LINDENMAYER SYSTEMS AND THEIR COMPLEXITY: A BRIDGE BETWEEN COMPUTER SCIENCE AND DEVELOPMENTAL BIOLOGY

Jingnan Xie
Millersville University of Pennsylvania, Computer Sciences Department
jingnan.xie@millersville.edu

ABSTRACT

Lindenmayer systems (L systems) are parallel rewriting systems that were originally introduced in 1968 to model the development of multicellular organisms [1]. Since then, various types of L systems have been developed and their applications are widely discussed. The impacts of L systems are on many areas including computer science, developmental biology, computer graphics, and abstract algebra. In this paper, sufficient conditions for a language predicate concerning L systems to be as hard as the universality problem ("$= \{0,1\}^*$") are presented. By applying these conditions, we develop a uniform method to show undecidability and complexity results for many problems on L systems simultaneously via highly efficient many-one reductions. These problems include equivalence and containment problems of several types, and language class comparison problems (e.g., does an arbitrary regular expression or context-free grammar generate an L system language?). These problems are important in the interdisciplinary study of computer science and developmental biology.

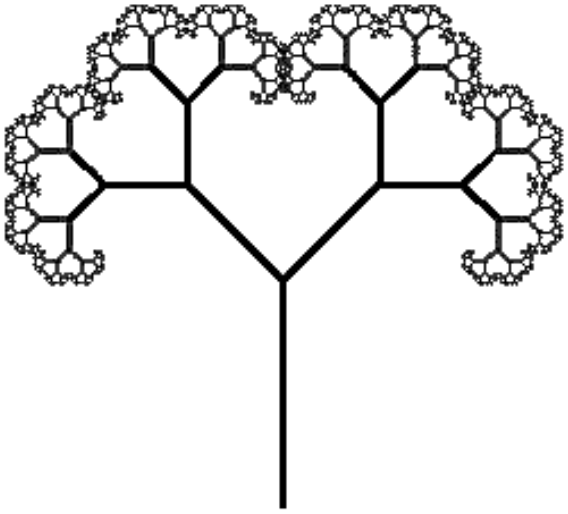

Figure 1: A "tree" generated by an L system

1 Introduction

Lindenmayer systems (L systems) are parallel rewriting systems that were originally introduced by Aristid Lindenmayer in 1968 to model the development of multicellular organisms [1]. Figure 1 is a tree structure created by an L system after several recursions.

Two main features brought about by the theory of L systems are

1. parallelism in the rewriting process, due to that languages were applied to model biological development in which parts of the developing organism change simultaneously;
2. the notion of a formal grammar conceived as a description of a dynamic process.

The latter feature makes L systems an important research topic in theoretical computer science. Since then, various types of L systems have been developed such as 0L, D0L, T0L, E0L, EDT0L, and ET0L systems (formal definitions are introduced in Section 2 of this paper), and their applications are widely discussed (for example, see [2] and [3]). The computational complexity of problems concerning L systems is also an important research topic in both theoretical computer science and developmental biology. For example, the membership and emptiness problems of E0L, ET0L, and EDT0L systems are discussed in [4], and the context-freeness, regularity, and 0L-ness[1] problems for E0L systems are shown to be undecidable in [5]. However, to our best knowledge, the 0L-ness problems for regular languages and context-free languages are still open due to very weak closure properties of 0L systems.

In this paper, sufficient conditions for a language predicate concerning L systems to be as hard as the universality problem ("$= \{0,1\}^*$") are presented. By applying these conditions, we develop a uniform method to show undecidability and complexity results for many problems on L systems simultaneously via highly efficient many-one reductions. For example, using the same proof, we show that the 0L-ness problem for $(\cup, \cdot, *)$-regular expressions is PSPACE-hard, and for context-free grammars is undecidable. These results

[1] 0L-ness problem is to ask whether a language is a 0L language.

are very important for interdisciplinary research in theoretical computer science and biology. The equivalence and containment problems for L systems are also studied in this paper. We investigate the problems of testing equivalence and containment to many fixed languages since these results are stronger and have more practical meanings. For example, we show that for any fixed unbounded regular set R_0, the predicates "$= R_0$" and "$\supseteq R_0$" are undecidable for EDT0L systems. This also shows that there is no approximating minimization algorithm between EDT0L systems and DFA accepting this fixed regular set R_0.

This paper is organized as follows.

In Section 2, we review the definitions of several widely discussed L systems. Some preliminary definitions and notations are also explained.

In Section 3, we study and summarize the complexity of the universality problem ("$= \{0,1\}^*$") for several classes of language descriptors.

In Section 4, sufficient conditions are given for a language predicate to be as hard as the language predicates "$= \{0,1\}^*$". These conditions yield a method to prove complexity results through highly efficient many-one reductions for L systems. Section 4.1 discusses the language class comparison problems. In Section 4.2, we study the equivalence and containment problems concerning L systems.

2 Definitions and Notations

In this section, we review the definitions of several types of L systems from [5]. A stronger form of non-recursive enumerability called productiveness and its importance are discussed. Several preliminary definitions and notations in language theory are also explained. The reader is referred to [6] for all unexplained notations and terminologies in language theory.

We use λ to denote the empty string and $\emptyset$ to denote the empty set. We use $\mathbb{N}$ to denote the set of natural numbers. Let **P** denote the class of sets that can be recognized in polynomial time by a deterministic Turing Machine. Let **PSPACE** denote the class of sets that can be recognized using polynomial space by a Turing Machine. Let **NEXPTIME** denote the class of sets that can be recognized in exponential time by a non-deterministic Turing Machine. We use **Co-NEXPTIME** to denote the set of the complements of the languages in **NEXPTIME**. If A is many-one reducible to B, we write $A \leqslant_m B$; If this reduction is polynomial-time bounded, we write $A \leqslant_{ptime} B$.

