as MIN and MIN'.

3. L and L'' in Chomsky's hierarchy

In this section we analyze levels of languages L and L'' in the context of Chomsky's hierarchy [10].

There are four levels in this hierarchy: regular languages, context-free languages, context-sensitive languages, and recursively enumerable languages. For simplicity we use the abbreviations RGL, CFL, CSL, and REL in this paper. We also need the languages beyond them: nonrecursively enumerable

languages and recursive languages. Their abbreviations are nonREL and RL. If the corresponding sets of these languages are denoted by $\mathcal{L}(\cdot)$, then we have

$$\mathcal{L}(\text{RGL}) \subset \mathcal{L}(\text{CFL}) \subset \mathcal{L}(\text{CSL}) \subset \mathcal{L}(\text{RL}) \subset \mathcal{L}(\text{REL}),$$

and all these inclusions are proper.

For the rest of this paper all languages considered are of class D.

Proposition 4. A language L is a RGL if and only if its L'' is a RGL also.

Proof. Since the set $\mathcal{L}(\text{RGL})$ is closed under complementation, *MIN*, concatenation, and R (i.e., reversal), it is a consequence of Proposition 1 and 3. ■

Since any finite language is a RGL, we can obtain Corollary 1.

Corollary 1. A finite complement language is a RGL.

Remark 3. This fact is proved in [21] by other methods.

Using new results in [19, 13] it is easy to obtain the same conclusion as Proposition 4 for CSL.

There is no difficulty to prove Propositions 5 and 6.

Proposition 5. A language L is a CSL if and only if its L'' is a CSL also.

Proposition 6. A language L is a recursive language if and only if its L'' is a recursive language also.

Proposition 7. If both L and L'' are not recursive languages, then there are only three possibilities:

1. L is a REL, L'' is a nonREL;
2. L is a nonREL, L'' is a REL;
3. both L and L'' are nonREL.

Proof. Assume the contrary, that both L and L'' are REL. Using Proposition 1, we have $L' = S^*L''S^*$ and that L' is also a REL. From Theorem 8.3 in [10] both L and L' are recursive. Now using Proposition 3 L'' is recursive too. This leads to a contradiction and completes the proof. ■

We propose a conjecture for CFL.

Conjecture 1. A language L is a CFL if and only if its L'' is a CFL also.

We do not know if this conjecture is true or false, but at least it is possible that both L and L'' are CFL, but not RGL.

Example 1. Let a language K be given by

$$K = \{10^n10^n1^p \mid n \geq 0, p \geq 0\}.$$

Then a language L is defined as

$$L = \{x \mid x \text{ is a substring of } y \text{ for some } y \in K\}.$$

It is easy to show that this L is of class D.

Using the fact that the language

$$\{0^n10^n \mid n \geq 0\}$$

is a CFL, it is easy to prove that L is a CFL also. On the other hand, from the relation

$$L \cap 10^*10^*1 = \{10^n10^n1 \mid n \geq 0\}$$

with its right-hand side being not a RGL, we know that L is not a RGL.

Now we calculate for L the set

$$L'' = \{110\} \cup \{10^n10^m \mid m > n\} \cup \{0^n10^m1 \mid m < n\} \cup \{010^n10 | n > 0\}.$$

Since

$$L'' \cap 0^*10^*1 = \{0^n10^m1 \mid m < n\}$$

is not a RGL, we know that L'' is a CFL, but not a RGL.

Other examples related to Propositions 5 through 7 can be found in [11, 12]. As a matter of fact, the basic idea of Example 1 is taken from [11]. The results of this section can be summarized as shown in Figure 1.

