

https://dspace.org/

Ogden's lemma for random permitting-and forbidding-context and ET0L languages

Rabkin, Max Stacey

2013-05-06
http://hdl.handle.net/10539/12711

Downloaded from a DSpace repository. https://dspace.org/

Ogden's Lemma for Random Permitting- and Forbidding-Context and ET0L Languages

Max Rabkin

A dissertation submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg, in fulfilment of the requirements for the degree of Master of Science

Johannesburg, October 2012

Declaration

I declare that this Dissertation is my own, unaided work. It is being submitted for the Degree of Master of Science at the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination at any other University.



11th day of October, 2012

Abstract

The pumping lemma for context-free languages is one of the best known and most useful theorems in the theory of formal languages. It has been generalised in two directions: firstly, to larger classes of languages – for example, to random permitting- and forbidding-context (rPc and rFc) languages by Ewert and Van der Walt; and secondly, to a stronger version for context-free languages by Ogden.

This dissertation aims to examine these theorems and language classes in order to reunify these two lines of research by proving analogues of Ogden's Lemma for Extended Table-driven 0-context Lindenmayer-system (ET0L) languages, rPc languages and rFc languages, and thereby to provide new tools for classifying which languages belong to these classes.

Acknowledgements

I would like to thank my supervisor, Sigrid Ewert, who suggested the topic and guided the work of this dissertation.

Some of the material on ETOL languages appeared as [Rabkin 2012] in the proceedings of the Sixth International Conference on Language and Automata Theory and Applications (LATA 2012).

This work is based upon research supported by the National Research Foundation of South Africa.

Contents

1 Introduction 1
- 1.1 Context-Free Languages . . . 1
- 1.2 Random Context . . . 2
- 1.3 Roadmap . . . 4

2 Background 5
- 2.1 Definitions . . . 5
- 2.2 Review of Literature on Ogden's Lemma . . . 6
- 2.3 Review of Literature on Restricted Rewriting . . . 8

3 Random Context 11
- 3.1 Relationships to Other Classes . . . 13
- 3.2 Previous Necessary Conditions . . . 14

4 Ogden's Lemma for RPC and RFC Languages 17
- 4.1 Permitting Context . . . 19
- 4.2 Forbidding Context . . . 22

5 ET0L Systems 24
- 5.1 Relationship to Random Context . . . 25
- 5.2 A Previous Iteration Theorem . . . 26

6 Ogden's Lemma for ET0L Languages 28
- 6.1 Main Results . . . 29
- 6.2 Applications . . . 33

7 Conclusion and Future Work 36

List of Figures

2.1 Hewett and Slutzki's results 7
2.2 Kløve's results 8

3.1 Diagram of containment relationships 14

6.1 Proof of Theorem 6.3 30
6.2 An example ET0L derivation 32

Chapter 1

Introduction

In formal language theory, some of the most popular formalisms for generating languages are string-rewriting systems. In such systems, one starts with a symbol, and step-by-step replaces a symbol or substring by another string according to the rules of the system (which is called a grammar) to obtain a new string, which can either be rewritten again or output.

The grammars vary in the allowed complexity of the rewriting rules. A major task in formal language theory is to understand the generative power of different classes of these grammars.

1.1 Context-Free Languages

Perhaps the most important of these classes is the class of *context-free grammars* (cfgs), where each rewriting rule consists of replacing a single symbol with a string of symbols. These grammars have been studied in depth, and have seen extensive use in programming languages.

There are various theorems which give properties shared by all context-free languages (cfls) and thus an understanding of the power and limitations of context-free grammars. Among the most useful are Parikh's Theorem [Parikh 1966, Theorem 2] and the iteration theorems – the pumping lemma of [Bar-Hillel *et al.* 1961] and Ogden's lemma [Ogden 1968, Theorem 1]. The pumping lemma essentially states that any sufficiently long string in a context-free language can be turned into a longer (or shorter) string in the same language by repeating (or deleting) two short segments of the original string.

Ogden's lemma is similar, but is more targeted in that the selection of repeatable segments can be partially controlled: if enough positions in the string are chosen as *marked*, then at least one of the repeatable segments will contain a marked position. The generalized version of Ogden's lemma developed by Bader and Moura [1982]

extends this control by allowing a small number (relative to the number of marked positions) of positions to be *excluded* from the repeatable segments.

Theorem 1.1 (Ogden's Lemma). *If L is a context-free language, then there exists an integer m such that for any $u \in L$ with at least m marked positions, u can be written as $u = vwxyz$ such that*

1. *x and at least one of w or y both contain a distinguished position;*
2. *wxy contains at most m distinguished positions;*
3. *$vw^t xy^t z \in L$ for all $t \in \{0, 1, 2, \dots\}$.*

These are called iteration theorems because they generate a sequence of strings in a language by iterating some repetition operation.

In contrast to these, Parikh's Theorem does not operate on particular strings but gives information about languages as wholes; furthermore, it ignores the order of symbols in strings and considers only their number. Specifically, it states that the sets of symbol-count vectors (the *commutative* or *Parikh image*) of context-free languages are exactly those that can be generated by regular languages.

This dissertation focuses on generalising the iteration theorems to larger classes of languages.

1.2 Random Context

Context-free languages are well understood, but have limited descriptive power. In contrast, fully context-sensitive languages are overly powerful and therefore resistant to analysis – in fact, they can be seen as non-deterministic machines with linearly-bounded memory, and thus many questions about them are essentially questions about computational complexity rather than languages.

In light of this situation, several variations of context-free grammars have been introduced which use *regulated rewriting*: these use the same rules as context-free grammars but place restrictions on which rules can be applied to a given string, or what order they can be applied in. Each of these grammar classes generates more than just the context-free but less than the context-sensitive languages, while remaining amenable to analysis. Dassow and Păun [1989] give an overview of these grammars, which are also known as *grammars with controlled derivations*.

One form of regulated rewriting is seen in *random-context grammars* (rcgs), introduced by van der Walt [1972], where rules can be applied only to strings containing certain symbols (the permitting set) and not containing others (the forbidding set), but the position and number of these symbols is ignored. In particular, one can consider random *permitting-* and *forbidding*-context grammars (rPcgs and

rFcgs) where one or the other of these sets is empty. The languages generated by these classes are called the random-context, random permitting-context and random forbidding-context languages (rcls, rPcls and rFcls) respectively. The word "random" in this case does not refer to probability, but the fact that the context is global rather than local.

The random-context restrictions on rewriting allow the development of strings in the language to be globally controlled to an extent. For example, the copy language, $\{xx : x \in \Sigma^*\}$, whose words are two copies of the same string, is not context-free (this can be shown with the pumping lemma) but is an rPcl and an rFcl (see Example 3.6 in Chapter 3); the language of strings whose lengths are the powers of two, $\{a^{2^n} : n \in \mathbb{N}\}$ is not context-free (this can be shown with either the pumping lemma or Parikh's Theorem) but is an rFcl [Dassow and Păun 1989, Example 1.1.7].

Despite this increased power, derivations in rPcgs and rFcgs are still sufficiently structured to obtain iteration theorems: a pumping lemma for rPcls and a shrinking lemma for rFcls are both known. The difference between pumping and shrinking comes from the fact that in an rPcg more context (i.e., more instances of non-terminals) gives more freedom (i.e., the set of applicable productions increases) since non-terminals *permit* productions, while in rFcgs, the opposite is true.

These systems could be considered as a model of computation rather than a type of grammar, and in the light of the relationship of string rewriting to the theory of computation (with context-sensitive grammars modelling space-bounded non-deterministic computation, and type-0 grammars modelling general computation) this perspective is not surprising. On the other hand, random-context languages and the two subclasses do not seem to correspond to any natural complexity class, and one can certainly obtain interesting language-theoretic results about them, as is done here.

One interesting subclass of the rFcls are the *ET0L* (Extended Table-driven 0-context Lindenmayer) languages. Lindenmayer systems are widely used, for example in modelling plants for computer graphics (see, for instance, Lindenmayer and Prusinkiewicz [1990]). In an ET0L system, productions are applied in parallel to all the symbols in a string, and the choice of productions is table-driven. This means that at each step, a table of productions is chosen from a fixed set, and the production applied to each symbol comes from this table.

Another subclass of the rcls is that of the *finite-index* random-context languages. These are generated by the random-context grammars where there is a number (called the *index*) such that any word has a derivation in which the number of non-terminals is bounded by the index. The classes of rPcls, rFcls, rcls and ET0L languages all coincide under this restriction.

1.3 Roadmap

In Chapter 2 we will examine the literature on iteration theorems and their application to random-context and ETOL languages. Chapter 3 will give more detailed background on random context and its subclasses; this is followed by proving versions of Ogden's Lemma for rPc and rFc languages in Chapter 4. A background on ETOL languages is given in Chapter 5. Then, in Chapter 6, a version of Ogden's Lemma is proved for these languages and, as a corollary, a proof is given for the theorem of Ehrenfeucht and Rozenberg [1976] on rare and non-frequent symbols in ETOL languages.

The original contribution of this research consists of the versions of Ogden's Lemma for the two subclasses of random context languages, presented in Chapter 4, and for ETOL languages, in Chapter 6, as well as corollaries of these theorems, and the concept of dense markings and related lemmas which were used to prove them.

Chapter 2

Background

In this chapter we review the literature on Ogden's Lemma and on iteration theorems for random-context and ETOL languages. We also give some basic definitions which are not specific to any particular class of languages. More specific definitions are given in the appropriate chapters.

2.1 Definitions

We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$, $\mathbb{N}$ for $\{0, 1, 2, \ldots\}$, and $\mathbb{N}^+$ for $\{1, 2, \ldots\}$.

We denote the set of *words* (or *strings*) over an alphabet (finite set) Σ by Σ^*. The empty word is denoted λ. The length of a word w is denoted $|w|$.

If $x = uvw$, then v is called a *subword* of x or symbolically, $v \sqsubseteq x$. If $v \sqsubseteq x$ and $v \neq x$ then we say that v is a strict subword of x and write $v \sqsubsetneq x$.

If Σ and Σ' are alphabets, then a function $f : \Sigma^* \to \Sigma'^*$ is called a *homomorphism* if $f(uv) = f(u)(v)$ for all u and v.

We let $\#_b(w)$ denote the number of appearances of the symbol b in the word w, and let $\#_B(w) = \sum_{b \in B} \#_b(w)$ when B is a set of symbols.

In order to extend Ogden's Lemma, several concepts related to marking will be needed. A *word with marked symbols* can be defined formally as a pair (w, M), where $M \subseteq [|w|]$: an instance of a symbol in w is called *marked* if its position is in M. Operations on words extend in the obvious way to marked words. For example, the concatenation of marked words (w, M) and (w', M') is obtained as the concatenation of the underlying words, with the marks from M' shifted by $|w|$; i.e., $(w, M)(w', M') = (ww', M \cup (M' + |w|))$. However, we will not use this notation, preferring for simplicity to identify a marked word with its underlying unmarked word.

When the meaning is clear from context, we will not always disambiguate between a symbol and an instance of a symbol in a word.