Let $\mathcal{D}$ be a class of language descriptors that describe languages over Σ. In this paper, we only consider finite Σ. Then, $\forall d \in \mathcal{D}$, $\mathcal{L}(d) = \{w \in \Sigma^* \mid w \text{ is described by } d\}$ and $\mathcal{L}(\mathcal{D}) = \{L \subseteq \Sigma^* \mid \exists d \in \mathcal{D}$ such that $L = \mathcal{L}(d)\}$. $\forall d \in \mathcal{D}$, let $|d|$ denote the size of d. The size of a context-free grammar is the number of symbols of all its productions. For example, the following context-free grammar d accepts the language $\{0,1\}^*$. $d = (\{s_1\}, \{0,1\}, \{(s_1, 0s_1), (s_1, 1s_1), (s_1, \lambda)\}, s_1)$. The size of d is 8 (denoted by $|d| = 8$).

A language class comparison problem is defined as follows: for two classes of language descriptors $\mathcal{D}_1$ and $\mathcal{D}_2$, determine for any $a \in \mathcal{D}_1$, whether $\mathcal{L}(a) \in \mathcal{L}(\mathcal{D}_2)$?

Definition 1. An *non-deterministic 1-reversal bounded 1-counter machine* (denoted by N 1-rbd 1-CM) is a pushdown automaton where the cardinality of the stack alphabet is two(including the bottom symbol) and the machine makes at most one single reversal on the stack. Hence, the class of languages accepted by N 1-rbd 1-CMs is a proper subset of linear context-free languages. Throughout the paper, we use **N11CM** to denote the set of N 1-rbd 1-CMs with input alphabet $\{0,1\}$. □

Definition 2. A *finite substitution* σ over alphabet Σ is a mapping of Σ^* into the set of all finite nonempty languages (possibly over another alphabet Δ) defined as follows. For each letter $a \in \Sigma$, $\sigma(a)$ is a finite nonempty language, $\sigma(\lambda) = \{\lambda\}$ and for all $w_1, w_2 \in \Sigma^*$,

$$\sigma(w_1 w_2) = \sigma(w_1)\sigma(w_2).$$

For any language L over Σ, $\sigma(L) = \bigcup_{\forall w \in L} \sigma(w)$.

If $\forall a \in \Sigma$, $\lambda \notin \sigma(a)$, the substitution σ is referred to as *λ-free or non-erasing*. If each $\sigma(a)$ contains a single string, σ is called a *morphism*. □

In this paper, we only consider L systems over the terminal alphabet $\{0,1\}$. This restriction has been taken into account in the following definitions.

Definition 3. A *0L system* is a triple $G = (\{0,1\}, \sigma, s)$ where σ is a finite substitution over $\{0,1\}$ and $s \in \{0,1\}^*$ is the axiom. The 0L system G generates the language

$$\mathcal{L}(G) = \{s\} \cup \sigma(s) \cup \sigma(\sigma(s)) \cup ... = \bigcup_{i \geq 0} \sigma^i(s).$$

A 0L system is *deterministic* or a *D0L system* if and only if σ is a morphism. □

The letter E ("extended") in the name of an L system means that the use of nonterminals is allowed. Thus, an E0L system is a 0L system augmented with nonterminals.

Definition 4. An *E0L system* is a 4-tuple $G = (\{0,1\}, V, \sigma, s)$ where V is the set of nonterminals (disjoint with $\{0,1\}$), σ is a finite substitution over $V \cup \{0,1\}$ and $s \in (V \cup \{0,1\})^*$ is the axiom. The E0L system G generates the language

$$\mathcal{L}(G) = \bigcup_{i \geq 0} \sigma^i(s) \cap \{0,1\}^*.$$

An E0L system is *deterministic* or a *ED0L system* if and only if σ is a morphism. □

The letter T ("table") in the name of an L system means in-

stead of having one finite substitution, the system has a finite number of finite substitutions.

Definition 5. A *T0L system* is a triple $G = (\{0,1\}, P, s)$ where P is a finite set of finite substitutions such that for each $\sigma \in P$, $(\{0,1\}, \sigma, s)$ is a 0L system. For a T0L system $G = (\{0,1\}, P, s)$,

1. let $X = x_1x_2...x_k$ $(k \geq 1)$ where x_i $(1 \leq i \leq k) \in \{0,1\}$. Let σ be a finite substitution in P and let $Y \in \{0,1\}^*$. We write $X \rightarrow_\sigma Y$ if there exists $y_1, y_2, ..., y_k \in \{0,1\}^*$ such that $y_i \in \sigma(x_i)$ $(1 \leq i \leq k)$ and $Y = y_1y_2...y_k$. We write $X \rightarrow_P Y$ if there exists $\sigma \in P$ such that $X \rightarrow_\sigma Y$;
2. $\rightarrow_P^*$ denotes the transitive and reflexive closure of the binary relation $\rightarrow_P$; and
3. $\mathcal{L}(G) = \{w \in \{0,1\}^* \mid s \rightarrow_P^* w\}$.