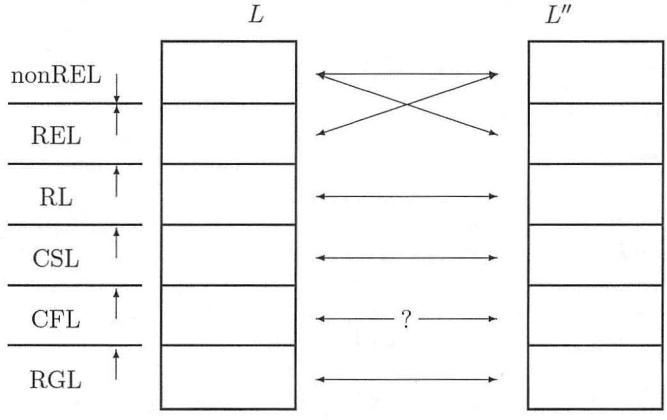

Figure 1: Levels of L and L'' in Chomsky's hierarchy

4. A natural equivalence relation

If a language $L \subset S^*$ is given, where S is an alphabet, then a natural equivalence relation R_L is introduced by L into S^*: for $x, y \in S^*$, xR_Ly holds if and only if for any $z \in S^*, xz \in L$ exactly when $yz \in L$.

Here are some basic results from [10].

1. The relation R_L is right-invariant, that is, if xR_Ly holds, then for any $z \in S^*$, xzR_Lyz holds also.
2. The language L is a RGL if and only if the index of R_L, that is, the number of equivalence classes of S^* with respect to R_L, is finite.
3. If L is a RGL, then the finite index of R_L is the number of states in the minimal deterministic finite automata (minDFA) accepting L.

These last two facts are usually referred to as the Myhill-Nerode theorem. Following [21], we may call the index N of R_L the regular language complexity of L.

In this section we show that the index N of R_L can be calculated from L''. If this N is finite, it is easy to obtain the minDFA. Some examples are given to explain this method.

Proposition 8. If L is given and L'' is its set of DEBs, and if a set is defined as

$$V = \{v \mid v \text{ is a proper prefix of } y \text{ for some } y \in L''\},$$

then for every $x \in L$ there exists a $v \in V$ such that

$$xR_Lv.$$

Proof. For a given string x there exists a suffix v of x such that $v \in V$. We require also that this v is the longest string which has these properties. There are two possibilities to be considered.

1. $v = \varepsilon$. We should prove that

$$xR_L\varepsilon$$

is true. Let z be a string. If $z(= \varepsilon z) \notin L$, then from D1′ we have $xz \notin L$. On the other hand, if $xz \notin L$ but $z \in L$, then using Proposition 1 there exists a decomposition

$$x = x_1x_2, \ z = z_1z_2$$

such that $x_2 \neq \varepsilon, z_1 \neq \varepsilon$, and $x_2z_1 \in L''$. But this x_2 is the suffix of x and belongs to the set V, a contradiction.

2. $v \neq \varepsilon$. Now we will show that

$$xR_Lv$$

is true. Let z be a string and discuss xz and vz. Since vz is a suffix of xz, it is obvious that $vz \notin L \Rightarrow xz \notin L$. If $xz \notin L$ but $z \in L$, then we have the same decomposition as in the first case. Since both v and x_2 are suffixes of x, and v is longer than x_2, then x_2 is a suffix of v. Now it is obvious that $x_2 z \notin L \Rightarrow vz \notin L$. ■

Proposition 9. The set L' is an equivalence class of R_L.

This is a trivial consequence of property D1′ in section 1. It means that

$$x, y \in L' \Longrightarrow xR_L y,$$

and

$$x \in L' \text{ and } xR_L y \Longrightarrow y \in L'.$$

Remark 4. From Propositions 8 and 9 we see that in order to calculate the index L of R_L, it only needs to work in the set V. For any pair $u, v \in V$ the following rule is useful:

$uR_L v$ if and only if for any $z \in S^*$, uz contains a DEB as its suffix $\Rightarrow$ $vz \in L'$ and vice versa.

Example 2. If $S = \{0, 1\}$ and L'=$\{0100,0010,001100\}$, calculate the minDFA accepting L.

First obtain the set V in Proposition 8:

$$V = \{\varepsilon, 0, 00, 001, 0011, 00110, 01, 010\}.$$

Using the rule in Remark 4, we have

$$001R_L 00110$$
$$01R_L 0011$$
$$010R_L 00110.$$

From these results and Propositions 8 and 9 we know that the index N of R_L is 6 and the six equivalence classes are

$$[\varepsilon], [0], [00], [001], [0011], \text{ and } L',$$

where $[x]$ is the class containing x.

