A New Pumping Lemma for Indexed Languages, with an Application to Infinite Words[☆]

Tim Smith

Northeastern University
Boston, MA, USA

Abstract

We prove a pumping lemma in order to characterize the infinite words determined by indexed languages. An infinite language L determines an infinite word α if every string in L is a prefix of α. If L is regular or context-free, it is known that α must be ultimately periodic. We show that if L is an indexed language, then α is a morphic word, i.e., α can be generated by iterating a morphism under a coding. Since the other direction, that every morphic word is determined by some indexed language, also holds, this implies that the infinite words determined by indexed languages are exactly the morphic words. The pumping lemma which we use to obtain this result generalizes one recently proved for the class ET0L.

Keywords: infinite word, indexed language, prefix language, morphic word, pumping lemma

1. Introduction

Formal languages and infinite words can be related to each other in various ways. One natural connection is via the notion of a prefix language. A prefix language is a language L such that for all $x, y \in L$, x is a prefix of y or y is a prefix of x. Every infinite prefix language determines a single infinite word. Prefix languages were introduced by Book [5] in an attempt to study the complexity of infinite words in terms of acceptance and generation by automata. Book used the pumping lemma for context-free languages to show that every context-free prefix language is regular, implying that any infinite word determined by such a language is ultimately periodic. Recent work has continued and expanded Book's project, classifying the infinite words determined by various classes of automata [21] and parallel rewriting systems [20].

In this paper we characterize the infinite words determined by indexed languages. The indexed languages, introduced in 1968 by Alfred Aho [1], fall between the context-free and context-sensitive languages in the Chomsky hierarchy. More powerful than the former class and more tractable than the latter, the indexed languages have been applied to the study of natural languages [10] in computational linguistics. Indexed languages are generated by indexed grammars, in which nonterminals are augmented with stacks which can be pushed, popped, and copied to other nonterminals as the derivation proceeds. Two automaton characterizations are the nested stack automata of Aho [2], and the order-2 pushdown automata within the Maslov pushdown hierarchy [16].

The class of indexed languages IL includes all of the stack automata classes and rewriting system classes whose infinite words are characterized by Smith [21, 20]. In particular, IL properly includes ET0L [9], a broad class within the hierarchy of parallel rewriting systems known as L systems. L systems have close connections with a class of infinite words called morphic words, which are generated by repeated application of a morphism to an initial symbol, under a coding [3]. It has recently been shown that every infinite word determined by an ET0L language is morphic [20]. This raises the question of whether the indexed languages

[☆]This journal paper comprises the full version of the author's conference paper [22].

Email address: smithtim@ccs.neu.edu (Tim Smith)

too determine only morphic words, or whether indexed languages can determine infinite words which are not morphic.

To answer this question, we employ a new pumping lemma for IL. In Book's paper, as well as in those of Smith [20, 21], pumping lemmas played a prominent role in characterizing the infinite words determined by various language classes. A pumping lemma for a language class C is a powerful tool for proving that certain languages do not belong to C, and thereby for proving that certain infinite words cannot be determined by any language in C. For the indexed languages, a pumping lemma exists due to Hayashi [12], as well as a "shrinking lemma" due to Gilman [11]. We were not successful in using these lemmas to characterize the infinite words determined by IL, so instead have proved a new pumping lemma for this class (Theorem 1), which may be of independent interest.

Our lemma generalizes a pumping lemma recently proved for ET0L languages by Rabkin [18]. Roughly, it states that for any indexed language L, any sufficiently long word $w \in L$ may be written as $u_1 \cdots u_n$, each u_i may be written as $v_{i,1} \cdots v_{i,n_i}$, and the $v_{i,j}$s may be replaced with u_is to obtain new words in L. Using this lemma, we extend to IL a theorem about frequent and rare symbols proved for ET0L by Ehrenfeucht and Rozenberg [8], which can be used to show that certain languages are not indexed. We also use the lemma to obtain the new result that every infinite indexed language has an infinite subset in a smaller class of L systems called CD0L. This implies that every infinite word determined in IL can also be determined in CD0L, and thus that every such word is morphic. Since every morphic word can be determined by some CD0L language [20], we therefore obtain a complete characterization of the infinite words determined by indexed languages: they are exactly the morphic words.