An *ET0L system* is a 4-tuple $G = (\{0,1\}, V, P, s)$ where V is the set of nonterminals (disjoint with $\{0,1\}$), P is a finite set of finite substitutions over $V \cup \{0,1\}$ and $s \in (V \cup \{0,1\})^*$ is the axiom. For a ET0L $G = (\{0,1\}, V, P, s)$,

1. let $X = x_1x_2...x_k$ $(k \geq 1)$ where x_i $(1 \leq i \leq k) \in (V \cup \{0,1\})$. Let σ be a finite substitution in P and let $Y \in (V \cup \{0,1\})^*$. We write $X \rightarrow_\sigma Y$ if there exists $y_1, y_2, ..., y_k \in (V \cup \{0,1\})^*$ such that $y_i \in \sigma(x_i)$ $(1 \leq i \leq k)$ and $Y = y_1y_2...y_k$. We write $X \rightarrow_P Y$ if there exists $\sigma \in P$ such that $X \rightarrow_\sigma Y$;
2. $\rightarrow_P^*$ denotes the transitive and reflexive closure of the binary relation $\rightarrow_P$; and
3. $\mathcal{L}(G) = \{w \in \{0,1\}^* \mid s \rightarrow_P^* w\}$.

An ET0L system is *deterministic* or an *EDTOL system* if every finite substitution in P is a morphism.

□

For a better understanding of these definitions, we give several examples here.

Example 2.1. Let the D0L system $G = (\{0,1\}, h, 01)$ with $h(0) = \{0\}$ and $h(1) = \{01\}$.
Hence, $h(01) = \{001\}$, $h(h(01)) = \{0001\}$, $h(h(h(01))) = \{00001\}$, ...
Then, $\mathcal{L}(G) = \{0^n1 \mid n \geq 1\}$.

Example 2.2. Let the 0L system $G = (\{0,1\}, h, 0)$ with $h(0) = \{\lambda, 1, 0, 00, 01\}$ and $h(1) = \{1, 10, 11\}$. Then $\mathcal{L}(G) = \{0,1\}^*$.

Example 2.3. Let the EDT0L system $G = (\{0,1\}, \{A, B, C, D\}, P, CD)$ where $P = \{h_1, h_2, h_3\}$ and
$h_1(0) = \{0\}$, $h_1(1) = \{1\}$, $h_1(A) = \{A\}$, $h_1(B) = \{B\}$, $h_1(C) = \{ABC\}$, $h_1(D) = \{DA\}$;
$h_2(0) = \{0\}$, $h_2(1) = \{1\}$, $h_2(A) = \{A\}$, $h_2(B) = \{B\}$, $h_2(C) = \{CB\}$, $h_2(D) = \{D\}$;
$h_3(0) = \{0\}$, $h_3(1) = \{1\}$, $h_3(A) = \{0\}$, $h_3(B) = \{1\}$, $h_3(C) = \{\lambda\}$, $h_3(D) = \{\lambda\}$.
Then $\mathcal{L}(G) = \{0^n1^m0^n \mid n \geq 0, m \geq n\}$.

The following figure shows the relationship between several classes of languages discussed in this paper. REG, Lin-CFL, and CFL denote the class of regular languages, linear context-free languages, and context-free languages respectively. In this figure, $A \rightarrow B$ means B properly contains A and if no line between A and B, it means A and B are incomparable.

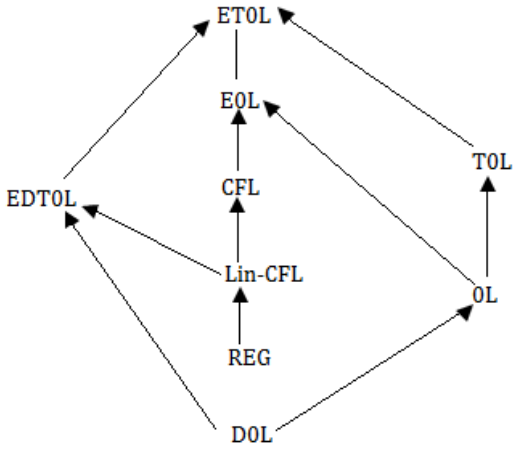

Figure 2: Relationship between Several Classes of Languages

Productive sets and their properties are a standard topic in mathematical logic/recursion theory textbooks such as [7] and [8]. Productiveness is a recursion-theoretic abstraction of what causes Gödel's first incompleteness theorem to hold. Definition 6 recalls the definition of a productive set on $\mathbb{N}$, as developed in [7].

Definition 6. Let W be an effective Gödel numbering of the recursively enumerable sets. A set A of natural numbers is called *productive* if there exists a total recursive function f so that for all $i \in \mathbb{N}$, if $W_i \subseteq A$ then $f(i) \in A - W_i$. The function f is called the *productive function* for A. □

From this definition, we can see that no productive set is recursively enumerable. It is well-known that the set of all provable sentences in an effective axiomatic system is always a recursively enumerable set. So for any effective axiomatic system F, if a set A of Gödel numbers of true sentences in F is productive, then there is at least one element in A which is true but cannot be proven in F. Moreover, there is an effective procedure to produce such an element.

Let W be an effective Gödel numbering of the recursively enumerable sets. $\mathbf{K}$ denotes the set $\{i \in \mathbb{N} \mid i \in W_i\}$. $\overline{\mathbf{K}}$ denotes the set $\{i \in \mathbb{N} \mid i \notin W_i\}$. Two well-known facts

of productive sets (see [7]) that are necessary for the research developed here are as follows:

Proposition 1. 1. $\overline{\mathbf{K}}$ is productive.

2. For all $A \subseteq \mathbb{N}$, A is productive if and only if $\overline{\mathbf{K}} \leq_m A$.

□

The following proposition is proved in [9] and is used to prove productiveness results. It also shows in which way the productiveness is stronger than non-recursive enumerability, i.e., every productive set A has an infinite recursively enumerable subset, and for any sound proof procedure P, one can effectively construct an element that is in A, but not provable in P.