Now it is routine to construct the minDFA for L as shown in Figure 2(a) (see also [21]).

In Figure 2 the final states are shown as filled circles. The unique nonfinal state is labeled by L'. As a matter of fact, this language L is the set of configurations after one time step of elementary CA of rule 222 (see [21] for these concepts).

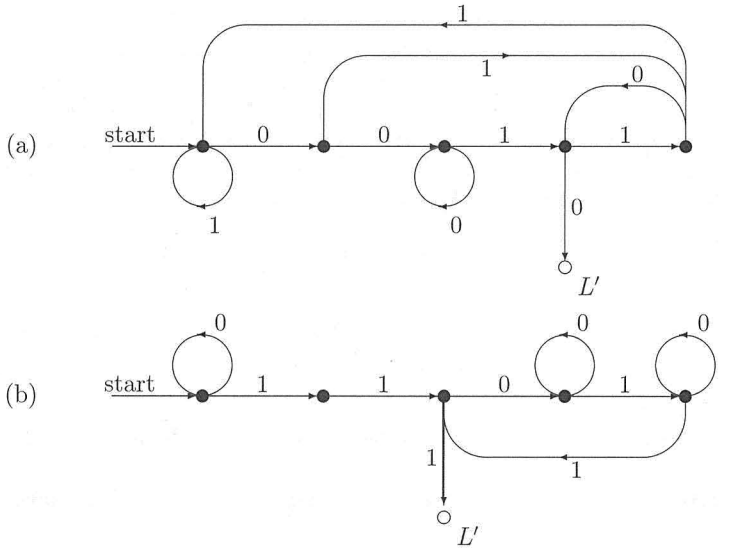

Figure 2: (a) minDFA for Example 3. (b) minDFA for Example 4.

Example 3. This is taken from [21] as the language of elementary CA of rule 18 after one time step. The set of DEBs is given by a regular expression (e.g., [10]):

$$L'' = 11(0^{\dagger}10^{\dagger}1)^{*}1.$$

Although the set V is infinite now, we can still use the rule in Remark 4 to determine the following equivalence classes:

$$[\varepsilon], [1], [11], [110], [1101], \text{ and } L'.$$

The minDFA is illustrated in Figure 2(b).

5. Languages of unimodal maps

In this section we discuss a subclass of languages of class D—the languages generated in studies of unimodal maps. The terminology used here can be found in [4, 8].

Here a unimodal map f is a continuous map from the interval [0,1] to itself, and satisfies the following conditions:

1. $f(0) = f(1) = 0$;
2. f reaches its maximum value at some interior point c of interval [0,1]; and
3. f is strictly monotonic on both subintervals $[0, c)$ and $(c, 1]$.

Using coarse-grained description, every orbit of f can be represented by an infinite string over an alphabet $\{0, 1, c\}$, where symbols 0 and 1 are used (instead of L and R as in [4, 8]) to denote the left and right sides of point c.

Let KS be the kneading sequence of f, that is, the coarse-grained description of the orbit of f starting from the point $f(c)$. It is well known that the dynamical behavior of f is determined by KS. Introducing ordering relations between strings and shift operator σ as in [4, 8], we know that KS is shift maximal. Now define the language of unimodal map f by

$$L = L(KS) = \{x \in \{0,1\} \mid x \text{ is a finite substring of } y \text{ for some admissible string } y\}.$$

Here an admissible string y is an infinite string satisfying the condition

$$\sigma^i(y) \leq KS \quad \forall i \geq 0.$$

It can be shown that this language reflects all symbolic dynamical behavior of f.

A convenient criterion for a string belonging to $L = L(KS)$ is Proposition 10.

Proposition 10. A string $x \in L = L(KS)$ if and only if every suffix of x, say v, satisfies the condition

$$v \leq KS.$$

Its proof is easy and omitted here (e.g., [22, 20]).

A notation of string $\hat{x}$ is used, where $\hat{x}$ is formed from a nonempty string x by changing its last symbol. For instance,

$$\hat{1} = 0, \hat{0} = 1, \widehat{1011} = 1010.$$

When we say a string x over $S = \{0, 1\}$ is even (or odd), it means that x contains an even or odd number of the symbol 1.