2.2 Review of Literature on Ogden's Lemma

Ogden [1968] originally developed the lemma which now bears his name by generalizing the pumping lemma for context-free languages of Bar-Hillel *et al.* [1961]. This form of the pumping lemma expresses the idea that essentially any part of a string can be pumped in a context-free language (the exceptions being bounded in size), by allowing a certain number of symbols to be chosen as marked; the pumping operation is then guaranteed to affect some of the marked symbols. Ogden's main aim in proving this theorem was to use it to prove that certain context-free languages are inherently ambiguous (a language is inherently ambiguous if every grammar for the language derives two different derivation trees for some word). However, he also noted its usefulness as a necessary condition for context-freeness. It is this aspect which the present work seeks to extend to larger classes of languages.

Ogden's Lemma was strengthened by Bader and Moura [1982] to allow a small number of excluded positions in addition to the marked positions allowed by Ogden's Lemma. More precisely, the number of marked positions required grows exponentially with the number of excluded positions. While Ogden's Lemma guarantees that a marked position will be pumped, the theorem of Bader and Moura also guarantees that no excluded position will be. This result demonstrates the value of considering alternative forms of marking to that used by Ogden – in this case, the change is a generalisation and strengthens the theorem, but a restricted form of marking is used in Chapter 4 to make it possible to prove theorems about random permitting- and forbidding-context. One might hope to obtain a necessary and sufficient condition for context-freeness by choosing an appropriate relation between marked and excluded positions, but Bader and Moura discuss a family of further generalized conditions, where the exponential function is replaced by any other function, and prove that each of these conditions is satisfied by a non-context-free language, so none of them are sufficient conditions for a language to be context-free.

Hewett and Slutzki [1988] give an overview of the classes of languages which satisfy various iteration theorems for context-free languages (since these are necessary but not sufficient conditions, the classes strictly include the context-free). The conditions include the pumping lemma, Ogden's Lemma, Bader and Moura's Lemma, Sokolowski's Condition (which is a generalisation of the fact that the copy language is not context-free: it says that if a language contains $uxvxw$ for all words x over some subalphabet, then it also contains $uxvyw$ for some $x \neq y$) and Grant's Condition (an extension of Sokolowski's Condition). They find all containment relations between these conditions by constructing examples. For instance, they prove that Sokolowski's Condition is implied by Bader and Moura's Lemma, but is incomparable with Ogden's Lemma. The results are summarised in Figure 2.1 (adapted from Figure 4 in [Hewett and Slutzki 1988]).

Ramos-Jiménez *et al.* [1998] extended this overview by comparing Parikh's The-

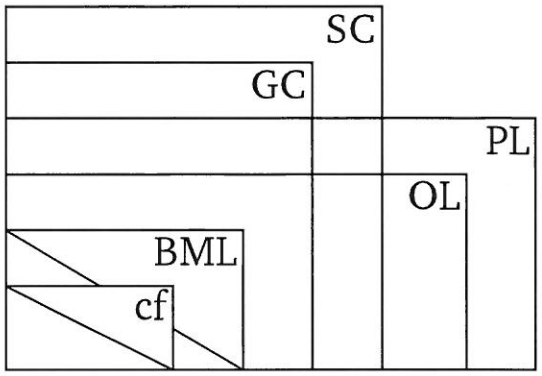


Figure 2.1: The relationships between the class of context-free languages (cf) and the languages satisfying the pumping lemma (PL), Ogden's Lemma (OL), Bader and Moura's Lemma (BML), Sokolowski's Condition (SC) and Grant's Condition (GC). Hewett and Slutzki showed that each region is non-empty.

orem to the theorems discussed by Hewett and Slutzki. They show that the class of languages satisfying Parikh's Theorem is incomparable with all of the above-mentioned conditions.

Another paper which examines the classes of languages satisfying iteration theorems is that of Kløve [1977]. Kløve investigates pumping conditions of the following forms:

Definition 2.1. Let $r \in \mathbb{N}^+$ and $d \in \{0,1\}$. Then $L \in P_d(r)$ if there is an integer m such that for any $u \in L$ with $|u| \geq m$, u can be written as $u = v_1 w_1 v_2 w_2 \dots v_r w_r v_{r+1}$ such that

- $w_1 w_2 \dots w_r \neq \lambda$;
- $v_1 w_1^t v_2 w_2^t \dots v_r w_r^t v_{r+1} \in L$ for all $t \geq d$.

Similarly, if $d \in \{0,1\}$, then $L \in P_d(\infty)$ if there is an integer m such that for any $u \in L$ with $|u| \geq m$, there exists an $r \in \mathbb{N}^+$ such that u can be written as $u = v_1 w_1 v_2 w_2 \dots v_r w_r v_{r+1}$ and

- $w_1 w_2 \dots w_r \neq \lambda$;
- $v_1 w_1^t v_2 w_2^t \dots v_r w_r^t v_{r+1} \in L$ for all $t \geq d$.

For example, the usual pumping lemma for regular languages is similar to the condition $P_0(1)$, and the pumping lemma for context-free languages to $P_0(2)$ (the difference is that Kløve's conditions do not place any upper bound on the lengths of the subwords). Kløve investigates the relationships between these classes, and with well-known classes of languages. These results are summarized in Figure 2.2

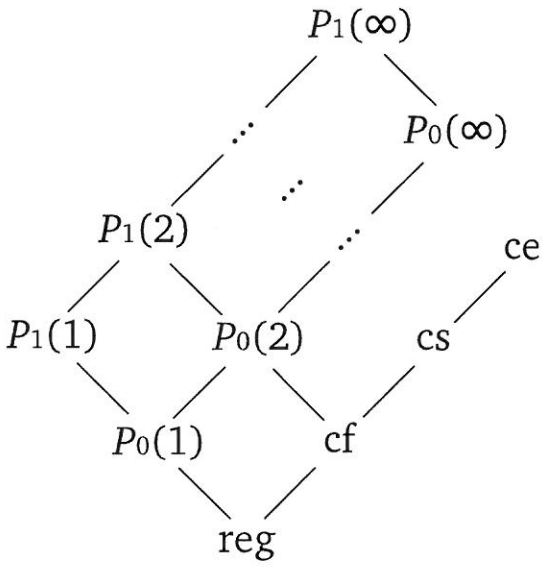


Figure 2.2: The relationships between the classes of regular (reg), context-free (cf), context-sensitive (cs), and computably enumerable (ce) languages and Kløve's pumping conditions. Containment is represented by edges, with the larger classes above the smaller.

(adapted from Figure 1 in [Kløve 1977]). He also investigates the closure properties of the classes: all are closed under union, concatenation, reversal, λ-free homomorphisms and Kleene's plus operation; none are closed under intersection, intersection with regular languages, complementation or inverse homomorphisms. The $P_0(\cdot)$ classes are closed under general homomorphisms, while the $P_d(\infty)$ classes are closed under permutation.

2.3 Review of Literature on Restricted Rewriting

The iteration theorems have been generalised for some classes of grammars with restricted rewriting. Ehrenfeucht and Rozenberg [1974] found a pumping lemma for deterministic ETOL (EDTOL) languages; however, only certain words can be pumped according to this theorem, and not all EDTOL languages have such words. It also differs from the pumping lemma for cfls in that more than two subwords may be repeated. This is similar to Kløve's condition $P_1(\infty)$ (the difference being that not all words can be pumped).

In proving that the family of finite-index matrix languages is incomparable with the family of context-free languages, Beauquier [1979, Lemma 2] gave a version of Ogden's Lemma for such languages. Like the pumping lemmas of Ehrenfeucht

and Rozenberg [1974] and Kløve [1977], this lemma may repeat more than two subwords (at most, the number is equal to the index of the language). However, this result is erroneous, as the index-2 matrix language $\{a^n b^n c^n d^n : n \in \mathbb{N}^+\}$ shows (the lemma effectively claims that index-2 matrix languages satisfy Ogden's Lemma for context-free languages, but this language clearly does not). It is not immediately clear how to correct the error.

The classes of random context, random permitting-context, random forbidding-context, ETOL and matrix languages all coincide under the finite-index restriction, as shown by Păun [1979b] and Rozenberg [1976]. Thus all the classes with which we are concerned can be considered generalisations of this class.

Ewert and van der Walt [2002] proved a pumping lemma for random permitting-context languages. This differs from the previously mentioned theorems in that the pumping operation can move subwords as well as repeating them. They also proved a shrinking theorem for random forbidding-context languages [van der Walt and Ewert 2000], which is similar but instead of making a larger word, it deletes subwords to make a smaller word.

Van der Walt and Ewert [2003] also give a necessary condition for random context which is not of the iteration type, but this applies only to picture languages. It is a commutation result: i.e., it states that in large enough pictures of a random-context picture language, there is a pair of (large) subpictures that can be swapped without leaving the language. The distinction between picture languages and string languages arises because the structure of the derivation tree of a word is almost completely hidden in a word (only the ordering of the leaves is preserved), while the positions and sizes of squares in a picture language retain some information about the shape of the derivation tree. While this result does not apply directly to the string languages which are the topic of this work, the proof sheds some light on the structure of random-context derivations, which adds to the understanding of random context, at least intuitively.

Dassow and Păun [1989] describe in depth the state of the art in regulated rewriting as it was in the late 1980's, including many results on the relationships between different types of regulated rewriting. They give matrix, programmed and random-context grammars central place, and describe how other types of grammars such as L systems fit into this framework. They also give many results of general interest about these systems, including a pumping lemma for finite-index random-context grammars [Dassow and Păun 1989, Lemma 3.1.6] (included in the present work as Theorem 3.8). This theorem only states the existence of a pumpable word in every infinite language in the class, in contrast to other results which say that any long enough word can be pumped.

We have now seen some of the previous work in the areas of iteration theorems and random-context and ETOL languages. In the next chapter, we will examine

random context in more detail, including more detailed descriptions of some of the results mentioned above. This will prepare the ground to prove the desired results: Ogden's Lemma for rPcls and rFcls.

Chapter 3

Random Context

Random-context grammars extend context-free grammars with the ability to sense global context, unlike the local context-dependence of context-sensitive grammars. Each production has two context sets of non-terminal symbols which respectively allow or block the production from being applied, depending on whether these symbols appear in the string or not. The positions of the symbols in the context are irrelevant – they can be at arbitrary or "random" positions. This is the intention of the name *random context*: there is no probability in the definition.

Definition 3.1. A *random-context grammar* is a tuple $G = (V, \Sigma, S, P)$, where V and Σ are disjoint finite sets (the alphabets of non-terminals and terminals, respectively), $S \in V$, and $P \subseteq V \times (V \cup \Sigma)^+ \times 2^V \times 2^V$ is a finite set of rules or productions. An element $(A, x, \mathcal{P}, \mathcal{F})$ of P is written $A \to x\ (\mathcal{P}; \mathcal{F})$. $\mathcal{P}$ and $\mathcal{F}$ are respectively called the permitting and forbidding sets of the production, while A and x are respectively called the left-hand side and right-hand side.

If $\mathcal{F} = \emptyset$ for every production in P, then G is called a *random permitting-context* grammar; if $\mathcal{P} = \emptyset$ for every production in P, then G is called a *random forbidding-context* grammar.

When writing a rule $A \to x\ (\mathcal{P}; \mathcal{F})$ in a grammar, either set $\mathcal{P}$ or $\mathcal{F}$ may be omitted if it is empty, and if $\mathcal{P} = \mathcal{F} = \emptyset$ then we may simply write $A \to x$.