1.1. Proof techniques

Our pumping lemma for IL generalizes the one proved by Rabkin [18] for ET0L. Derivations in an ET0L system, like those in an indexed grammar, can be viewed as having a tree structure, but with certain differences. In ET0L, symbols are rewritten in parallel, and the tree is organized into levels corresponding to the steps of the derivation. Further, each node in the tree has one of a finite set of possible labels, corresponding to the symbols in the ET0L system. Rabkin's proof classifies each level of the tree according to the set of symbols which appear at that level, and then finds two levels with the same symbol set, which are used to construct the pumping operation. By contrast, the derivation tree of an indexed grammar is not organized into levels in this way, and there is no bound on the number of possible labels for the nodes, since each nonterminal can have an arbitrarily large stack. We deal with these differences by assigning each node a "type" based on the set of nonterminals which appear among its descendants immediately before its stack is popped. These types then play a role analogous to the symbol sets of Rabkin [18] in our construction of the pumping operation.

1.2. Related work

The model used in this paper, in which infinite words are determined by languages of their prefixes, originates in Book's 1977 paper [5]. Book formulated the "prefix property" in order to allow languages to "approximate" infinite sequences, and showed that for certain classes of languages, if a language in the class has the prefix property, then it is regular. A follow-up by Latteux [15] gives a necessary and sufficient condition for a prefix language to be regular. Languages whose complement is a prefix language, called "coprefix languages", have also been studied; see Berstel [4] for a survey of results on infinite words whose coprefix language is context-free. Smith [20] uses prefix languages to categorize the infinite words determined by a hierarchy of L system classes. The infinite words determined by several classes of one-way stack automata are characterized by Smith [21], with partial results for multihead deterministic finite automata. The present journal paper comprises the full version of the author's conference paper [22].

Hayashi's 1973 pumping lemma for indexed languages is proved in a dense thirty-page paper [12]. The main theorem states that if a given terminal derivation tree is big enough, new terminal derivation trees can be generated by the insertion of other trees into the given one. Hayashi applies his theorem to give a new proof that the finiteness problem for indexed languages is solvable and to show that certain languages are not indexed. Gilman's 1996 "shrinking lemma" for indexed languages [11] is intended to be easier to employ,

and operates directly on terminal strings rather than on derivation trees. Our lemma generalizes the recent ET0L pumping lemma of Rabkin [18]. Like Gilman's lemma, it is stated in terms of strings rather than derivation trees, making it easier to employ, while like Hayashi's lemma and unlike Gilman's, it provides a pumping operation which yields infinitely many new strings in the language.

Another connection between indexed languages and morphic words comes from Braud and Carayol [6], in which morphic words are related to a class of graphs at level 2 of the pushdown hierarchy. The string languages at this level of the hierarchy are the indexed languages. A pumping lemma for pushdown graphs of any level in the hierarchy is due to Parys [17].

1.3. Outline of paper

The paper is organized as follows. Section 2 gives preliminary definitions and propositions. Section 3 proves our pumping lemma for indexed languages. Section 4 gives applications for the lemma, in particular characterizing the infinite words determined by indexed languages. Section 5 gives our conclusions.

2. Preliminaries

An **alphabet** A is a finite set of symbols. A **word** is a concatenation of symbols from A. We denote the set of finite words by A^* and the set of infinite words by A^ω. A **language** is a subset of A^*. A **string** x is a finite word. The length of x is denoted by $|x|$. We denote the empty string by λ. For a symbol c, $\#_c(x)$ denotes the number of appearances of c in x, and for an alphabet B, $\#_B(x)$ denotes $\sum_{c\in B} \#_c(x)$. A **prefix** of x is a string s such that $x = sx'$ for some string x'. A **prefix** of an infinite word α is a string s such that $\alpha = s\alpha'$ for some infinite word α'. For a string $x \neq \lambda$, x^ω denotes the infinite word $xxx\cdots$. An infinite word of the form xy^ω, where x and y are strings and $y \neq \lambda$, is called **ultimately periodic**.

2.1. Prefix languages

A **prefix language** is a language L such that for all $x, y \in L$, x is a prefix of y or y is a prefix of x. A language L **determines** an infinite word α iff L is infinite and every $x \in L$ is a prefix of α. For example, the infinite prefix language $\{\lambda, \mathtt{ab}, \mathtt{abab}, \mathtt{ababab}, \dots\}$ determines the infinite word $(\mathtt{ab})^\omega$. For a language class C, let $\omega(C) = \{\alpha \mid \alpha \text{ is an infinite word determined by some } L \in C\}$. The following propositions are basic consequences of the definitions.