Proposition 11. A string x is a DEB of $L = L(KS)$ if and only if

1. $\hat{x}$ is an even prefix of KS and
2. no proper suffix of $\hat{x}$ is an even prefix of KS.

Proof. We first prove the "only if" part. If $x \in L''$, then from the definition of DEB every proper suffix of x belongs to L. Using Proposition 10 we see that

$$x > KS.$$

Combining this inequality with $x\pi \leq KS$ leads to the conclusion that both $\hat{x}$ and $x\pi$ belong to L. Since $x > \hat{x}$, then $\hat{x}$ is even.

If $\hat{x}$ has a proper suffix, say $\hat{v}$, which is an even prefix of KS, then v is a suffix of DEB x. Since v is odd and $v > \hat{v}$, we know that $v \in L'$, a contradiction to the definition of DEB.

Now we prove the "if" part. Since x is odd and $x > \hat{x}$, we have $x > KS$ and $x \in L'$. In order to prove that x is a DEB of L, it is enough to consider every proper prefix and proper suffix of x.

From the fact that $\hat{x}$ is a prefix of KS, its every proper prefix belongs to L. As to suffixes, assuming the contrary that x has a proper suffix $v \in L'$, then we have

$$v\pi \in L \text{ and } v > KS.$$

This leads to the statement that $\hat{v}$ is an even prefix of KS, which is a contradiction. ■

Many works have shown that if KS is periodic, or eventually periodic, then the language $L(KS)$ is regular (e.g., [6, 9] and references therein).

It is proved in [22, 20] that the converse is also true. This is shown in Theorem 1.

Theorem 1. The language $L(KS)$ is a RGL if and only if KS is periodic or eventually periodic.

We now consider the levels of L and L'' in Chomsky's hierarchy to continue the discussion from section 3.

Proposition 12. If $L = L(KS)$ and L'' is a CFL, then L'' is also a RGL.

Proof. Let $L'' = L(G)$, where G is a context-free grammar (CFG) (e.g., [10, 18]):

$$G = (V, S, P, s_0),$$

V is an alphabet of variables (nonterminals), S is an alphabet of terminals, P is a set of grammar rules, and s_0 is an initial symbol. Without loss of generality, we can assume that the grammar is reduced, that is, for each variable $x \neq s_0$, (i) s_0 generates a string containing x; and (ii) x generates a string ($\in L(G)$) (e.g., [18]).

Now we need a theorem from [18]: $L(G)$ is a RGL if and only if it is not self-embedding.

The definition of self-embedding is: a CFG is self-embedding if and only if $A \overset{*}{\Rightarrow} pAq$ for some $A \in V$ and $p, q \in (V \cup S)^*$ such that $p \neq \varepsilon$ and $q \neq \varepsilon$. A CFL L is self-embedding if and only if all CFG generating L are self-embedding.

If the grammar G in $L'' = L(G)$ is not self-embedding then our proof is completed. Assume the contrary, that this G is self-embedding. Then there is a variable A such that

$$A \overset{*}{\Longrightarrow} pAq,$$

and since G is reduced, and $\varepsilon \notin L''$ we have

$$p \neq \varepsilon, q \neq \varepsilon, \text{ and } p, q \in S^*.$$

Again using the fact that G is reduced, we obtain

$$s_0 \stackrel{*}{\Longrightarrow} wAx$$

and these strings w and x have the same properties as p, q. Now we can write

$$s_0 \stackrel{*}{\Longrightarrow} wp^n Aq^n x \in L'' \quad \forall n \geq 0.$$

Using Proposition 11 we have

$$KS = wp^{\infty}.$$

Finally, using Theorem 1 completes our proof. ■

A weaker conjecture than Conjecture 1 is as follows.

Conjecture 2. If $L = L(KS)$ is a CFL, then it is a RGL also.

Proposition 13. If $L = L(KS)$, its L'' is a REL, then both L and L'' are recursive languages.

Proof. From [10] we know there exists a Turing machine (TM) M as an enumerator or generator of L''. What we have to do is to construct another TM M' for L'' such that its strings can be generated in order of increasing size. This will complete our proof.