Definition 3.2. If $G = (V, \Sigma, S, P)$ is a random-context grammar, then we define $\Rightarrow_G$ such that $x \Rightarrow_G y$ if $x = x_1 A x_2$ and $y = x_1 w x_2$, where $A \to w\ (\mathcal{P}; \mathcal{F})$ is in P, no non-terminal in $\mathcal{F}$ appears in $x_1 x_2$, and every non-terminal in $\mathcal{P}$ appears in $x_1 x_2$.

If the grammar G is clear from context, we will simply write $\Rightarrow$.

Definition 3.3. If $G = (V, \Sigma, S, P)$ is a random-context grammar, the *language generated by* G is

$$L(G) = \left\{ x \in \Sigma^* : S \Rightarrow_G^* x \right\},$$

where $\Rightarrow_G^*$ is the transitive and reflexive closure of $\Rightarrow_G$.

A language is called random-context, random permitting-context, or random forbidding-context if it is generated by a grammar of the same type.

Definition 3.4. If $G = (V, \Sigma, S, P)$ is a random-context grammar, and $x \in (V \cup \Sigma)^*$, then we say x is a *sentential form of* G if $S \Rightarrow^* x$.

One measure of the syntactic complexity of languages is the number of non-terminals needed at any point in a derivation of that language. This is called the index. The languages with finite index form an interesting subclass of the random-context languages.

Definition 3.5. Let G be a random-context grammar.

A derivation $S = u_1 \Rightarrow_G u_2 \Rightarrow_G \ldots \Rightarrow_G u_n$ is said to have *index* k if k is the maximum number of non-terminal symbols in any u_i, for $1 \leq i \leq n$.

A word $u \in L(G)$ is said to have *index* k, if k is the minimum index of any such derivation of u.

G is called *index* k if k is the supremum of the indices of words in $L(G)$ (i.e., the maximum if it exists, and ∞ otherwise).

A language L is an *index-k* as a language of a certain type (random-context, ETOL, etc.) if k is the minimum index of any such grammar generating it.

A grammar or language is said to have *finite index* if it has index k for some $k \in \mathbb{N}$, and *infinite index* otherwise.

The following example shows some of the power of random context.

Example 3.6. The copy language over a two-symbol alphabet, $\{xx : x \in \{a, b\}\}$, is an rPcl. It is generated by the rPc grammar $(\{S, L, L_a, L_b, L'_a, L'_b, R, R'\}, \{a, b\}, S, P\}$, where P contains the following productions:

$$\left.\begin{array}{ll} L \to L_s & (\{R\};) \\ L \to L'_s & (\{R\};) \\ L_s \to sL & (\{R'\};) \\ L'_s \to s & \\ R \to sR' & (\{L_s\};) \\ R \to s & (\{L'_s\};) \end{array}\right\} \text{for } s \in \{a, b\}$$

$$S \to LR$$

$$R' \to R \quad (\{L\};)$$

This grammar can be generalised to larger alphabets in the obvious way. Note that in each sentential form of this grammar, at most two non-terminals appear,

and thus this language has index (at most) two. Since the finite-index random forbidding- and permitting-context grammars generate the same languages, there must also be a forbidding-context grammar for this language.

3.1 Relationships to Other Classes

The random permitting-context and forbidding-context grammars are both stronger than context-free grammars: the inclusion is obvious, since the definitions of these grammars include the context-free grammars, simply by setting both context sets to $\emptyset$; the strictness can be seen by the examples given in Section 1.2. On the other hand, there are random-context languages which are neither random permitting- nor forbidding-context languages (the language of van der Walt and Ewert [2000, Lemma 4] is an example). Context-sensitive grammars are strictly stronger than random-context grammars, but curiously, no explicit example which demonstrates this is known [Atcheson *et al.* 2006].

We will prove the inclusion here, but not the strictness (the proof of strictness is quite involved, and will not be needed for the main results).

Theorem 3.7. *Given a random-context grammar G, there is a non-deterministic algorithm which decides the membership problem for $L(G)$ in linear space. Thus $L(G)$ is a context-sensitive and computable language.*

Proof. Since the class of context-sensitive languages is exactly the class of languages decidable in non-deterministic linear space, it will suffice to give the algorithm.

To decide if a word w is in the language, we simulate all possible derivations of G whose sentential forms are no longer than $|w|$:

1. Set x to S
2. Non-deterministically pick a rule r from G
3. Non-deterministically pick a position in x where r can be applied, and assign the result of this application to x
4. If $x = w$, accept
5. If $|x| > |w|$, reject
6. Go to step 2

The amount of storage used by this algorithm exceeds $|w|$ by a constant.

Because random-context productions cannot be erasing (i.e., cannot have λ as a right-hand side), if $S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n = w$, then $|w_i| \leq |w|$ for all $i \in [n]$. Thus if there is a derivation of w in G, this algorithm will find it. □

Note that by Savitch's Theorem (see, for example, [Sipser 2006, Theorem 8.5]), non-deterministic linear space languages can be decided deterministically with at most quadratic space.

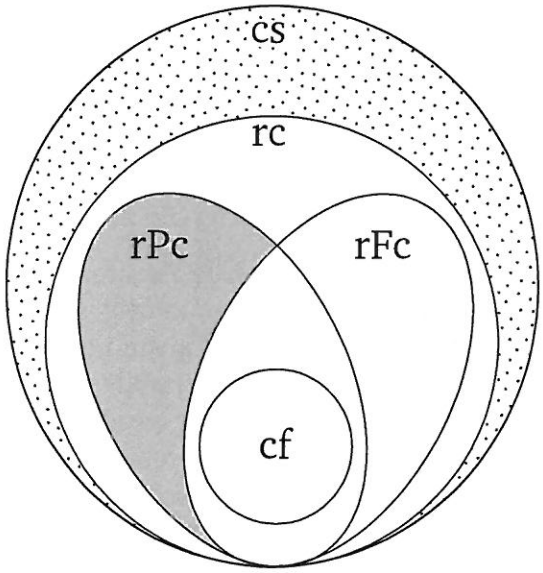


Figure 3.1: A diagram showing the containment relationships between the classes of context-sensitive (cs), random-context (rc), random permitting-context (rPc), random forbidding-context (rFc) and context-free (cf) languages. It is not known whether the shaded region contains any languages; the speckled region is not empty, but no example is known. All other regions are nonempty.

If one allows the empty word λ to appear on the right-hand side of a rule in the definition of a random-context grammar, then such a grammar can generate exactly the computably enumerable languages [Dassow and Păun 1989, Theorem 1.2.5]. The same is true of the context-sensitive grammars. However, it was recently shown by Zetzsche [2010] that if one does the same for permitting grammars, the generative power is unaffected (except that the empty word itself may appear in the language).

3.2 Previous Necessary Conditions

Dassow and Păun [1989, Lemma 3.1.6] proved the following pumping lemma for finite-index matrix grammars, but these generate the same languages as finite-index random-context grammars, the permitting and forbidding subclasses of these, and the finite-index ETOL systems [Păun 1979b; Rozenberg 1976]. The theorem is *exis-*

tential in the sense that it only gives the existence of a pumpable word, rather than stating that any long enough word is pumpable.

Theorem 3.8. *If L is an infinite index-k matrix language, then there exists a word $z \in L$ and a number $n \leq k$ such that*

1. *z can be written as $z = u_1 v_1 w_1 x_1 u_2 v_2 w_2 x_2 \cdots u_n v_n w_n x_n u_{n+1}$;*
2. *$|v_1 x_1 v_2 x_2 \cdots v_n x_n| > 0$;*
3. *$u_1 v_1^t w_1 x_1^t u_2 v_2^t w_2 x_2^t \cdots u_n v_n^t w_n x_n^t u_{n+1} \in L$ for all $t \in \mathbb{N}^+$.*

For the same class of languages, Parikh's Theorem also holds. According to Păun [1979a], this was proved by Salomaa [1969]; since I have been unable to obtain that paper, I have developed the proof given here independently. The proof is relatively direct (and easier than for context-free languages). It involves building an automaton whose states are the finitely many words of non-terminals with length at most equal to the index. The automaton is a non-deterministic finite automaton where the transitions are labelled by general words – not necessarily single symbols. Such automata are easily seen to be equivalent to the usual kind.

Theorem 3.9 (Parikh's Theorem for Finite-Index RCLs). *If L is a random-context language of finite index, then there exists a regular language L' which is letter equivalent to L (i.e., every word in L is an anagram of a word in L' and vice versa).*

Proof. Suppose $G = (V, \Sigma, P, S)$ generates L with index k. Let $g : (V \cup \Sigma)^* \to V^*$ be the homomorphism which maps elements of V to themselves and elements of Σ to λ, and let $h : (V \cup \Sigma)^* \to \Sigma^*$ be the similar homomorphism which maps elements of Σ to themselves and elements of V to λ.

Construct an automaton M with states labelled by words over V with length at most k ($V^{\leq k}$). Whenever $v \Rightarrow u$ with v and $g(u) \in V^{\leq k}$, add a transition from v to $g(u)$ labelled by $h(u)$. The initial state is S and the terminal state is λ. Since $V^{\leq k}$ is finite, the language generated by M is regular.

Since the positions of terminals do not affect the applicability of productions in an rcg, the paths in M correspond to derivations in G, ignoring the order of terminals: the states on the path correspond to the non-terminals in the sentential forms of the derivations, and the transition labels to the terminals produced. □

The next results are the pumping lemma for rPcls from Ewert and van der Walt [2002] and the shrinking lemma for rFcls from van der Walt and Ewert [2000]. It is of these theorems that we prove Ogden-like generalisations in Chapter 4. Their proofs can be obtained easily by specialising Theorem 4.12 and 4.16, so we will not include them here.

Theorem 3.10 (Pumping Lemma for RPCLs). *If L is an rPcl, then there exists an integer m (the pumping threshold) such that for any $w \in L$ with $|w| \geq m$, there is a number $l \in [m]$ such that*

1. *w contains l mutually disjoint non-empty subwords $u_1, u_2, \ldots, u_l$ and l mutually disjoint non-empty subwords $v_1, v_2, \ldots, v_l$ such that for each $i \in [l]$ there exists a $j \in [l]$ such that $v_i \sqsubseteq u_j$, and $v_i \sqsubsetneq u_j$ for at least one pair $(i, j) \in [l]^2$;*

2. *the word obtained from w by replacing v_i with u_i for all $i \in [l]$ is in L; and,*

3. *iterating this operation arbitrarily many times yields a word in L.*

Roughly, this says that whenever we have a long enough word in an rPcl, we can find some big subwords $(u_1, u_2, \ldots, u_l)$ and some smaller subwords $(v_1, v_2, \ldots, v_l)$, such that if we replace the smaller subwords by the larger, and repeat this any number of times, the resulting word is still in the language.

As noted by Ewert and van der Walt [2002], this immediately implies that the set of word lengths of any infinite rPcl contains an arithmetic progression. Furthermore, all but finitely many lengths belong to an arithmetic progression contained in the length set. For example, $\{a^{2^n} : n \in \mathbb{N}\} \cup \{a^{3n} : n \in \mathbb{N}^+\}$ contains the infinite arithmetic progression $3, 6, 9, \ldots$ in its length set, but the length set nevertheless does not satisfy the pumping lemma.