Remark 1. A language determines at most one infinite word.

Remark 2. A language L determines an infinite word iff L is an infinite prefix language.

Remark 3. If a language L determines an infinite word α and L' is an infinite subset of L, then L' determines α.

2.2. Morphic words

A **morphism** on an alphabet A is a map h from A^* to A^* such that for all $x, y \in A^*$, $h(xy) = h(x)h(y)$. Notice that $h(\lambda) = \lambda$. The morphism h is a **coding** if for all $a \in A$, $|h(a)| = 1$. A string $x \in A^*$ is **mortal** (for h) if there is an $m \geq 0$ such that $h^m(x) = \lambda$. The morphism h is **prolongable** on a symbol a if $h(a) = ax$ for some $x \in A^*$, and x is not mortal. If h is prolongable on $a, h^\omega(a)$ denotes the infinite word $a\ x\ h(x)\ h^2(x)\ \cdots$. An infinite word α is **morphic** if there is a morphism h, coding e, and symbol a such that h is prolongable on a and $\alpha = e(h^\omega(a))$. For example, let:

$$\begin{array}{ll} h(\mathtt{s}) = \mathtt{sbaa} & e(\mathtt{s}) = \mathtt{a} \\ h(\mathtt{a}) = \mathtt{aa} & e(\mathtt{a}) = \mathtt{a} \\ h(\mathtt{b}) = \mathtt{b} & e(\mathtt{b}) = \mathtt{b} \end{array}$$

Then $e(h^\omega(\mathtt{s})) = \mathtt{a}^1\mathtt{ba}^2\mathtt{ba}^4\mathtt{ba}^8\mathtt{ba}^{16}\mathtt{b}\cdots$ is a morphic word. See Allouche and Shallit [3] for more on morphic words. Morphic words have close connections with the parallel rewriting systems known as L systems. Many

classes of L systems appear in the literature; here we define only HD0L and CD0L. For more on L systems, including the class ET0L, see Kari et al.[14] and Rozenberg and Salomaa [19]. An **HD0L system** is a tuple $G = (A, h, w, g)$ where A is an alphabet, h and g are morphisms on A, and w is in A^*. The language of G is $L(G) = \{g(h^i(w)) \mid i \geq 0\}$. If g is a coding, G is a **CD0L system**. **HD0L** and **CD0L** are the sets of HD0L and CD0L languages, respectively. It is known that CD0L $\subset$ HD0L $\subset$ ET0L $\subset$ IL [14, 9]. Smith [20] shows that $\omega(\text{CD0L}) = \omega(\text{HD0L}) = \omega(\text{ET0L})$, and α is in this class of infinite words iff α is morphic.

2.3. Indexed languages

The class of indexed languages IL consists of the languages generated by indexed grammars. These grammars extend context-free grammars by giving each nonterminal its own stack of symbols, which can be pushed, popped, and copied to other nonterminals as the derivation proceeds. Indexed grammars come in several forms [1, 10, 13], all generating the same class of languages, but varying with respect to notation and which productions are allowed. The following definition follows the form of Hopcroft and Ullman [13].

An **indexed grammar** is a tuple $G = (N, T, F, P, S)$ in which N is the nonterminal alphabet, T is the terminal alphabet, F is the stack alphabet, $S \in N$ is the start symbol, and P is the set of productions of the forms