This TM can be designed as follows (see Figure 3). For each DEB x generated from M, the processor A can calculate all DEBs whose length is less than x and print out all new DEBs (if there are any) in the order of increasing size. This is possible by Proposition 11. At the same time the length of x is recorded and used to remember that all DEBs of length up to this number have been output already. It is obvious that the TM M', combined from M and A, is the required one. Using Proposition 6 completes our proof. ■

Corollary 2. If $L = \mathcal{L}(KS)$ is a nonREL, then its L'' is also a nonREL.

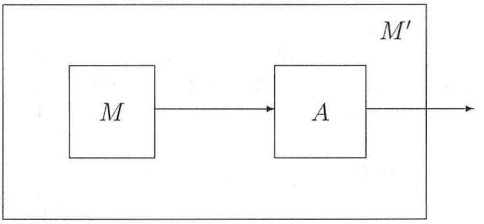

Figure 3: Turing machine for L'' being recursive. The box labeled with A is a processor.

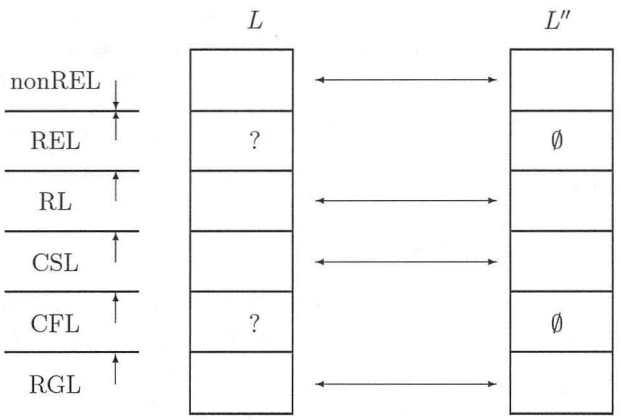

Figure 4: Levels of L and L'' for unimodal maps.

Here is another conjecture.

Conjecture 3. If $L = \mathcal{L}(KS)$ is a REL, then it is also a recursive language.

It is known that for languages of class D generated from CA this conjecture is false (e.g., [11]). These results can be summarized as shown in Figure 4.

Now we discuss what is meant when we say that a KS is given. From the viewpoint of the theory of computability (e.g., [16]), we have Definition 3.

Definition 3. A $KS = a_1 \dots a_n \dots$ is computable if the function

$$\begin{aligned} a : \mathbf{N} &\longrightarrow S = \{0, 1\} \\ n &\longmapsto a(n) = a_n \end{aligned}$$

is recursive.

Using the same idea as in the proof of Proposition 13, it is easy to obtain Proposition 14.

Proposition 14. The language $L = \mathcal{L}(KS)$ is recursive if and only if its KS is computable.

Proposition 15 clarifies the relation between KS and finite complement languages.

Proposition 15. The language $L = \mathcal{L}(KS)$ is finite complement if and only if KS is periodic or 10^∞.

Proof. The "only if" part is simple. If KS is 10^∞, then $L = S^*$ and $L' = \emptyset$, that is, its unimodal map is surjective. If $KS = x^\infty$, it is easy to show that any DEB is of length less than or equal to $|x|$ if x is even (or $2|x|$ if x is odd), so L'' is finite.

We now consider the "if" part. Since it is easy to establish that

$$KS = 10^\infty \text{ iff } L'' = \emptyset,$$
$$KS = 0^\infty \text{ iff } L'' = \{1\},$$
$$KS = 1^\infty \text{ iff } L'' = \{10\},$$

we can assume that $KS = 10^n 1 \ldots$ for some $n > 0$. Since L'' is finite, we denote the longest DEB by x. From Proposition 11, the string $\hat{x}$ is an even prefix of KS.

We will show that

$$KS = \hat{x}^\infty$$

and finish our proof. We need a lemma for this step, its proof will be given later.

Lemma 1. If $\hat{x}\alpha$ is an even prefix of KS and $\alpha \neq \varepsilon$, then α has a nonempty even suffix, which is a prefix of KS also.

Now decompose KS as

$$KS = \hat{x}b_1b_2 \cdots b_m \cdots,$$

where each b_i is the shortest even substring of KS, that is, either 0 or 10^p1 provided $0 \leq p \leq n$.