Theorem 3.11 (Shrinking Lemma for RFCLs). *For any rFcl L and $t \in \mathbb{N}^+$, there is an $m \in \mathbb{N}^+$ (the shrinking threshold) such that for any word $w \in L$ with $|w| \geq m$ there are t words $w^{(1)}, w^{(2)}, \ldots, w^{(t)} = w$ such that for every $j \in \{2, 3, \ldots, t\}$:*

1. *there exists a number $l \in [m]$;*

2. *$w^{(j)}$ contains l mutually disjoint non-empty subwords $u_1, u_2, \ldots, u_l$ and l mutually disjoint non-empty subwords $v_1, v_2, \ldots, v_l$, such that for each $i \in [l]$, there exists a $k \in [l]$ such that $v_i \sqsubseteq u_k$, and for at least one $(i, k) \in [l]^2$, $v_i \sqsubsetneq u_k$;*

3. *if each u_i is replaced with v_i, then the resulting word is $w^{(j-1)}$.*

Roughly, if we want to shrink a sufficiently large word of an rFcl $t - 1$ times, we can find some big subwords $(u_1, u_2, \ldots, u_l)$ and some small subwords $(v_1, v_2, \ldots, v_l)$, such that if we replace the big subwords by the smaller, the resulting word is still in the language, and this can be done a total of $t - 1$ times.

Now that we have some understanding of random context, and particularly the pumping lemma for rPcls and shrinking lemma for rFcls, we can extend these two results to marking variants in the next chapter.

Chapter 4

Ogden's Lemma for RPC and RFC Languages

In this chapter we will prove versions of Ogden's Lemma for rPcls and rFcls. It is not immediately clear how to add marking to the pumping and shrinking lemmas. It is not sufficient to require, for example, that one of the words $u_1, u_2, \ldots, u_l$ and one of the words $v_1, v_2, \ldots, v_l$ each contain a marked position: the theorems do not guarantee that all of these words are meaningfully affected (some could be replaced by themselves), while Ogden's original lemma really does treat the marked symbols specially. Perhaps the most natural requirement (which is used here) is that the pumping operation increase the number of marked symbols, and conversely the shrinking operation decrease the number.

The theorems in this chapter apply only to so-called *densely-marked* words, in which there are no large unmarked subwords (note that this restricts only the markings and not the underlying words). This property is hereditary in the sense that if $u \Rightarrow^* v$ and v is densely marked, then so is u. This allows one to prove that the introduction of marked symbols in a derivation cannot be postponed indefinitely, giving a bound on the size of sentential forms as a function of the number of marked symbols. This turns out to be important in finding a repeated context in a derivation, which in turn allows the derivation to be extended (in the case of rPcgs) or shortened (in rFcgs). The concept of dense markings and the related lemmas appear to be original.

First we will need some definitions.

Definition 4.1. A marked word w is *k-densely marked* if every subword u of w with length at least k contains at least one marked symbol.

Definition 4.2. For an alphabet Σ, if $w \in \Sigma^*$ and $a \in \Sigma$, then $\#_a^{\mathrm{m}}(w)$ is the number of marked appearances of a in w.

Definition 4.3. If $G = (V, \Sigma, P, S)$ is an rcg and $V = \{A_1, A_2, \ldots, A_n\}$, then $\mathrm{cnt}^{\mathrm{m}} : (V \cup \Sigma)^* \to \mathbb{N}^{2n}$, $\mathrm{cnt}^{\mathrm{m}}(w) = (\#^{\mathrm{m}}_{A_1}(w), \#_{A_1}(w) - \#^{\mathrm{m}}_{A_1}(w), \#^{\mathrm{m}}_{A_2}(w), \#_{A_2}(w) - \#^{\mathrm{m}}_{A_2}(w), \ldots, \#^{\mathrm{m}}_{A_n}(w), \#_{A_n}(w) - \#^{\mathrm{m}}_{A_n}(w))$; i.e. the vector of counts of marked and unmarked non-terminals.

The name cnt stands for "count of non-terminals". Since the sum of the entries in $\mathrm{cnt}^{\mathrm{m}} w$ is $|w|$, it is natural to make the following definition.

Definition 4.4. If v is a vector, we write $|v|$ for the sum of the entries in v.

Definition 4.5. If $G = (V, \Sigma, P, S)$ is an rcg and $u \in (V \cup \Sigma)^*$, $w \in \Sigma^*$, with $u \Rightarrow^* w$, then the *contribution* of a symbol in u is the subword of w formed by the terminals which descend from that symbol.

We will consider a non-terminal to be marked if there is a marked symbol in its contribution.

Definition 4.6. A marked instance of a non-terminal in a derivation is called *branch* if it is rewritten to a word containing more than one marked symbol.

The concept of branch symbols is due to Ogden [1968], who called the corresponding nodes of a derivation tree *B-nodes*.

The following lemma formally states the heredity property of densely-marked words mentioned above.

Lemma 4.7. *Let $G = (V, \Sigma, S, P)$ be an rcg. If $S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n \in (V \cup \Sigma)^*$ and w_n is k-densely marked, then w_i is k-densely marked for all $i \in [n]$.*

Proof. Suppose v is a subword of w_i with length at least k which contains no marked symbols. If it appears in w_{i+1} as-is, then w_{i+1} is not k-densely marked. Otherwise, some (unmarked) symbol in v was rewritten; since an unmarked non-terminal can only derive unmarked symbols (by the definition of a marked non-terminal), w_{i+1} contains an unmarked subword of length at least $|v|$, so is not k-densely marked.

By downward induction, if w_n is k-densely marked, so are w_i for all $i \in [n]$. □

The following lemmas concern the growth of strings and marked symbols in an rPcg derivation. We first prove a non-marking version, which we do not use directly but is a stepping stone to the marking version (Lemma 4.9).

Lemma 4.8. *Let $G = (V, \Sigma, S, P)$ be an rcg with $S \Rightarrow^n w \in (V \cup \Sigma)^*$. Let p be the maximum length of the right-hand side of a rule in P. Then $|w| \leq n(p-1)+1$.*

Proof. If $n = 0$ then $w = S$, so the statement is true.

Suppose it is true for n, and $S \Rightarrow^n w' \Rightarrow w$. Then w is obtained from w' by removing a single non-terminal and adding the right-hand side of a production, so $|w| \leq |w'| - 1 + p \leq n(p-1) + p = (n+1)(p-1) + 1$. By induction, it is true for all $n \in \mathbb{N}$. □

Lemma 4.9. *Let $G = (V, \Sigma, S, P)$ be an rcg with $S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n \in (V \cup \Sigma)^*$, such that at most n' branch non-terminals are rewritten in the derivation. Let p be the maximum length of the right-hand side of a rule in P. Then w_n contains at most $n'(p-1)+1$ marked symbols.*

Proof. As for Lemma 4.8. □

Lemma 4.10. *Let $G = (V, \Sigma, S, P)$ be an rcg with $S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n \in (V \cup \Sigma)^*$, with w_n k-densely marked, such that at most n' branch non-terminals are rewritten in the derivation. Let p be the maximum length of the right-hand side of a rule in P. Then $|w_n| < k[n'(p-1)+2]$.*

Proof. A k-densely marked word with m marks has length less than $k(m+1)$; the statement then follows from Lemma 4.9. □

The following order-theoretic lemma allows us to find a pumping threshold which is independent of the word we wish to pump.

Lemma 4.11. *Let $c_1, c_2, \ldots \in \mathbb{N}$, and $n, t \in \mathbb{N}^+$. Then there exists a number $b = b(t) \in \mathbb{N}$ such that if $v_1, v_2, \ldots$ is a sequence of vectors in $\mathbb{N}^n$ satisfying $|v_i| \leq c_i$ for all $i \in \mathbb{N}^+$, then there exist $i_1 < i_2 < \cdots < i_t \in [b]$ such that $v_{i_1} \leq v_{i_2} \leq \cdots \leq v_{i_t}$.*

Proof. See [van der Walt and Ewert 2000, Lemma 3]. □

We can now prove the main theorems of this chapter.

4.1 Permitting Context

Theorem 4.12 (Ogden's Lemma for RPCLs). *For any rPcl L and $k \in \mathbb{N}^+$, there is an $m \in \mathbb{N}^+$ (the pumping threshold) such that for any k-densely marked word $w \in L$ with $|w| > m$ there is a number $l \in [m]$ such that:*

1. *w contains l mutually disjoint non-empty subwords $u_1, u_2, \ldots, u_l$ and l mutually disjoint non-empty subwords $v_1, v_2, \ldots, v_l$, such that for each $i \in [l]$ there exists a $j \in [l]$ such that $v_i \sqsubseteq u_j$;*
2. *if each v_i is replaced with u_i, then the resulting word is still in L, and this process can be applied iteratively to always yield a word in L;*
3. *if v_i contains a marked symbol, then so does u_i;*
4. *there are strictly more marked symbols in $u_1 u_2 \cdots u_l$ than in $v_1 v_2 \cdots v_l$.*

Proof. Let $G = (V, \Sigma, S, P)$ be an rPcg generating L, and let p be the maximum length of a right-hand side of a production in P. Suppose $S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n = w$ is a derivation in G with n' branch points.

Let $i_1, i_2, \ldots, i_{n'}$ be the indices of the sentential forms where a branch symbol is rewritten. By Lemmas 4.7 and 4.10, $|w_{i_j}| < k[(j-1)(p-1)+2]$. Let $b = b(2)$ be the number in Lemma 4.11 for the sequence $k[(j-1)(p-1)+2])_{j\in\mathbb{N}^+}$. Note that b depends only on G and k. If $n' \geq b$, then there exist $r, s \in [b]$ such that $r < s$ and $\mathrm{cnt}^{\mathrm{m}}(w_{i_r}) \leq \mathrm{cnt}^{\mathrm{m}}(w_{i_s})$. To avoid double subscripts we will call these sentential forms u and v respectively. Since a branch node is rewritten in u, there are more marked symbols in v than in u.

Let $l = |\mathrm{cnt}^{\mathrm{m}}(u)|$. Since $\mathrm{cnt}^{\mathrm{m}}(u) \leq \mathrm{cnt}^{\mathrm{m}}(v)$, we can repeat the derivation $u \Rightarrow^* v$ arbitrarily many times, since any needed context will be available. Associate the l non-terminals in u with non-terminals in v injectively, in such a way that each instance of a symbol is associated with another instance of the same symbol, and marked symbols are associated with other marked symbols. Let $u_1, u_2, \ldots, u_l$ be the contributions of the non-terminals in u, and $v_1, v_2, \ldots, v_l$ the contributions of the corresponding non-terminals in v. If we repeat the derivation $u \Rightarrow^* v$, mapping the productions applied to symbols in u to the corresponding symbols in v, the effect will be to replace v_i with u_i for all $i \in [l]$.

By setting m to $k[b(p-1)+2]$, we ensure $n' > b$ (by Lemma 4.10), and since $l = |\mathrm{cnt}^{\mathrm{m}}(u)| \leq |u| \leq m$, we obtain the theorem. □

If all symbols in w are marked (making w 1-densely marked), Theorem 3.10 is obtained as a special case.