Proposition 2. Let $A \subseteq \Sigma^*$, $B \subseteq \Delta^*$, and $A \leq_m B$. Then, the following hold:

1. If A is productive, then so is B.
2. If A is productive, then there exists a total recursive function $\Psi : \Sigma^* \to \Sigma^*$, called a *productive function for A*, such that for all $x \in \Sigma^*$,

   $\mathcal{L}(M_x) \subseteq A \Rightarrow \Psi(x) \in A - \mathcal{L}(M_x)$, where $\{M_x \mid x \in \Sigma^*\}$ is some Gödel-numbering of Turing machines over alphabet Σ.
3. If A is productive, then A is not recursively enumerable (RE). However, A does have an infinite RE subset.

□

3 The Universality Problem

The widely discussed universality problem ("$= \{0,1\}^*$") plays an important role in this research. Therefore, in this section, we study and summarize the complexity of "$= \{0,1\}^*$" for several classes of language descriptors. To make our results stronger and more applicable, we investigate the complexity of a restricted version of "$= \{0,1\}^*$":

> testing equivalence to $\{0,1\}^*$ for languages whose complements' cardinalities are less than or equal to one (denoted by "$= \{0,1\}^* \mid_{|L^c| \leq 1}$").

The instances of this restricted predicate have very important semantic properties: they are the simplest regular sets. These restrictions make the predicate more widely applicable: for example, they directly apply to promise problems, predicates on regular sets, and descriptional complexity of language descriptors.

In [9], the predicate "$= \{0,1\}^* \mid_{|L^c| \leq 1}$" is shown to be productive for N 1-rbd 1-CMs. And it is easy to see that EDT0L, E0L, and ET0L systems efficiently contain linear context-free grammars. Hence, for EDT0L, E0L, and ET0L systems, the predicate "$= \{0,1\}^* \mid_{|L^c| \leq 1}$" is also productive, hence, non-recursively enumerable. Theorem 3.1 summarizes this result.

Theorem 3.1. There exists a subset D of **N11CM($\{0,1\}$)** (**EDT0L($\{0,1\}$)**, **E0L($\{0,1\}$)**, or **ET0L($\{0,1\}$)**) such that

1. $D \in \mathbf{P}$;
2. $\forall d \in D$, $\mathcal{L}(d) \subseteq \{0,1\}^*$ and $|\{0,1\}^* - \mathcal{L}(d)| \leq 1$; and
3. $\overline{\mathbf{K}} \leqslant_{ptime} \{d \mid d \in D, \mathcal{L}(d) = \{0,1\}^*\}$, hence, it is non-recursively enumerable.

The following theorem is from [10] and shows that the predicate "$= \{0,1\}^* \mid_{|L^c| \leq 1}$" is PSPACE-hard for $(\cup, \cdot, *)$-regular expressions.

Theorem 3.2. There exists a subset R of **REG($\{0,1\}$)** such that

1. $R \in \mathbf{P}$;
2. $\forall d \in R$, $\mathcal{L}(d) \subseteq \{0,1\}^*$ and $|\{0,1\}^* - \mathcal{L}(d)| \leq 1$; and
3. **PSPACE** $\leqslant_{ptime} \{d \mid d \in R, \mathcal{L}(d) = \{0,1\}^*\}$.

Besides the universality problem, our proof technique can also be applied to reductions of other sources. For example, it may be practically more relevant to only consider finite languages. One theorem in [11] states that the predicate "$= \{0,1,\lambda\}^{2^{cn}}$" is Co-NEXPTIME-hard for context-free grammars generating finite languages. Here we state that theorem with a slight modification. The proof is the same as in [11]. Let CFG_{fin} be the set of context-free grammars over terminal alphabet $\{0,1\}$ generating finite languages.

Theorem 3.3. [11] There exists a constant $c > 0$ such that $\mathbf{Co-NEXPTIME} \leqslant_{ptime} \{d \mid d \in CFG_{fin}, \mathcal{L}(d) = \{0,1,\lambda\}^{2^{cn}} \text{ where } n = |d|\}$. □

4 Computational Complexity of L Systems

4.1 Language Class Comparison Problems

Language class comparison problems involving L systems are studied by many groups. For example, in [5], the context-freeness, regularity, and 0L-ness problems for E0L systems are shown to be undecidable. However, to our best knowledge, the 0L-ness problems for regular languages and context-free languages are still open. The 0L-ness problem is hard to study for mainly two reasons.

1. It is known that 0L languages and finite languages are incomparable. Then proof techniques in [12] and [13] do not apply to the 0L-ness problem since they require the predicates to be true for all regular sets.
2. 0L systems are shown to be an anti-AFL [14]. So they have almost no closure properties.

In contrast, the properties of the predicate "$= \{0,1\}^* \mid_{|L^c| \leq 1}$" (compared to the predicate "$= \{0,1\}^*$") enable us to investigate 0L-ness problem for many classes

of languages. In this section, we develop a new method to study the complexity/productiveness of the 0L-ness problem. Through this method, we show the 0L-ness problem is productive for N 1-rbd 1-CMs, linear context-free grammars, context-free grammars, EDT0L systems, E0L systems, and ET0L systems. Additionally, we show the 0L-ness problem for $(\cup, \cdot, *)$-regular expressions is PSPACE-hard and for context-free grammars generating finite languages is Co-NEXPTIME-hard. The following theorem is the key result to this new method.

Theorem 4.1. For any $w \in \{0,1\}^+$ such that $|w| \geq 4$, the language $L_w = \{0,1\}^* - \{w\}$ is not a 0L language.

Proof. Assume L_w is a 0L language, then there exists a 0L system $G = (\{0,1\}, \sigma, s)$ such that $\mathcal{L}(G) = L_w$.