We prove inductively that for any integer m, the string $b_1b_2 \cdots b_m$ is always a prefix of KS. If this is true, then we have

$$\sigma^{|x|}(KS) = KS$$

and

$$KS = \hat{x}^\infty.$$

Proceeding inductively on m, for $m = 1$ using the Lemma 1 leads to $b_1 \neq 0$ (as KS begins from 1). If $b_1 = 10^p1$, then using Lemma 1 means that $p = n$ as required. Assume now that our statement is already true for m and consider the case of $m + 1$. Using Lemma 1 on $\alpha = b_1 \cdots b_{m+1}$ we see that there exists an integer k such that $b_{k+1} \cdots b_{m+1}$ is an even prefix of KS. From the inductive hypothesis the string $b_1 \cdots b_k$ is also a prefix of KS. We write KS as

$$KS = b_1 \cdots b_k\beta \cdots,$$

where the string β is of the same length as the string $b_{k+1} \cdots b_{m+1}$. Since KS is maximal shift we see that

$$\beta \leq b_{k+1} \cdots b_{m+1}.$$

On the other hand, we also have

$$b_1 \cdots b_k \cdots b_{k+1} \cdots b_{m+1} \leq b_1 \cdots b_k \beta,$$

and since every b_i is even, we have

$$b_{k+1} \cdots b_{m+1} \leq \beta.$$

Combining these inequalities leads to

$$b_{k+1} \cdots b_{m+1} = \beta$$

and completes our proof. ■

Remark 5. The conclusion of Proposition 15 can be obtained indirectly from Theorem 1 and from results in [20, 23].

Proof of Lemma 1. From the condition of the lemma we have $\hat{x}\hat{\alpha} > KS$ and $\hat{x}\hat{\alpha} \in L'$. Since the string $\hat{x}\hat{\alpha}$ cannot be a DEB itself, and $\hat{x}\alpha\pi \in L$, there exists a proper suffix of $\hat{x}\hat{\alpha}$ which is a DEB. Now it is enough to use the Proposition 11. ■

6. Examples

6.1 Three cases of regular languages

There are three cases of RGL in languages of unimodal maps.

1. $KS = 10^\infty$ and $L'' = \emptyset$;
2. KS is periodic and L'' is finite;
3. KS is eventually periodic, but not of the cases above, then L'' is a special semi-linear infinite set.

Some finite L'' are:

$KS = (101)^\infty, L'' = \{100\}$;
$KS = 10c, L'' = \{100\}$; and
$KS = (100)^\infty,\ L'' = \{1000, 10011, 100101\}$.

These include all period 3 KS that can occur in unimodal maps.

A more complex example is $KS = 10110(10111)^\infty$, which corresponds to a crisis phenomenon of a period 5 window in the Feigenbaum diagram (e.g., [8]). Its L'' is a so-called semi-linear set:

$$L'' = \{100\} \cup \{10110(10111)^n 0\}_{n\geq 0} \cup \{10110(10111)^n 11\}_{n\geq 0} \cup \{10110(10111)^n 1010\}_{n\geq 0}.$$

6.2 An example of a context-sensitive language

There are uncountably many languages beyond RGL in unimodal maps, a general discussion is being prepared. Here we only give the easiest example—the Feigenbaum attractor in unimodal maps.

Introducing a morphism h by

$$h = \{1 \longrightarrow 10, 0 \longrightarrow 11\}.$$

From [3] we have KS of the Feigenbaum attractor by

$$KS = \lim_{n\to\infty} h^n(1).$$

It was proved in [3] that this $L = \mathcal{L}(KS)$ is an extended table zero sided Lindenmayer (ET0L) language in Lindenmayer systems (e.g., [17]), also a CSL, but not a CFL. In [5] the same problem was discussed, but the language defined there is formed by taking all substrings of KS, thus reflecting only dynamical behaviors on the attractor itself.

It is easy to obtain L'' for this language:

$$L'' = \{h^n(100)\}_{n\geq 0} \cup \{h^n(10110)\}_{n\geq 0}.$$

If we apply the method of section 4 to this L'', an infinite automaton like Figure 4 in [7] will be obtained. Although the number of states in this automaton is infinite, it has a simple recursive structure and is useful in the study of symbolic dynamics.