Corollary 4.13. *If $w^{(0)}, w^{(1)}, \ldots$ is a sequence of pumped words produced by the replacement operation of Theorem 4.12, then $\#_a(w^{(0)}), \#_a(w^{(1)}), \ldots$ is a* non-decreasing *arithmetic progression (for any symbol a) and the numbers of marked symbols form an* increasing *arithmetic progression.*

Proof. In the replacement operation the count of any terminal symbol, or of marked symbols, increases by the difference between the count in $u_1 u_2 \ldots u_l$ and the count in $v_1 v_2 \ldots v_l$. □

This observation about the numbers of a symbol in pumped words is, naturally, also true for the special case of the pumping lemma, but this fact does not seem to have been noted before in that case.

We now give an example of how Theorem 4.12 can be used to prove a language is not an rPcl, with the help of Corollary 4.13.

Example 4.14. Let $L = \{w \in \{a, b\}^* : \#_a(w) = 2^k, k \in \mathbb{N}, bb \sqsubseteq w\}$.

L satisfies the pumping lemma for rPcls – and even the pumping lemma for regular languages – but does not satisfy Theorem 4.12, and so is not an rPcl. This shows that Theorem 4.12 is strictly stronger than Theorem 3.10.

To see that it satisfies the pumping lemma, it suffices to observe that in any word of L there is a b which can be replaced by bb while remaining in L. On the other hand, suppose it satisfies Theorem 4.12 for $k = 3$ with threshold m. Then let $w = bba^{2^m}$ with all the a's and neither of the b's marked; this word is 3-densely marked. Let $w^{(0)}, w^{(1)}, \ldots$ be the sequence of pumped words. Since only the a's are marked, the sequence $\#_a(w^{(0)}), \#_a(w^{(1)}), \ldots$ is an increasing arithmetic progression, by Corollary 4.13; on the other hand, $\#_a(v)$ is a power of two for all v in L. This is a contradiction, showing that L does not satisfy Theorem 4.12, and so cannot be an rPcl.

Although L itself satisfies the pumping lemma, it can be shown that it is not an rPcl using the pumping lemma in conjunction with known closure results. Let f be a homomorphism from $\{a, b\}$ to $\{a\}$ defined by $f(a) = a$ and $f(b) = \lambda$. As shown by Zetzsche [2010, Corollary 1], the rPcls are closed under erasing homomorphisms, but $f(L) = \{a^{2^k} : k \in \mathbb{N}\}$ does not satisfy the pumping lemma, and is therefore not an rPcl, and so L is not either.

One can construct similar examples which require stronger closure properties. The language $L = \{a^{n_1} b^{m_1} \cdots a^{n_k} b^{m_k} : k = 2^p, p \in \mathbb{N} \text{ and } n_i, m_i \geq 2 \text{ for all } i \in [k]\}$ does not satisfy Theorem 4.12, but although any homomorphic image of L satisfies the pumping lemma, there is a generalised state-machine mapping which maps this language to $\{a^{2^k} : k \in \mathbb{N}\}$ (and rPcls are closed under such mappings, by [Dassow and Păun 1989, Table 1.3.1] and [Zetzsche 2010, Corollary 1]). Specifically, the mapping outputs a at every boundary between a's and b's, and λ everywhere else.

The requirement for the marking to be dense makes it more difficult to find examples to which the theorem applies than, for example, for Ogden's Lemma for context-free languages. Unfortunately, density (or some other way to obtain a bound similar to Lemma 4.10) is used in an essential way in the proof. While an alternative proof method which would remove this requirement may exist, it seems that it would require a much deeper understanding of the structure of rPcg derivations – for example, when and how derivation steps can be reordered. The situation in the forbidding case is similar.

On the other hand, the density of markings makes it easier to find examples of non-rPcls which nevertheless satisfy Theorem 4.12.

Example 4.15. Let $L = \{a^m b^n : n \geq m = 2^k, k \in \mathbb{N}\}$. This satisfies Theorem 4.12 because, by setting the thresholds appropriately, one can ensure that at least two marks appear in the b's alone, and therefore the b's can be pumped without affecting

the a's. However, L is not an rPcl. This can be seen using the same homomorphism f as above: $f(L)$ is again $\{a^{2^k} : k \in \mathbb{N}\}$.

4.2 Forbidding Context

We now prove the corresponding result to Theorem 4.12 for rFcls.

Theorem 4.16 (Ogden's Lemma for RFCLs). *For any rFcl L and $k, t \in \mathbb{N}^+$, there is an $m \in \mathbb{N}^+$ (the shrinking threshold) such that for any k-densely marked word $w \in L$ with $|w| > m$ there are t words $w^{(1)}, w^{(2)}, \ldots, w^{(t)} = w$ such that for every $j \in \{2, 3, \ldots, t\}$:*

1. *there exists a number $l \in [m]$;*
2. *$w^{(j)}$ contains l mutually disjoint non-empty subwords $u_1, u_2, \ldots, u_l$ and l mutually disjoint non-empty subwords $v_1, v_2, \ldots, v_l$, such that for each $i \in [l]$, there exists a $k \in [l]$ such that $v_i \sqsubseteq u_k$;*
3. *if each u_i is replaced with v_i, then the resulting word is $w^{(j-1)}$;*
4. *if v_i contains a marked symbol, then so does u_i;*
5. *there are strictly more marked symbols in $u_1 u_2 \cdots u_l$ than in $v_1 v_2 \cdots v_l$.*

Proof. Let $G = (V, \Sigma, S, P)$ be an rFcg generating L, and let p be the maximum length of a right-hand side of a production in P. Suppose $S = w_1 \Rightarrow^* w_2 \Rightarrow^* \cdots \Rightarrow^* w_{n'} = w$ is a derivation in G with n' branch points, where $w_1, w_2, \ldots, w_{n'}$ are the sentential forms in which a branch symbol is rewritten.

By Lemmas 4.7 and 4.10, $|w_j| < k[(j-1)(p-1)+2]$. Let $b = b(t)$ be the number in Lemma 4.11 for the sequence $(k[(j-1)(p-1)+2])_{j \in \mathbb{N}^+}$. Note that b depends only on G, k and t. If $n' \geq b$, then there exist $r_1 < r_2 < \cdots < r_t \in [b]$ such that $\mathrm{cnt}^{\mathrm{m}}(w_{r_1}) \leq \mathrm{cnt}^{\mathrm{m}}(w_{r_2}) \leq \cdots \leq \mathrm{cnt}^{\mathrm{m}}(w_{r_t})$.

We now follow essentially the opposite procedure to the proof of Theorem 4.12, since in the forbidding case, less context allows one to do more: to obtain $w^{(j-1)}$ from $w^{(j)}$, let $l = |\mathrm{cnt}^{\mathrm{m}}(w_{r_{j-1}})|$, let $u_1, u_2, \ldots, u_l$ be the contributions of the nonterminals in $w_{r_{j-1}}$, and let $v_1, v_2, \ldots, v_l$ be the contributions of the corresponding nonterminals in w_{r_j}. The replacement operation then results from applying the same rules to the nonterminals in $w_{r_{j-1}}$ (and their descendants) as they were originally applied to the nonterminals in w_{r_j}.

By setting m to $k[b(p-1)+2]$, we ensure $n' > b$ (by Lemma 4.10), and since $l = |\mathrm{cnt}^{\mathrm{m}}(u)| \leq |u| \leq m$, we obtain the theorem. □

Unlike the pumping lemma for rPcls, the shrinking lemma for rFcls gives no information about the possible length sets of rFcls, because all unary languages (and thus all length sets) satisfy the shrinking property. Unfortunately, Theorem 4.16 does not improve this situation: again, all unary languages satisfy it.

This observation also immediately leads to an example of a non-rFc language which satisfies Theorem 4.16: for instance, $\{a^i : M_i \text{ halts}\}$ where $M_1, M_2, \ldots$ is a standard enumeration of Turing machines (this is not an rFcl since rcls are computable, by Theorem 3.7).

To construct an example of a language which satisfies the shrinking lemma for rFcls but not Theorem 4.16, it seems we cannot use the same trick as in Example 4.14 (i.e., inserting arbitrarily many junk symbols at every position), because inserting padding does not cause a language to satisfy the shrinking lemma: we can simply take an example with minimal padding, and then any shrinking will necessarily affect non-padding symbols.

In this chapter, we have proven two generalisations of Ogden's Lemma: one for rPcls and another for rFcls. Examples were given for the rPcl lemma, but none were found for the rFcl version. In the next chapter we will move on to the remaining type of grammar to be discussed in this work: ET0L systems. This will prepare the ground for a rather different generalisation of Ogden's Lemma for the languages they generate.

Chapter 5

ET0L Systems

ET0L systems are a class of L systems which generalise the context-free grammars. However, rather than arising from the study of human and computer languages, L systems were originally developed by theoretical biologist Aristid Lindenmayer to model phenomena relating to growth and morphology, and have since been used in computer graphics and computer-aided design.

Our definition of an ET0L language differs slightly from the usual: we use a single symbol, rather than a word, as the axiom. This simplifies our work slightly, but has no effect on the class of languages generated (see Theorem 5.4 below), unlike in the simpler types of L systems (e.g. 0L systems).

Definition 5.1. An *ET0L system* is a tuple $G = (V, \Sigma, \mathcal{T}, S)$ where Σ, the terminal alphabet, is a non-empty subset of the alphabet V, $S \in V$ and $\mathcal{T}$ is a finite collection $\{T_1, \ldots, T_n\}$ of tables. Each table is a finite set of productions $A \to u$, such that for every $A \in V$ there is a production $A \to u$ in T_i for every $i \in [n]$.

Definition 5.2. If $G = (V, \Sigma, \mathcal{T}, S)$ is an ET0L system, with $A_1, A_2, \ldots, A_n \in V$, and $u_1, u_2, \ldots, u_n \in V^*$, then $A_1 A_2 \cdots A_n \Rightarrow u_1 u_2 \ldots u_n$ if there is a table $T \in \mathcal{T}$ such that $(A_i \to u_i) \in T$ for all $i \in [n]$. We write $\Rightarrow^*$ for the reflexive and transitive closure of $\Rightarrow$.

Definition 5.3. If $G = (V, \Sigma, \mathcal{T}, S)$ is an ET0L system, then it *generates* the language $L(G) = \{w \in \Sigma^* : S \Rightarrow^* w\}$. Such languages are called *ET0L languages*.

ET0L stands for Extended Table-driven 0-context Lindenmayer-system: extended refers to the distinction between terminal and non-terminal symbols, and table-driven refers to the tables of productions. Without these (i.e. when $\Sigma = V$ and $\mathcal{T}$ is a singleton) we have a plain 0L system.

Theorem 5.4. *If a starting word (axiom) ω is used instead of the starting symbol S in the definition of ET0L languages, the same class of languages is generated.*

Proof. Any ETOL language by our definition is also an ETOL language with an axiom, since we can take $\omega = S$. To see that the converse is true, add a new non-terminal symbol S, and add a production $S \to \omega$ to every table. Then $S \Rightarrow \omega$, and since there are no other productions involving S, we have $S \Rightarrow^* w$ if and only if $\omega \Rightarrow^* w$. □

To see the power of ETOL systems, consider the following example.