1. If $|s| = 1$, let $w = w_1 w_2$ where $|w_1| \geq 2$ and $|w_2| \geq 2$. Then we know $w_1 \neq s$, $w_2 \neq s$ and $w_1, w_2 \in L_w$. $\Rightarrow$ There exist $t_1, t_2 \in L_w$ such that $w_1 \in \sigma(t_1)$ and $w_2 \in \sigma(t_2)$. $\Rightarrow w = w_1 w_2 \in \sigma(t_1 t_2)$. If $t_1 t_2 \neq w$, then $t_1 t_2 \in L_w$. $\Rightarrow w \in L_w \Rightarrow$ Contradiction. If $t_1 t_2 = w$, since $\lambda \in L_w$, there exists $t_3 \in L_w (t_3 \neq \lambda)$ such that $\lambda \in \sigma(t_3)$. $\Rightarrow w \in \sigma(t_1 t_2 t_3)$ and $t_1 t_2 t_3 \neq w$. $\Rightarrow w \in L_w \Rightarrow$ Contradiction.
2. If $|s| > 1$, let $w = w_1 w_2 w_3 ... w_k$ where w_i $(1 \leq i \leq k) \in \{0,1\}$. $\Rightarrow w_i$ $(1 \leq i \leq k) \in L_w$. $\Rightarrow \exists t_1, t_2, ..., t_k \in L_w$ such that $w_i \in \sigma(t_i)$ $(1 \leq i \leq k)$. $\Rightarrow w = w_1 w_2 ... w_k \in \sigma(t_1 t_2 ... t_k)$. If $t_1 t_2 ... t_k \neq w \Rightarrow t_1 t_2 ... t_k \in L_w \Rightarrow w \in L_w \Rightarrow$ Contradiction. If $t_1 t_2 ... t_k = w$, since $\lambda \in L_w$, there exists $t_{k+1} \in L_w (t_{k+1} \neq \lambda)$ such that $\lambda \in \sigma(t_{k+1})$. $\Rightarrow w \in \sigma(t_1 t_2 ... t_k t_{k+1})$ and $t_1 t_2 ... t_k t_{k+1} \neq w \Rightarrow w \in L_w \Rightarrow$ Contradiction.

□

The theorem below follows directly from Theorem 3.1 and Theorem 4.1.

Theorem 4.2. There exists a subset D of **N11CM** such that

1. $D \in \mathbf{P}$;
2. $\forall d \in D$, $\mathcal{L}(d)$ is co-finite; and
3. for any fixed set $\Gamma \subseteq \mathcal{L}(\mathbf{0L})$ and $\{0,1\}^* \in \Gamma$, $\overline{\mathbf{K}} \leqslant_{ptime} \{< d > \mid d \in D, \mathcal{L}(d) \in \Gamma\}$.

Proof. Let D be the same D defined in Theorem 3.1. Then $\forall d \in D$, for any fixed Γ, if $\mathcal{L}(d) = \{0,1\}^*$, $\mathcal{L}(d) \in \Gamma$; otherwise, $\mathcal{L}(d) = \{0,1\}^* - \{w\}$ where $|w| \geq 4$. So by Theorem 4.1, $\mathcal{L}(d)$ is not a 0L language. Hence, $\mathcal{L}(d) \notin \Gamma$. □

The following corollary is a corollary of this proof.

Corollary 1. The 0L-ness problem is productive for N 1-rbd 1-CMs, linear context-free grammars, context-free grammars, E0L systems, EDT0L systems, and ET0L systems.

Proof. In Example 2.2, we showed that $\{0,1\}^*$ is a 0L language. □

Recall that the predicate "$= \{0,1\}^*|_{|L^c| \leq 1}$" is PSPACE-hard for $(\cup, \cdot, *)$-regular expressions. So similarly, the following theorem holds.

Theorem 4.3. There exists a subset D of **REG{0,1}** such that

1. $D \in \mathbf{P}$;
2. $\forall d \in D$, $\mathcal{L}(d)$ is co-finite; and
3. for any fixed set $\Gamma \subseteq \mathcal{L}(\mathbf{0L})$ and $\{0,1\}^* \in \Gamma$, **PSPACE** $\leqslant_{ptime} \{< d > \mid d \in D, \mathcal{L}(d) \in \Gamma\}$.

Corollary 2. The 0L-ness problem for $(\cup, \cdot, *)$-regular expressions is PSPACE-hard.

One natural extension is to study 0L-ness problems for more restricted classes of language descriptors such as context-free grammars generating finite languages. Let CFG_{fin} denote the set of context-free grammars over terminal alphabet $\{0,1\}$ generating finite languages. Recall Theorem 3.3 which is proven in Hunt et al [11]. In its proof, the definitions of $INVALCM(w)$ and $VALCM(w)$ are used. Let M be a nondeterministic 2^{cn} time-bounded single tape Turing machine for some constant $c > 0$. Hunt et al constructed a context-free grammar G such that $\mathcal{L}(G) = INVALCM(w) \cap (\Delta_M \cup \{\lambda\})^{2^{c_0 n}}$ for some constant $c_0 > 0$. If M does not accept w, then $\mathcal{L}(G) = (\Delta_M \cup \{\lambda\})^{2^{c_0 n}}$. Otherwise, $\mathcal{L}(G) = (\Delta_M \cup \{\lambda\})^{2^{c_0 n}} - VALCM(w)$. It is clear that $VALCM(w)$ is finite. The proof in [11] shows that $\forall t \in VALCM(w)$, $|t| < 2^{c_0 n}$. We can efficiently code the language $\mathcal{L}(G)$ into alphabet $\{0,1\}$. The following theorem shows that when M accepts w, the coded $\mathcal{L}(G)$ is not a 0L language.