Acknowledgement

This research was jointly supported by the National Basic Research Project "Nonlinear Science" and Natural Science Foundation of Jiangsu Province.

References

[1] R. L. Adler, "Geodesic Flows, Interval Maps, and Symbolic Dynamics," in *Ergodic Theory, Symbolic Dynamics, and Hyperbolic Spaces*, edited by T. Bedford, M. Keane, and C. Series (Oxford University Press, Oxford, 1991).

[2] G. D'Alessandro and A. Politi, "Hierarchical Approach to Complexity with Applications to Dynamical Systems," *Physical Review Letters*, **64** (1990) 1609–1612.

[3] X. Chen, Q.-H. Lu, and H.-M. Xie, "Grammatical Complexity of Feigenbaum Attractor," *Preprint*, its abstract is in *Advances in Mathematics (China)*, **22** (1993) 185–186.

[4] P. Collet and J.-P. Eckmann, *Iterated Maps on the Interval as Dynamical Systems*, (Birkhäuser, Boston, 1980).

[5] J. P. Crutchfield and K. Young, "Computation at the Onset of Chaos," in *Computation, Entropy, and the Physics of Information,* SFI Studies in the Sciences of Complexity, vol. VIII, edited by W. H. Zurek (Addison-Wesley, Reading, MA, 1990).

[6] E. J. Friedman, "Structure and Uncomputability in One-dimensional Maps," *Complex Systems*, **5** (1991) 335–349.

[7] P. Grassberger, "On Symbolic Dynamics of One-humped Maps of the Interval," *Zeitschrift für Naturforschung*, **43a** (1988) 671–680.

[8] B.-L. Hao, *Elementary Symbolic Dynamics and Chaos in Dissipative Systems*, (World Scientific, Singapore, 1989).

[9] B.-L. Hao, "Symbolic Dynamics and Characterization of Complexity," *Physica D*, **51** (1991) 161–176.

[10] J. Hopcroft and J. Ullman, *Introduction to Automata Theory, Languages and Computation,* (Addison-Wesley, Reading, MA, 1979).

[11] L. P. Hurd, "Recursive Cellular Automata Invariant Sets," *Complex Systems*, **4** (1990) 119–129.

[12] L. P. Hurd, "Nonrecursive Cellular Automata Invariant Sets," *Complex Systems*, **4** (1990) 131–138.

[13] N. Immerman, "Nondeterministic Space is Closed under Complementation," *SIAM Journal of Computing*, **17** (1988) 935–938.

[14] A. de Luca and S. Varricchio, "Some Combinatorial Properties of Factorial Languages," in *Sequences*, edited by R. M. Capocelli (Springer-Verlag, New York, 1990).

[15] M. Morse and G. A. Hedlund, "Symbolic dynamics," *American Journal of Mathematics*, **60** (1938) 815–866.

[16] M. B. Pour-El and J. I. Richards, *Computability in Analysis and Physics* (Springer-Verlag, Berlin, 1989).

[17] G. Rozenberg and A. Salomaa, *The Mathematical Theory of L Systems* (Academic Press, New York, 1980).

[18] A. Salomaa, *Formal Languages* (Academic Press, Boston, 1973).

[19] R. Szelepcsényi, "The Method of Forcing for Nondeterministic Automata," *Bulletin of the European Association of Theoretical Computation Science*, **33** (1987) 96–100.

[20] Y. Wang and H.-M. Xie, "Grammatical Complexity of Unimodal Maps with Eventually Periodic Kneading Sequences," *Nonlinearity*, **7** (1994) 1419–1436.

[21] S. Wolfram, "Computational Theory of Cellular Automata," *Communications in Mathematical Physics*, **96** (1984) 15–57.

[22] H.-M. Xie, "On Formal Languages of One-dimensional Dynamical Systems," *Nonlinearity*, **6** (1993) 997–1007.

[23] H.-M. Xie, "The Finite Automata of Eventually Periodic Unimodal Maps on the Interval," *Journal of Suzhou University*, **9** (1993) 112–118.