Example 5.5. The language of composite numbers in unary, $\{a^{pq} : 2 \leq p, q\}$ is an ETOL language generated by the system $G = \{V, \{a\}, \{T_1, T_2, T_3\}, S\}$, where $V = \{S, P, Q, X, a\}$ and

$$T_1 = \{S \to PP, P \to PP, P \to P, Q \to X, X \to X, a \to X\}$$
$$T_2 = \{S \to X, P \to PQ, Q \to Q, X \to X, a \to X\}$$
$$T_3 = \{S \to X, P \to aa, Q \to a, X \to X, a \to X\}.$$

Since X is not a terminal symbol and cannot be rewritten to any other symbol, in any terminating derivation, T_1 must be applied one or more times, followed by T_2 zero or more times, and finally T_3 once. It is not hard to see that such a derivation has the form

$$S \Rightarrow^n P^p \Rightarrow^m (PQ^m)^p \Rightarrow (a^{m+2})^p = a^{(m+2)p},$$

where $m \in \mathbb{N}$, $n \geq 1$ and $2 \leq p \leq 2^n$, and furthermore all such derivations are possible.

We now define two subclasses of ETOL systems, one of which is the subject of a previously known iteration theorem, and the other is an equivalent form which is useful in some proofs.

Definition 5.6. An ETOL system G is called *propagating* (an *EPTOL system*) if every production $A \to u$ has $|u| \geq 1$. It is called *deterministic* (an *EDTOL* system) if whenever $A \to u$ and $A \to v$ appear in the same table then $u = v$.

5.1 Relationship to Random Context

There are ETOL languages which are not random permitting-context languages – for example $\{a^{2^n} : n \in \mathbb{N}\}$. On the other hand, the ETOL languages form a strict subclass of the random forbidding-context languages.

Theorem 5.7. *For every ETOL system G, there exists a random forbidding-context grammar G' which generates the same language.*

Proof. This is most easily done for EPT0L systems, which generate the same languages as the ET0L systems [Rozenberg and Salomaa 1980, Theorem 1.6]. One builds an rFcg to simulate an EPT0L system (this is more natural than directly simulating an ET0L system since the rules of an rFcg, like those of an EPT0L, cannot have λ on the right-hand side). In each sentential form of this grammar, there is at most one instance of a "leader" symbol which ensures that all the symbols are rewritten in lockstep; this leader can non-deterministically label itself by one of the tables in the EPT0L system; productions in the rFcg corresponding to those in the EPT0L system are forbidden except when the leader is labelled by the correct table.

The formal details of this construction can be found in Dassow and Păun [1989, Theorem 1.5.12], which also demonstrates the strictness of the inclusion. □

As mentioned previously, under the restriction of finite index, the ET0L languages coincide with the random-context languages. However, the definition of index has to be modified slightly for ET0L systems and languages: in an ET0L system, terminal symbols can be rewritten, so in an ET0L system, a symbol A is called active if there is a rule $A \rightarrow u$ where $A \neq u$, and the number of active symbols is considered instead of the number of non-terminals.

5.2 A Previous Iteration Theorem

Ehrenfeucht and Rozenberg [1974] proved a pumping lemma for EDT0L languages. This lemma can only be applied to f-random words, which we define here.

Definition 5.8. If f and g are functions from $\mathbb{N}$ to $\mathbb{N}$, f is said to be *eventually larger than* (resp. *eventually less than*) g if there exists an $N \in \mathbb{N}$ such that $f(n) > g(n)$ (resp. $f(n) < g(n)$) for all $n > N$.

Definition 5.9. A function $f : \mathbb{N} \rightarrow \mathbb{N}$ is called *slow* if, for all $\epsilon > 0$, $f(n)$ is eventually larger than n^{ϵ}.

For example, the logarithmic and constant functions are slow.

Definition 5.10. For any $f : \mathbb{N} \rightarrow \mathbb{N}$, a word w is called *f-random* if there is no word u with $|u| > f(|w|)$ such that $w = v_1 u v_2 u v_3$.

Roughly, a random word does not contain two copies of any long subword.

Theorem 5.11 (Pumping Lemma for EDT0L Languages). *If L is an EDT0L language and f is a slow function, then there exists an $l \in \mathbb{N}$ such that for any f-random word $w \in L$ with $|w| > l$, w can be written as $w = u_0 v_1 u_1 v_2 u_2 \ldots v_n u_n$ (with $|v_1 v_2 \ldots v_n| > 0$) such that $u_0 v_1^t u_1 v_2^t u_2 \ldots v_n^t u_n \in L$ for all $t \in \mathbb{N}$.*

Not only does this theorem apply only to certain words, but also only to certain languages, since some languages contain no such words (for example, languages over a singleton alphabet).

Having reviewed the ET0L class of Lindenmayer systems and the languages we generate, we now move on to our main work on these languages: a version of Ogden's Lemma. This is more like the result for rPcls than the above pumping lemma for EDT0L languages: it can move the subwords around, and even – unlike Theorem 4.12 – increase the length of the word superlinearly.

Chapter 6

Ogden's Lemma for ET0L Languages

In this chapter, we develop a version of Ogden's Lemma for ET0L languages. As a corollary, we derive a pumping lemma. We give some examples, and use the theorem to prove Ehrenfeucht and Rozenberg's theorem about rare and non-frequent symbols.

Because ET0L derivations operate on all the symbols of a sentential form in parallel, it is not enough to find a single repeated branch node, as in Ogden's Lemma: in repeating the segment of the derivation between the two nodes, we would have to rewrite other symbols, but if corresponding symbols did not appear in the original derivation segment, it would not be clear how they should be rewritten. To avoid this obstacle, we require that the same set of symbols appear in the sentential forms where the nodes appear. Since there are only finitely many such sets, we can find such points in any large enough derivation.

Definition 6.1. If G is an ET0L system, and $w_1, w_2, \ldots, w_n$ are sentential forms in G, with $S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n$, then a tree τ is a *derivation tree of* w_n *in* G if

1. the root of τ is labelled with S;
2. a node has children labelled with the symbols of a word u, in order, if the symbol corresponding to that node was rewritten to u in the derivation.

A *level* refers to the set of nodes at the same depth (and therefore corresponds to one of the sentential forms in a derivation). The root is called the first level, its children the second level, and so on. The *maximum out-degree* of a tree is the maximum number of children of any node in the tree.

We extend the concept of marking to nodes of a derivation tree of a marked word w thus:

1. a leaf is marked if it corresponds to a marked symbol in w;

2. a non-leaf node is marked if any of its child nodes are marked.

We will call a node a *branch node* if it has more than one marked child.

When the context allows, we will sometimes not distinguish between an instance of a symbol, the corresponding node in a derivation tree, and the subtree rooted at that node.

6.1 Main Results

The following simple combinatorial lemma on trees will be needed. It basically gives a lower bound on the height of a derivation tree in terms of the length of the derived word, but is modified slightly to be useful for words with marked positions. The same fact is used to prove Ogden's original Lemma.

Lemma 6.2. *For any $m \in \mathbb{N}$, if a tree with maximum out-degree at most p has more than p^m marked leaves, it must have a path from the root to a leaf with more than m branch nodes.*

Proof. We prove this by structural induction on the tree.

A tree consisting of a single leaf has at most $1 = p^0$ marked leaf, and so trivially satisfies the lemma.

Let τ be a (non-leaf) tree with more than p^m marked leaves, for some m, and suppose its children $\tau_1, \tau_2, \ldots, \tau_n$ satisfy the lemma ($n \in [p]$). If only a single τ_i is marked ($i \in [n]$), then τ has the same number of marked leaves and a path with the same number of branch nodes as τ_i, and therefore τ satisfies the lemma.

Otherwise, the root of τ is a branch node. One of the τ_i must have more than $p^m/n \geq p^{m-1}$ marked leaves, and therefore has a path with more than $m-1$ branch nodes. Since the root of τ is also a branch node, τ has a path with more than m branch nodes. □

We can now prove our main result.

Theorem 6.3. *If L is an ETOL language, then there exists an $l \in \mathbb{N}$ (which we will call the* threshold *for L) such that for any word $w \in L$ with at least l marked positions,*

1. *w can be written as $w = u_1 u_2 \cdots u_n$ and each u_i can be written $u_i = v_{(i,1)} v_{(i,2)} \cdots v_{(i,n_i)}$ (we will denote the set of subscripts of v, i.e. $\{(i,j) : i \in [n], j \in [n_i]\}$, by I);*

2. *there is a map $\phi : I \to [n]$ such that if each $v_{(i,j)}$ is replaced with $u_{\phi(i,j)}$, then the resulting word is still in L, and this process can be applied iteratively to always yield a word in L;*

3. *if* $v_{(i,j)}$ *contains a marked position then so does* $u_{\phi(i,j)}$;

4. *there is an* $(\mathbf{i},\mathbf{j}) \in I$ *such that* $\phi(\mathbf{i},\mathbf{j}) = \mathbf{i}$, *and there are at least two marked positions in* $v_{(\mathbf{i},\mathbf{j})}$ *and at least one in* $u_{\mathbf{i}}$ *but outside of* $v_{(\mathbf{i},\mathbf{j})}$.

Proof. Let $G = (V, \Sigma, \mathcal{T}, S)$ be an ET0L system. Let p be the maximum length of the right-hand side of a rule in G, and let $m = |V|4^{|V|}$. Let w be a word in $L(G)$ with at least $p^m + 1$ marked positions.

Consider a derivation tree of w in G. Since the tree has more than p^m marked leaves, it must have a path with more than m branch nodes.

There must be a symbol $A \in V$ which appears more than $m/|V| = 4^{|V|}$ times as the label of a branch node on this path. For each of these, consider the set of symbols which appear on the same level in the derivation tree, and the set of *marked* symbols which appear on the same level. Since there are only $4^{|V|}$ distinct such pairs of sets, there must be two branch nodes labelled by A, with one an ancestor of the other, with the same pair of sets. Call the ancestor node A_1 and the descendant A_2, and the sentential forms in which they appear w_1 and w_2, respectively. This situation is illustrated on the left of Figure 6.1.

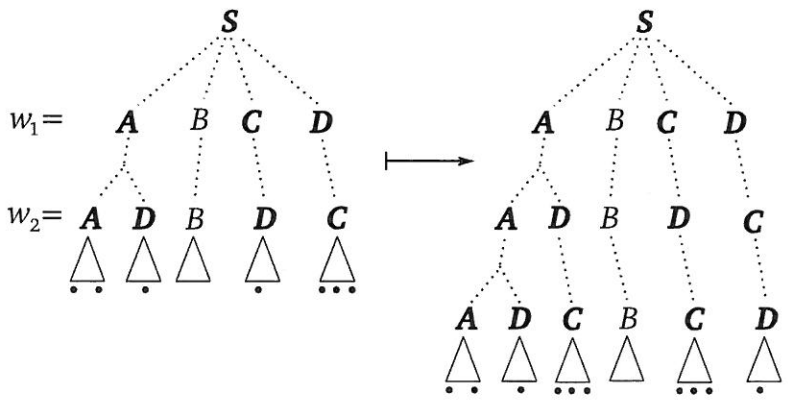


Figure 6.1: A situation which allows a portion of a derivation to be repeated, and the result of such repetition. Bold symbols and dots indicate marked symbols. Note the repetition of the set $\{A, B, C, D\}$ of symbols, and $\{A, C, D\}$ of marked symbols.