Theorem 4.4. Let $L_0 = \{0,1,\lambda\}^m - F$ for any integer $m > 0$ where

1. $F \neq \emptyset$ is a finite set over $\{0,1\}$;
2. for any $w \in F$, $|w| < m$; and
3. $L_0 \neq \{0,1\}^m$.

Then L_0 is not a 0L language.

Proof. Suppose L_0 is a 0L language. Then there exists a 0L system $G = (\{0,1\}, \sigma, s)$ such that $\mathcal{L}(G) = L_0$. For $A, B \in \{0,1\}^*$, we call $A \in \sigma(B)$ a growing rule in σ if $|A| > |B|$. We call $\lambda \in \sigma(A)$ a erasing rule in σ.
Observation: Any string over $\{0,1\}$ of length m is in L_0.
Claim 1: There is no growing rule in σ.
Proof of Claim 1: Suppose there is at least one growing rule in σ. Then we know $w_0 \in \sigma(0)$ or $w_0 \in \sigma(1)$ where $|w_0| > 1$. Hence $w_0^m \in \sigma(0^m)$ or $w_0^m \in \sigma(1^m)$. Since $0^m, 1^m \in L_0$, we know $w_0^m \in L_0$. Hence, there is a contradiction because $|w_0^m| > m$.

Claim 2: There is at least one erasing rule in σ.
Proof of Claim 2: By claim 1, we know $|s| = m$ where s is the axiom. Otherwise since there is no growing rule, every string in L_0 has length $< m$, a contradication. Suppose there is no erasing rule in σ, then any string in L_0 has length m. Since $L_0 \neq \{0,1\}^m$, L_0 must contain at least one string of length $< m$. Hence, there is a contradiction.
Proof of the theorem: For any string $w \in F$, since $|w| < m$, two strings $w0^{m-|w|}$ and $w1^{m-|w|}$ are in L_0. At least one of these two strings is not the axiom s. WLOG, we assume $w0^{m-|w|} \neq s$. Then there exists $w' \in L_0$ such that $w0^{m-|w|} \in \sigma(w')$.
By Claim 1, we know $|w'| = m$. Let $w' = w_1w_2$ where $|w_1| = |w|$. From Claim 1, we have $w \in \sigma(w_1)$. From Claim 2, $\lambda \in \sigma(0)$ or $\lambda \in \sigma(1)$ must be true. Then we know $w \in \sigma(w_1 0^{m-|w_1|})$ or $w \in \sigma(w_1 1^{m-|w_1|})$. Since $w_1 0^{m-|w_1|}, w_1 1^{m-|w_1|} \in L_0$, we have $w \in L_0$, a contradiction. □

Lemma 4.1. The language $L = \{0, 1, \lambda\}^m$ for any integer $m > 0$ is a 0L language.

Proof. Consider the 0L system $G = (\{0,1\}, \sigma, 1^m)$ where $\sigma(1) = \{0, 1, \lambda\}$. Clearly, $\sigma(1^m)$ contains every string of length m over $\{0,1\}$. Since $\lambda \in \sigma(1)$, we know $\lambda, 1, 11, 111, ..., 1^{m-1}$ are all in $\sigma(1^m)$. So $\mathcal{L}(G) = L$. □

By Theorem 4.4, Lemma 4.1 and Theorem 3.3, we have the following Co-NEXPTIME-hardness result.

Theorem 4.5. Co-NEXPTIME $\leqslant_{ptime} \{< d > \mid d \in CFG_{fin}, \mathcal{L}(d)$ is a 0L language$\}$.

Proof. Let M be a nondeterministic 2^{cn} time bounded single tape Turing machine for some constant $c > 0$. In the proof of Theorem 3.2 [11], a context-free grammar G is constructed such that $\mathcal{L}(G) = INVALCM(w) \cap (\Delta_M \cup \{\lambda\})^{2^{c_0 n}}$ for some constant $c_0 > 0$. If M does not accept w, then $\mathcal{L}(G) = (\Delta_M \cup \{\lambda\})^{2^{c_0 n}}$. By Lemma 4.1, $\mathcal{L}(G)$ is a 0L language. Otherwise, $\mathcal{L}(G) = (\Delta_M \cup \{\lambda\})^{2^{c_0 n}} - VALCM(w)$. By Theorem 4.4, $\mathcal{L}(G)$ is not a 0L language. □

4.2 Equivalence and Containment Problems

Equivalence and containment problems for L-systems are widely discussed and have many applications [14]. In this section, we study several types of equivalence and containment problems for E0L systems, EDT0L systems, and ET0L systems. These problems include testing equivalence to any fixed language with an unbounded regular subset, testing containment of any fixed language with an unbounded context-free subset, and testing equivalence for L-systems generating finite languages.

The following lemma is a well-known result.

Lemma 4.2. $\mathcal{L}(\mathbf{EDT0L})$, $\mathcal{L}(\mathbf{E0L})$ and $\mathcal{L}(\mathbf{ET0L})$ are efficiently closed under union, concatenation, homomorphisms, and intersection with regular sets.

Proof. Proofs can be seen in [5].

The following definition from [15] is necessary for results in this section.

Definition 7. A regular set $R_0 \subseteq \{0,1\}^*$ is unbounded if and only if there exist strings $r, s, x, y \in \{0,1\}^*$ such that $R_0 \supseteq \{r\} \cdot \{0x, 1y\}^* \cdot \{s\}$. □

By Lemma 4.2, the following results can be shown directly.