$$A \to r \qquad A \to Bf \qquad Af \to r$$

with $A, B \in N$, $f \in F$, and $r \in (N \cup T)^*$. In an expression of the form $Af_1 \cdots f_n$ with $A \in N$ and $f_1, \ldots, f_n \in F$, the string $f_1 \cdots f_n$ can be viewed as a stack joined to the nonterminal A, with f_1 denoting the top of the stack and f_n the bottom. For $r \in (N \cup T)^*$ and $x \in F^*$, we write $r\{x\}$ to denote r with every $A \in N$ replaced by Ax. For example, with $A, B \in N$ and $\mathtt{c}, \mathtt{d} \in T$, $\mathtt{c}\mathtt{d}AB\{f\} = \mathtt{c}\mathtt{d}AfBf$. For $q, r \in (NF^* \cup T)^*$, we write $q \longrightarrow r$ if there are $q_1, q_2 \in (NF^* \cup T)^*$, $A \in N$, $p \in (N \cup T)^*$, and $x, y \in F^*$ such that $q = q_1\ Ax\ q_2$, $r = q_1\ p\{y\}\ q_2$, and one of the following is true: (1) $A \to p$ is in P and $y = x$, (2) $A \to pf$ is in P and $y = fx$, or (3) $Af \to p$ is in P and $x = fy$. Let $\stackrel{*}{\longrightarrow}$ be the reflexive, transitive closure of $\longrightarrow$. For $A \in N$ and $x \in F^*$, let $L(Ax) = \{s \in T^* \mid Ax \stackrel{*}{\longrightarrow} s\}$. The language of G, denoted $L(G)$, is $L(S)$. The class IL of indexed languages is $\{L(G) \mid G \text{ is an indexed grammar}\}$. See Example 1 in Section 3.3 for a sample indexed grammar.

For convenience, we will work with a form of indexed grammar which we call "grounded", in which terminal strings are produced only at the bottom of the stack. G is **grounded** if there is a symbol $\$ \in F$ (called the bottom-of-stack symbol) such that every production has one of the forms

$$S \to A\$ \qquad A \to r \qquad A \to Bf \qquad Af \to r \qquad A\$ \to s$$

with $A, B \in N \setminus S$, $f \in F \setminus \$$, $r \in (N \setminus S)^+$, and $s \in T^*$.

Proposition 1. *For every indexed grammar G, there is a grounded indexed grammar G' such that $L(G') = L(G)$.*

Proof. Let $G = (N, T, F, P, S)$. Add a nonterminal S' to N and replace every occurrence of S in P with S'. Then add a symbol $\$$ to F and add to P the production $S \to S'\$$. Next, for every $t \in T$, add a nonterminal X_t to N and replace every occurrence of t in P with X_t. Then add a nonterminal X_λ to N and replace every production of the form $A \to \lambda$ with $A \to X_\lambda$ and every production of the form $Af \to \lambda$ with $Af \to X_\lambda$. Finally, for every $s \in T \cup \{\lambda\}$, add to P the production $X_s\$ \to s$ and for every $f \in F$, add to P the production $X_s f \to X_s$. The resulting grammar G' is grounded and $L(G') = L(G)$.

3. Pumping Lemma for Indexed Languages

In this section we present our pumping lemma for indexed languages (Theorem 1) and give an example of its use. Our pumping lemma generalizes the ET0L pumping lemma of Rabkin [18]. Like that lemma, it allows positions in a word to be designated as "marked", and then provides guarantees about the marked positions during the pumping operation. We begin with definitions regarding derivation trees of indexed grammars, followed by a lemma about paths in those trees, and then the main theorem.

3.1. Derivation trees

Let $G = (N, T, F, P, S)$ be a grounded indexed grammar. A derivation tree D of a string s has the following structure. Each internal node of D has a label in NF^* (a nonterminal with a stack), and each leaf has a label in T^* (a terminal string). Each internal node has either a single leaf node as a child, or one or more internal children. The root of D is labelled by the start symbol S, and the terminal yield of D is the string s. See Figure 1 in Section 3.3 for an example of a derivation tree.

Any positions in the string s may be chosen to be "marked". If s contains marked positions, then we will take D to be marked in the following way. Mark every leaf whose label contains a marked position of s, and then mark every internal node which has a marked descendant. Call any node with more than one marked child a **branch node**.

A **path** H in D is a list of nodes $(v_0, \ldots, v_m)$ with $m \geq 0$ such that for each $1 \leq i \leq m$, v_i is a child of v_{i-1}. For convenience, we will sometimes refer to nodes in H by their indices; e.g. node i in the context of H means v_i. When we say that there is a branch node between i and j we mean that the branch node is between v_i (inclusive) and v_j (exclusive).

For a node v, $D(v)$ means the subtree of D whose root is v. We denote the terminal yield of $D(v)$ by yield(v). For nodes v_1, v_2 in D, we say that v_1 is to the left of v_2, and v_2 is to the right of v_1, if neither node is descended from the other, and if v_1 would be encountered before v_2 in a depth-first traversal of D in which edges are chosen from left to right. A node v_1 is **reachable** from a node v_2 if v_1 is identical to v_2 or descended from v_2.