The sequence of tables which were applied to w_1 to derive w can be applied again to w_2. Due to non-determinism, there may be several possible results from this, but one result can be obtained by replacing each subtree corresponding to an instance of a symbol in w_2 with a subtree corresponding to an instance of the same symbol in w_1. To obtain the desired result, it is necessary to choose A_1 when an instance of A is sought, and if there is a marked instance of a symbol, we must choose it. This replacement can be applied as many times as desired. This gives us parts 2 and 3 of the theorem.

Part 4 follows from the fact that A_2 is replaced with A_1, and both are branch nodes. Set $l = p^m + 1$ to complete the proof. □

It will be useful to give here a more formal description of the replacement operation in part 2 (which we will call the *pumping operation*). The operation produces the words $w^{(t)}$ for all $t \in \mathbb{N}$, where

$$v_{(i,j)}^{(0)} = v_{(i,j)} \tag{6.1}$$

$$u_i^{(t)} = v_{(i,1)}^{(t)} v_{(i,2)}^{(t)} \cdots v_{(i,n_i)}^{(t)} \tag{6.2}$$

$$v_{(i,j)}^{(t+1)} = u_{\phi(i,j)}^{(t)} \tag{6.3}$$

$$w^{(t)} = u_1^{(t)} u_2^{(t)} \cdots u_n^{(t)}. \tag{6.4}$$

Note that $w^{(0)} = w$. This pumping operation is more complicated than that in Theorems 3.8 and 5.11. At least some complication is necessary: those operations generate a sequence of words whose lengths form an infinite arithmetic progression, but there are infinite ETOL languages which do not contain such sequences.

It may be interesting to remark that $\{w^{(t)} : t \in \mathbb{N}\}$ is a homomorphic image of a deterministic zero-context Lindenmayer (DOL) language. Consider $\{v_{(i,j)} : (i,j) \in I\}$ as symbols (rather than the words they stand for). The replacement operation defined above can be interpreted as a homomorphism from this set to itself, and if we use the axiom $\omega = v_{(1,1)} v_{(1,2)} \cdots v_{(1,n_1)} v_{(2,1)} \cdots v_{(n,n_i)}$, we obtain the sequence corresponding to $\{w^{(t)} : t \in \mathbb{N}\}$. We can map the elements of this sequence to words by the homorphism which sends each $v_{(i,j)}$ to the word it stands for.

There is a simpler form of Theorem 6.3 which may nevertheless sometimes be useful. Their relationship is analogous to the relationship between Ogden's Lemma and the pumping lemma for context-free languages.

Corollary 6.4. *If L is an ETOL language, then there exists an $l \in \mathbb{N}$ such that for any word $w \in L$ with $|w| \geq l$,*

1. *w can be written as $w = u_1 u_2 \cdots u_n$ and each u_i can be written $u_i = v_{(i,1)} v_{(i,2)} \cdots v_{(i,n_i)}$ (we will again use I for the set of subscripts of v);*
2. *there is a map $\phi : I \to [n]$ such that if each $v_{(i,j)}$ is replaced with $u_{\phi(i,j)}$, then the resulting word is still in L, and this process can be applied iteratively to always yield a word in L;*
3. *there is an $(\mathbf{i},\mathbf{j}) \in I$ such that $\phi(\mathbf{i},\mathbf{j}) = \mathbf{i}$, and $v_{(\mathbf{i},\mathbf{j})}$ is a strict subword of $u_{\mathbf{i}}$.*

Proof. This follows directly from Theorem 6.3 if we mark all positions in w. □

We will now see how Theorem 6.3 and its proof apply to an example.

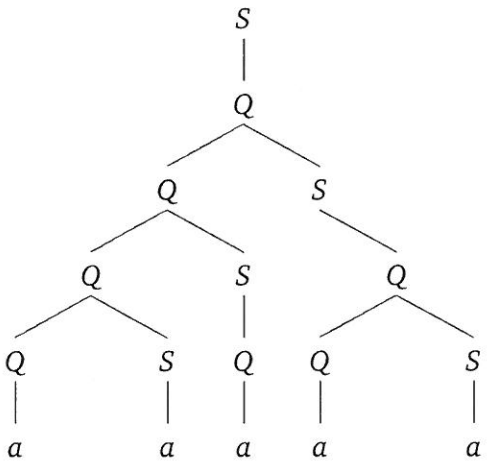


Figure 6.2: An example ETOL derivation

Example 6.5. The set $\{a^{F_n} : n \in \mathbb{N}^+\}$, where F_n denotes the nth Fibonacci number, is an ETOL language generated by the system $(V, \Sigma, \mathcal{T}, S)$ where $V = \{Q, S, a\}$, $\Sigma = \{a\}$, and $\mathcal{T} = \{\{Q \to QS, S \to Q, a \to a\}, \{Q \to a, S \to a, a \to a\}\}$.

A derivation tree in this system for a^5 is given in Fig. 6.2. For simplicity, we will consider the case where all symbols are marked, as in Corollary 6.4.

Observe that the third and fourth level of the tree contain the same set of symbols – $\{Q, S\}$ – and since all symbols are marked, also the same set of marked symbols. Furthermore, the Q in the third level is an ancestor of the leftmost Q in the fourth level, and both are branch nodes, so the pumping operation can be applied to these two levels of the tree.

This leads us to break the string up as $\overline{\overline{aa}\,\overline{a}}\,\overline{\overline{aa}}$, where the outer brackets delimit the subwords u_1 and u_2, and the inner brackets, $v_{(1,1)}$, $v_{(1,2)}$ and $v_{(2,1)}$, and to set $\phi(1,1) = \phi(2,1) = 1$ and $\phi(1,2) = 2$. Applying the pumping operation yields $w^{(1)} = \overline{\overline{aaa}\,\overline{aa}}\,\overline{\overline{aaa}}$, which is indeed a string in the language. Applying it again yields $w^{(2)} = \overline{\overline{aaaaa}\,\overline{aaa}}\,\overline{\overline{aaaaa}}$, and so on.

Theorem 6.3 is a necessary condition for a language to be ETOL, but it is not sufficient. In fact, as the following example shows, a language which satisfies it may even be uncomputable, while all ETOL languages are computable, by Theorem 3.7.

Example 6.6. Let $M_1, M_2, \ldots$ be a standard enumeration of the set of Turing machines, and let $L = \{a^{2^k(2i+1)} : M_i \text{ halts}, k \in \mathbb{N}\}$. This language is not computable, and is therefore not ETOL, but it satisfies the conditions of Theorem 6.3.

The halting problem is reducible to L, since $a^{2i+1} \in L$ if and only if M_i halts. Thus L is uncomputable.

Now we will show that it satisfies Theorem 6.3 with threshold $l = 3$. Let $w = a^{2^k(2i+1)}$ be a word in L with at least three marked positions, where $k \in \mathbb{N}$ and M_i halts. Let $u_1 = w$ and let $v_{(1,1)}v_{(1,2)} = w$ in such a way that $v_{(1,1)}$ contains at least two marked positions and $v_{(1,2)}$ contains at least one. Let $\phi(1,1) = \phi(1,2) = 1$. Then applying the pumping operation yields the words $a^{2^{k+1}(2i+1)}, a^{2^{k+2}(2i+1)}, \ldots$, which are all in L, and all the other conditions of Theorem 6.3 are satisfied.

This example can be extended to a whole family of languages by replacing the condition "M_i halts" with "$i \in S$" for any set $S \subseteq \mathbb{N}$. Each choice of S yields a distinct language, showing that there are uncountably many languages over $\{a\}$ which satisfy Theorem 6.3, while there are only countably many ETOL languages over any fixed alphabet.

Since every ETOL language is also an rFcl, one might wonder about the relationship between Theorems 4.16 and 6.3. For instance, do these two theorems jointly characterise the ETOL languages? They do not, and this is shown by the same example: since it is an unary language, it also satisfies Theorem 4.16.

6.2 Applications

We will now prove two facts about the pumping operation which will help in using Theorem 6.3 to prove that certain languages are not ETOL languages.

Unlike the situation in Theorem 4.12, the pumping operation of Theorem 6.3 can initially reduce the number of marked symbols by replacing subwords with many marked symbols by subwords with few. However, eventually the number of marked symbols must increase, as the following shows.

Corollary 6.7. *If L is an ETOL language with threshold l, and $w \in L$ has at least l marked symbols, then the number of marked symbols in $w^{(t)}$ tends to infinity, where $w^{(t)}$ is the result of applying the pumping operation t times. Specifically, $w^{(t)}$ contains at least $t + 2$ marked symbols for all $t \in \mathbb{N}$.*

Proof. Let $u_i^{(t)}$ and $v_{(i,j)}^{(t)}$ be as in (6.1)–(6.4), and $\mathbf{i}$ and $\mathbf{j}$ as in part 4 of Theorem 6.3.

By part 4 of Theorem 6.3, $v_{(\mathbf{i},\mathbf{j})}^{(0)}$ contains at least two marked symbols.

Suppose $v_{(\mathbf{i},\mathbf{j})}^{(t)}$ contains at least $t + 2$ marked symbols. Then $v_{(\mathbf{i},\mathbf{j})}^{(t+1)} = u_{\mathbf{i}}^{(t)}$; this has $v_{(\mathbf{i},\mathbf{j})}^{(t)}$ as a subword, which contributes $t + 2$ marked symbols. However, $u_{\mathbf{i}}$ also contains a marked symbol outside of $v_{(\mathbf{i},\mathbf{j})}$, and since subwords containing a marked symbol are replaced with other such subwords, this property is inherited: $u_{\mathbf{i}}^{(t)}$ has a marked symbol outside of $v_{(\mathbf{i},\mathbf{j})}^{(t)}$. This contributes another marked symbol, so in total there are at least $t + 3$ marked symbols in $v_{(\mathbf{i},\mathbf{j})}^{(t+1)}$.

By induction, we conclude that $w^{(t)}$ contains at least $t+2$ marked symbols for all $t \in \mathbb{N}$. □

On the other hand, the number of symbols cannot grow superexponentially by this pumping operation:

Corollary 6.8. *If L is an ETOL language with threshold l, and $w \in L$ is at least l symbols long, then $|w^{(t)}| \leq |I|^t |w|$, where $w^{(t)}$ is the result of applying the pumping operation t times and I is the set defined in part 1 of Theorem 6.3.*

Proof. Let $u_i^{(t)}$ and $v_{(i,j)}^{(t)}$ be as in (6.1)–(6.4).

For all $(i,j) \in I$ and all t, we have $|v_{(i,j)}^{(t+1)}| = |u_{\phi(i,j)}^{(t)}| \leq |w^{(t)}|$.

Since $|w^{(t+1)}| = \sum_{(i,j)\in I} |v_{(i,j)}^{(t+1)}|$, we get $|w^{(t+1)}| \leq |I|\,|w^{(t)}|$. □

Although Corollary 6.7 relates to marked symbols and Corollary 6.8 to the total number of symbols, each applies to both, since the number of marked symbols is always less than or equal to the total number of symbols.