Theorem 4.6. For any fixed language $L_0 \in \mathcal{L}(\mathbf{EDT0L})$ (or $\mathcal{L}(\mathbf{E0L})$, $\mathcal{L}(\mathbf{ET0L})$) having an unbounded regular subset,
$\overline{\mathbf{K}} \leqslant_{ptime} \{< d > \mid d \in \mathbf{EDT0L}$ (or **E0L**, **ET0L**), $\mathcal{L}(d) = L_0\}$.
$\overline{\mathbf{K}} \leqslant_{ptime} \{< d > \mid d \in \mathbf{EDT0L}$ (or **E0L**, **ET0L**), $\mathcal{L}(d) \supseteq L_0\}$.

Proof. The proof is similar to that used in [16] to show that the predicate "$= L_0$" is undecidable for context-free grammars where L_0 is any fixed context-free language with unbounded regular subset. Since R_0 is unbounded, there exist $r, s, x, y \in \{0,1\}^*$ such that $\{r\} \cdot \{0x, 1y\}^* \cdot \{s\} \subseteq R_0$. $\forall G \in D$ where D is defined in Theorem 3.1, we can efficiently construct a N 1-rbd 1-CM H such that

$$\mathcal{L}(H) = \{r\} \cdot h(\mathcal{L}(G)) \cdot \{s\}$$

$$\cup$$

$$R_0 \cap \overline{\{r\} \cdot \{0x, 1y\}^* \cdot \{s\}}$$

where $h : \{0,1\}^* \mapsto \{0,1\}^*$ is the homomorphism defined by $h(0) = 0x$ and $h(1) = 1y$. Let N' be the set of H. Since R_0, x, y, s and r are fixed, we can determine G from H in polynomial time in $|H|$. $D \in \mathbf{P} \Rightarrow N' \in \mathbf{P}$. If $\mathcal{L}(G) = \{0,1\}^*$, then $\mathcal{L}(H) = R_0$; otherwise, $\mathcal{L}(G) = \{0,1\}^* - \{w\} \Rightarrow \mathcal{L}(H) = R_0 - \{rh(w)s\}$. □

Two direct corollaries from this proof are as follows:

Corollary 3. For any fixed linear context-free language L_0 with an unbounded regular subset, there exists a subset D of **EDT0L** (or **E0L**, **ET0L**) such that

1. $D \in \mathbf{P}$;
2. $\forall d \in D$, $\mathcal{L}(d)$ is a linear context-free language; and
3. $\overline{\mathbf{K}} \leqslant_{ptime} \{< d > \mid d \in D, \mathcal{L}(d) = L_0\}$.
4. $\overline{\mathbf{K}} \leqslant_{ptime} \{< d > \mid d \in D, \mathcal{L}(d) \supseteq L_0\}$.

Corollary 4. For any fixed unbounded regular set R_0, there exists a subset D of **EDT0L** (or **E0L**, **ET0L**) such that

1. $D \in \mathbf{P}$;
2. $\forall d \in D$, $\mathcal{L}(d)$ is regular; and
3. $\overline{\mathbf{K}} \leqslant_{ptime} \{< d > \mid d \in D, \mathcal{L}(d) = R_0\}$.
4. $\overline{\mathbf{K}} \leqslant_{ptime} \{< d > \mid d \in D, \mathcal{L}(d) \supseteq R_0\}$.

Moreover, [16] shows a method to study the complexity of the predicate "$\supseteq L_0$" for any L_0 with an unbounded context-free subset. The following lemma is necessary for this method.

Lemma 4.3. A context-free language is unbounded if and only if the set of its prefixes or the set of its suffixes contains an unbounded regular subset.

Proof. Proof can be found in [15]. □

Theorem 4.7. For any fixed language L_0 over $\{0,1\}$ having an unbounded context-free subset,
$\overline{\mathbf{K}} \leqslant_{ptime} \{< d > \mid d \in \mathbf{EDT0L}$ (or $\mathbf{E0L}$, $\mathbf{ET0L}$), $\mathcal{L}(d) \supseteq L_0\}$.

Proof. L_0 has an unbounded context-free subset. Hence, by Lemma 4.3 there exists a context-free language $L_1 \subseteq L_0$ where the set of L_1's prefixes or the set of L_1's suffixes contains an unbounded regular subset. $\Rightarrow$ There exist string $w, x, y \in \{0,1\}^*$ such that every string in $\{w\}\cdot\{0x, 1y\}^*$ is a prefix of some string in L_1 or every string in $\{0x, 1y\}^* \cdot \{w\}$ is a suffix of some string in L_1. This two cases are symmetrical, we only prove the first case here. For any EDT0L system $G \in \mathbf{EDT0L}$, we can efficiently construct an EDT0L system H such that

$$L(H) = \{w\} \cdot h(\mathcal{L}(G)) \cdot \{1y1y\} \cdot \{0,1\}^* \cup \overline{(\{w\} \cdot \{0x0x, 0x1y\}^*\{1y1y\} \cdot \{0,1\}^*)}$$

where $h : \{0,1\}^* \mapsto \{0,1\}^*$ is the homomorphism defined by $h(0) = 0x0x$ and $h(1) = 0x1y$. If $\mathcal{L}(G) = \{0,1\}^*$, clearly $\mathcal{L}(H) = \{0,1\}^*$. So $\mathcal{L}(H) \supseteq L_0$.
Otherwise, there exists a string $t \notin \mathcal{L}(G)$. $\Rightarrow wh(t)1y1y$ is not a prefix of $\mathcal{L}(H)$. So $\mathcal{L}(H) \not\supseteq L_1$. $\Rightarrow \mathcal{L}(H) \not\supseteq L_0$. □