We define several operations on internal nodes of D. Each such node v has the label Ax for some $A \in N$ and $x \in F^*$. Let $\sigma(v) = A$ and $\eta(v) = |x|$. $\sigma(v)$ gives the nonterminal symbol of v and $\eta(v)$ gives the height of v's stack. We say that a node v' is in the scope of v iff v' is an internal node and there is a path in D from v to v' such that for every node v'' on the path (including v'), $\eta(v'') \geq \eta(v)$. Every node in the scope of v shares v's stack, perhaps with some additional symbols on top. Let $\beta(v)$ be the set of nodes v' such that v' is in the scope of v but no child of v' is in the scope of v. The set $\beta(v)$ can be viewed as the "last" nodes in the scope of v. Notice that for all $v' \in \beta(v)$, $\eta(v') = \eta(v)$, so v' and v have the same stack. Finally, we give v a "type" $\tau(v)$ based on which nonterminal symbols appear in $\beta(v)$. Let $\tau(v)$ be a 3-tuple such that:

- $\tau(v)[1] = \{A \in N \mid \text{for all } v' \in \beta(v),\ \sigma(v') \neq A\}$
- $\tau(v)[2] = \{A \in N \mid \text{for some } v' \in \beta(v),\ \sigma(v') = A, \text{ and for all marked } v' \in \beta(v),\ \sigma(v') \neq A\}$
- $\tau(v)[3] = \{A \in N \mid \text{for some marked } v' \in \beta(v),\ \sigma(v') = A\}$

Notice that for each v, $\tau(v)$ partitions N: every $A \in N$ occurs in exactly one of $\tau(v)[1]$, $\tau(v)[2]$, and $\tau(v)[3]$. So there are $3^{|N|}$ possible values for $\tau(v)$.

3.2. Lemma about derivation trees

In this subsection we will prove a lemma about paths in derivation trees, which we will use in the next subsection to prove our pumping lemma for indexed languages. Again let $G = (N, T, F, P, S)$ be a grounded indexed grammar.

Lemma 1. *Let $H = (v_0, \ldots, v_m)$ be a path in a derivation tree D from the root to a leaf (excluding the leaf) with more than $(|N| \cdot 3^{|N|})^{|N|^2 \cdot 3^{|N|}+1}$ branch nodes. Then there are $0 \leq b_1 < t_1 < t_2 \leq b_2 \leq m$ such that*

- *$\sigma(b_1) = \sigma(t_1)$ and $\sigma(t_2) = \sigma(b_2)$,*
- *b_2 is in $\beta(b_1)$ and t_2 is in $\beta(t_1)$,*
- *$\tau(b_1) = \tau(t_1)$, and*
- *there is a branch node between b_1 and t_1 or between t_2 and b_2.*

PROOF (IDEA). To aid the reader's understanding, we first give a sketch, followed below by the full proof. If H is flat, i.e. if all of the nodes in H have the same stack, then after $|N| \cdot 3^{|N|}$ branch nodes, there will have been two nodes with the same σ and τ, with a branch node between them. Then we can set b_1 and t_1 to these two nodes and set $t_2 = b_2 = m$, since m will be in $\beta(v)$ for every node v on the path, because H is flat. If H is not flat, then consider just the "base" of H, i.e. the nodes in H with the smallest stack. These nodes are separated by "hills" in which the stack is bigger. The base of H can be viewed as a flat path with gaps corresponding to the hills. Then at most $|N| \cdot 3^{|N|}$ of the hills can contain branch nodes. We can then use an inductive argument to bound the number of branch nodes in each hill. In this argument, each hill is itself treated as a path, which is shorter than the original path H and so subject to the induction. Since node 0 and node m in H can serve as a potential b_1 and b_2 for any of the hills, each hill has fewer configurations of σ and τ to "choose from" if it is to avoid containing nodes which could serve as t_1 and t_2. Working out the details of the induction gives the bound stated in the lemma.