Corollaries 6.7 and 6.8 together with Theorem 6.3 can be useful in proving that certain languages are not ETOL languages.

Example 6.9. We wish to prove that $L = \{a^n b^m : n \in \mathbb{N}, m \geq 2^{2^n}\}$ is not an ETOL language. Suppose it is. Then, let l be pumping threshold for L, $m = 2^{2^l}$, and $w = a^l b^m$ with all the a's and none of the b's marked. Theorem 6.3 applies to w. As before, we will denote by $w^{(t)}$ the word obtained by applying the pumping operation t times.

By Corollary 6.7, $\#_a\left(w^{(t)}\right) \geq t+2$; on the other hand, $\#_b\left(w^{(t)}\right) \leq |w^{(t)}| \leq |I|^t |w|$ by Corollary 6.8. Since any doubly-exponential function is eventually larger than any singly-exponential function, we find that $|I|^t |w| < 2^{2^{t+2}}$ for large enough t. So $w^{(t)} \notin L$ for large enough t, which contradicts Theorem 6.3; thus L cannot be an ETOL language.

If all positions were marked, the pumping operation could increase the number of b's while leaving the a's unchanged, and so generate strings in L. Therefore the ability of Theorem 6.3 to focus on marked symbols is really necessary in this example.

A similar argument can be used if 2^{2^n} is replaced with any superexponential function.

A theorem by Ehrenfeucht and Rozenberg [1976] on rare and nonfrequent symbols in ETOL languages can be deduced from Theorem 6.3 without making any additional reference to the structure of ETOL derivations.

Definition 6.10. Let L be a language over Σ, and $B \subseteq \Sigma$.

1. B is called *nonfrequent* if there is a constant c_B such that $\#_B(w) \leq c_B$ for all $w \in L$. Otherwise it is called *frequent*.

2. B is called *rare* if for every $k \in \mathbb{N}^+$, there exists an $n_k \in \mathbb{N}^+$ such that if $\#_B(w) \geq n_k$ then the distance between any two appearances in w of symbols from B is at least k.

Theorem 6.11. *Let L be an ETOL language over Σ, with B a non-empty subset of Σ. If B is rare in L, then B is nonfrequent in L.*

Proof. Let l be the number for L from Theorem 6.3. Suppose B is frequent in L. Then there must be a word $w \in L$ with more than l symbols from B. If we mark them all (and only them) Theorem 6.3 applies. Again we will denote by $w^{(t)}$ the result of pumping this word t times.

By Corollary 6.7, $w^{(t)}$ contains at least $t+2$ symbols from B. However, by part 4 of Theorem 6.3, the subword $v_{(i,j)}$ contains two marked symbols, and this subword appears in all $w^{(t)}$, so these two symbols are a fixed distance apart. So B is not rare in L. □

This can be used to prove that, for example, $\{(ga^k)^k : k \in \mathbb{N}^+\}$ is not an ETOL language. Theorem 6.3 is, however, stronger than this one: the language of Example 6.9 has no rare set of symbols, so Theorem 6.11 cannot show that it is not an ETOL language.

In this chapter we have seen a version of Ogden's Lemma for the ETOL languages. This completes the trilogy of results mentioned in the title of the work. It also enabled a straightforward proof of an independently interesting result, and a pumping lemma for these languages.

Chapter 7

Conclusion and Future Work

In Chapter 4, generalisations of Ogden's Lemma for the two subclasses of random context languages were proved – these strengthen the known pumping and shrinking lemmas for these classes. In Chapter 6, the same was done for ETOL languages, which form a subclass of the rFcls of independent interest. In the case of ETOL languages, a pumping lemma appears not to have been already known, and thus one was given as a special case of the main result. For the theorems on ETOL and rPc languages, an example of a language which does not satisfy it was given (thus proving it not to belong to the respective class), and for all three classes an example was given of a language which does not belong to the class but nevertheless satisfies the theorem (thus showing the theorems to be a necessary but not sufficient condition).

Before these results were proved, overviews of the respective classes were given (in Chapters 3 and 5), focusing on their relationships to each other and to well-known classes such as the context-free and context-sensitive languages, and previously known necessary conditions (especially other iteration theorems).

It would be valuable and interesting to try to prove versions of Ogden's Lemma for rPcls and rFcls which, unlike Theorems 4.12 and 4.16 but like Ogden's Lemma itself, do not place any restrictions on how the word is marked (beyond requiring a minimum number of marks). This may require a deeper investigation into the structure of derivations in rPcgs and rFcgs. This may be possible, say, for rPcgs only, as in the results of Drewes and van der Merwe [2008] (that path languages of random permitting-context tree grammars are regular, while those of random forbidding-context tree grammars need not even be context-free).

Another avenue for further research would be to try to generalise to random context other necessary conditions known in the theory of context-free languages, such as Sokolowski's condition [Sokolowski 1978].

Sokolowski's condition itself is not true for even the finite-index random-context languages but the following generalisation, for example, may hold:

Conjecture 7.1. *Let $L \subseteq \Sigma^*$ be an rPcl and $\Sigma' \subseteq \Sigma$. There exists a number $k \in \mathbb{N}$ such that if $n \geq k$ and $u_0 x u_1 x u_2 \dots u_{n-1} x u_n \in L$ for all $x \in \Sigma'^*$, then there exist $x_1, x_2, \dots, x_n \in \Sigma'^*$ not all equal, such that $u_0 x_1 u_1 x_2 u_2 \dots u_{n-1} x_n u_n \in L$.*

The language $\{(ax)^k : x \in \Sigma^*, k \in \mathbb{N}^+\}$ is easily seen to be ET0L, and thus rFc, for any alphabet Σ. This shows that the conjecture does not hold for these classes.

Finally, it should be reasonably straightforward to prove these results for the corresponding classes of picture languages (as was done for the pumping and shrinking lemmas by Ewert [1999]) and tree languages [Drewes *et al.* 2005].

Bibliography

[Atcheson *et al.* 2006] B. Atcheson, S. Ewert, and D. Shell. A note on the generative capacity of random context. *South African Computer Journal*, 36:95–98, 2006.

[Bader and Moura 1982] C. Bader and A. Moura. A generalization of Ogden's lemma. *Journal of the Association for Computing Machinery*, 29(2):404–407, 1982.

[Bar-Hillel *et al.* 1961] Y. Bar-Hillel, M. Perles, and E. Shamir. On formal properties of simple phrase structure grammars. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung*, 14:143–172, 1961.

[Beauquier 1979] J. Beauquier. Deux familles de langages incomparables. *Information and Control*, 43(2):101–122, 1979.

[Dassow and Păun 1989] J. Dassow and G. Păun. *Regulated Rewriting in Formal Language Theory*. EATCS Monographs on Theoretical Computer Science. Springer Berlin, 1989.

[Drewes and van der Merwe 2008] F. Drewes and B. van der Merwe. Path languages of random permitting context tree grammars are regular. *Fundamenta Informaticae*, 82:47–60, 2008.

[Drewes *et al.* 2005] F. Drewes, C. du Toit, S. Ewert, J. Högberg, B. van der Merwe, and A. van der Walt. Random context tree grammars and tree transducers. *South African Computer Journal*, 34:11–25, 2005.

[Ehrenfeucht and Rozenberg 1974] A. Ehrenfeucht and G. Rozenberg. *A pumping theorem for EDT0L languages*. Technical Report CU-CS-047-74, University of Colorado, 1974.

[Ehrenfeucht and Rozenberg 1976] A. Ehrenfeucht and G. Rozenberg. On proving that certain languages are not ET0L. *Acta Informatica*, 6:407–415, 1976.

[Ewert and van der Walt 2002] S. Ewert and A. van der Walt. A pumping lemma for random permitting context languages. *Theoretical Computer Science*, 270(1-2):959–967, 2002.

[Ewert 1999] S. Ewert. *Random Context Picture Grammars*. PhD thesis, University of Stellenbosch, 1999.

[Hewett and Slutzki 1988] R. Hewett and G. Slutzki. Comparisons between some pumping conditions for context-free languages. *Theory of Computing Systems*, 21:223–233, 1988.

[Kløve 1977] T. Kløve. Pumping languages. *International Journal of Computer Mathematics*, 6(2):115–125, 1977.

[Lindenmayer and Prusinkiewicz 1990] A. Lindenmayer and P. Prusinkiewicz. *The Algorithmic Beauty of Plants*. Springer-Verlag, New York, 1990.

[Ogden 1968] W. Ogden. A helpful result for proving inherent ambiguity. *Mathematical Systems Theory*, 2:191–194, 1968.

[Parikh 1966] R. J. Parikh. On context-free languages. *Journal of the Association for Computing Machinery*, 13(4):570–581, 1966.

[Păun 1979a] G. Păun. On the family of finite index matrix languages. *Journal of Computer and System Sciences*, 18:267–280, 1979.

[Păun 1979b] G. Păun. Some further remarks on the family of finite index matrix languages. *RAIRO Informatique Théorique*, 13(3):289–297, 1979.

[Rabkin 2012] M. Rabkin. Ogden's lemma for ET0L languages. In A.-H. Dediu and C. Martín-Vide, editors, *LATA 2012*, volume 7183 of *Lecture Notes in Computer Science*, pages 460–469. Springer, 2012.

[Ramos-Jiménez *et al.* 1998] G. Ramos-Jiménez, J. López-Muñoz, and R. Morales-Bueno. Comparisons of Parikh's condition to other conditions for context-free languages. *Theoretical Computer Science*, 202(1-2):231–244, 1998.

[Rozenberg and Salomaa 1980] G. Rozenberg and A. Salomaa. *The Mathematical Theory of L Systems*. Pure and Applied Mathematics. Academic Press, 1980.

[Rozenberg 1976] G. Rozenberg. More on ET0L systems versus random context grammars. *Information Processing Letters*, 5(4):102–106, 1976.

[Salomaa 1969] A. Salomaa. On grammars with restricted use of productions. *Annales Academiæ Scientiarum Fennicæ. Series A 1*, page 451, 1969.

[Sipser 2006] M. Sipser. *Introduction to the Theory of Computation*. Thomson Course Technology, 2nd edition, 2006.

[Sokolowski 1978] S. Sokolowski. A method for proving programming languages non context-free. *Information Processing Letters*, 7(3):151–153, 1978.

[van der Walt and Ewert 2000] A. van der Walt and S. Ewert. A shrinking lemma for random forbidding context languages. *Theoretical Computer Science*, 237(1-2):149–158, 2000.

[van der Walt and Ewert 2003] A. van der Walt and S. Ewert. A property of random context picture grammars. *Theoretical Computer Science*, 301(1-3):313–320, 2003.

[van der Walt 1972] A. van der Walt. Random context languages. In C. V. Freiman, J. E. Griffith, and J. L. Rosenfeld, editors, *Information Processing 71, Proceedings of the IFIP Congress 71*, volume 1 – Foundations and Systems, pages 66–68. North-Holland, 1972.

[Zetzsche 2010] G. Zetzsche. On erasing productions in random context grammars. In *ICALP (2) '10*, pages 175–186, 2010.