In addition, we also consider equivalence problems for L-systems generating finite languages. We use $E0L_{fin}$, $EDT0L_{fin}$ and $ET0L_{fine}$ to denote the set of E0L systems generating finite languages, EDT0L systems generating finite languages, and ET0L systems generating finite languages, respectively. For any context-free grammar $G \in \mathbf{CFG}(\{\mathbf{0}, \mathbf{1}\})$, let $G = (\{0,1\}, V, P, s)$. If we add two productions $0 \to 0$ and $1 \to 1$ to P, certainly it does not change $\mathcal{L}(G)$. Then we can construct a finite substitution h such that for any production $A \to B$ in P, $B \in h(A)$. Clearly $H = (\{0,1\}, V, h, s)$ is an E0L system and $\mathcal{L}(H) = \mathcal{L}(G)$. This shows that E0L systems and ET0L systems can simulate context-free grammars in linear time. So Theorem 3.3 holds for $E0L_{fin}$ and $ET0L_{fin}$.

Theorem 4.8. There exists a constant $c > 0$ such that **Co-NEXPTIME** $\leqslant_{ptime} \{< d > \mid d \in E0L_{fin}$ (or $ET0L_{fin}$), $\mathcal{L}(d) = \{0,1,\lambda\}^{2^{cn}}$ where $n = |d|\}$.

5 Conclusion

Lindenmayer systems (L systems) are an important interdisciplinary research topic that has impacts on theoretical computer science, computer graphics, and developmental biology. In this paper, we study the complexity of many problems on L systems. These problems include equivalence and containment problems, and language class comparison problems. The undecidability or complexity lower bounds for some important open problems in the interdisciplinary research of L systems, such as the 0L-ness problems for regular languages, and context-free languages are established. By using highly efficient many-one reductions, we show that the 0L-ness problem for regular languages is PSPACE-hard, and for context-free languages is productive (a stronger form of undecidability). Also, testing equivalence and containment to some fixed languages for L systems are studied. The complexity of L systems only generating finite languages is also investigated. These results have significant practical meanings.

References

[1] A. Lindenmayer, Mathematical models for cellular interactions in development I. Filaments with one-sided inputs, *Journal of Theoretical Biology*, *18*(3):(1968), 280 – 299, ISSN 0022-5193, doi:http://dx.doi.org/10.1016/0022-5193(68)90079-9.

[2] G. Rozenberg, A. Salomaa, *Lindenmayer Systems: Impacts on Theoretical Computer Science, Computer Graphics, and Developmental Biology* (Springer-Verlag, Berlin, Heidelberg, 2001), ISBN 0387553207.

[3] L. Ciobanu, V. Diekert, M. Elder, Solution sets for equations over free groups are edt0l languages, in M. M. Halldórsson, K. Iwama, N. Kobayashi, B. Speckmann, editors, *Automata, Languages, and Programming*, pages 134–145 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2015), ISBN 978-3-662-47666-6.

[4] N. D. Jones, S. Skyum, Complexity of some problems concerning L systems, *Mathematical systems theory*, *13*:(1979), 29–43, doi:https://doi.org/10.1007/BF01744286.

[5] L. Kari, G. Rozenberg, A. Salomaa, *L Systems*, pages 253–328 (Springer Berlin Heidelberg, Berlin, Heidelberg, 1997), ISBN 978-3-642-59136-5, doi:10.1007/978-3-642-59136-5_5.

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

[7] H. Rogers, Jr., *Theory of Recursive Functions and Effective Computability* (MIT Press, Cambridge, MA, USA, 1987), ISBN 0-262-68052-1.

[8] R. I. Soare, *Recursively Enumerable Sets and Degrees* (Springer-Verlag New York, Inc., New York, NY, USA, 1987), ISBN 0-387-15299-7.

[9] J. Xie, H. B. Hunt, III, On the undecidability and descriptional complexity of synchronized regular expressions, *Acta Informatica*, *60*(3):(2023), 257–278, ISSN 0001-5903, doi:10.1007/s00236-023-00439-3.

[10] J. Xie, H. B. Hunt, III, On the computational and descriptional complexity of multi-pattern languages, *Available at SSRN*, doi:http://dx.doi.org/10.2139/ssrn.4493700.

[11] H. B. Hunt, D. J. Rosenkrantz, T. G. Szymanski, On the equivalence, containment, and covering problems for the regular and context-free languages, *Journal of Computer and System Sciences*, *12*(2):(1976), 222 – 268, ISSN 0022-0000, doi:http://dx.doi.org/10.1016/S0022-0000(76)80038-4.

[12] S. Greibach, A note on undecidable properties of formal languages, *Mathematical systems theory*, *2*(1):(1968), 1–6, ISSN 1433-0490, doi:10.1007/BF01691341.

[13] B. S. Baker, R. V. Book, Reversal-bounded multipushdown machines, *Journal of Computer and System Sciences*, *8*(3):(1974), 315 – 332, ISSN 0022-0000, doi:http://dx.doi.org/10.1016/S0022-0000(74)80027-9.

[14] G. Rozenberg, A. Salomaa, *The Mathematical Theory of L Systems*, pages 161–206 (Springer US, Boston, MA, 1976), ISBN 978-1-4615-8249-6, doi:10.1007/978-1-4615-8249-6_4.

[15] J. E. Hopcroft, J. D. Ullman, *Formal Languages and Their Relation to Automata* (Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1969).

[16] H. B. Hunt, III, D. J. Rosenkrantz, On equivalence and containment problems for formal languages, *J. ACM*, *24*(3):(1977), 387–396, ISSN 0004-5411, doi:10.1145/322017.322020.