We now give a full proof of Lemma 1. We will prove two supporting lemmas (Lemmas 2 and 3). Let D be a derivation tree of G. We will be working with a variation of a path which we call a descent. A **descent** H in D is a list of internal nodes $(v_0, \ldots, v_m)$ with $m \geq 0$ such that v_m is in $\beta(v_0)$ and for each $i \geq 1$, v_i is a descendant (not necessarily a child) of v_{i-1}. As with paths, we will sometimes refer to nodes in H by their indices; e.g. $\sigma(i)$ in the context of H means $\sigma(v_i)$. For $0 \leq i < m$, we say there is a **split** in H between i and $i+1$ iff any of the nodes on the path in D from v_i (inclusive) to v_{i+1} (exclusive) is a branch node. If this path in D has more than one branch node, we still say that there is just one split between i and $i+1$ in H. Thus H has at most m splits.

We call H **controlled** if for all $0 < i \leq m$, i is a child (not merely a descendant) of $i-1$. Notice that any path in D from the root to a leaf (excluding the leaf) is a controlled descent. We call H **flat** if for all $0 \leq i \leq m$, $\eta(i) = \eta(0)$. We call H **limited** iff for all $0 \leq b_1 < t_1 < t_2 \leq b_2 \leq m$ such that

- $\sigma(b_1) = \sigma(t_1)$ and $\sigma(t_2) = \sigma(b_2)$,
- b_2 is in $\beta(b_1)$ and t_2 is in $\beta(t_1)$, and
- $\tau(b_1) = \tau(t_1)$,

there are no splits between b_1 and t_1 or between t_2 and b_2.

Lemma 2. *Let $H = (v_0, \ldots, v_m)$ be a limited flat descent. Then H has at most $|N| \cdot 3^{|N|}$ splits.*

PROOF. For any i in H, there are $|N|$ possible values for $\sigma(i)$ and $3^{|N|}$ possible values for $\tau(i)$. Suppose there are more than $|N| \cdot 3^{|N|}$ splits between 0 and m. Then there are b_1, t_1 such that $0 \leq b_1 < t_1 < m$, $\sigma(b_1) = \sigma(t_1)$, $\tau(b_1) = \tau(t_1)$, and there is a split between b_1 and t_1. Let $t_2 = b_2 = m$. Obviously $\sigma(t_2) = \sigma(b_2)$. By the definition of a descent, m is in $\beta(0)$. Then since H is flat, m is in $\beta(i)$ for all i in H. Hence b_2 is in $\beta(b_1)$ and t_2 is in $\beta(t_1)$. But then H is not limited, a contradiction. So H has at most $|N| \cdot 3^{|N|}$ splits.

Let R be the set of possible values of $\tau(v)$; i.e. the set of 3-tuples each of which partitions N. We have $|R| = 3^{|N|}$. For any descent $H = (v_0, \ldots, v_m)$ and $W \subseteq N \times N \times R$, H **respects** W iff for every $0 \leq i < j \leq m$ such that j is in $\beta(i)$, the triple $(\sigma(i), \sigma(j), \tau(i))$ is in W.

Lemma 3. *Take any $W \subseteq N \times N \times R$. Any limited controlled descent which respects W has at most $(|N| \cdot 3^{|N|})^{|W|+1}$ splits.*

PROOF. Let $k = |N| \cdot 3^{|N|}$, the bound from Lemma 2. For $x \in \mathbb{N}$, let $f(x) = k^{x+1} - 2k$. We will show by induction on $|W|$ that any limited controlled descent which respects W has at most $f(|W|)$ splits.

If $|W| = 0$, then no limited controlled descent respects W, so the statement holds trivially. So say $|W| \geq 1$. Suppose for induction that for every W' such that $|W'| < |W|$, any limited controlled descent which respects W' has at most $f(|W'|)$ splits.

Take any n such that there is a limited controlled descent which respects W and has exactly n splits. We will show that $n \leq f(|W|)$. Take the lowest m such that there is a limited controlled descent $H = (v_0, \ldots, v_m)$ which respects W and has exactly n splits.

Suppose $m = 0$. Then $n = 0$. Since $|N| \geq 1$, $k \geq 3$. Then since $|W| \geq 1$ and f is increasing, $f(|W|) \geq f(1) \geq 3^{1+1} - 2 \cdot 3 \geq 3$. Then $n \leq f(|W|)$ as desired. So say $m \geq 1$.

Let *base* be the list consisting of every node v_i in H for which $\eta(i) = \eta(0)$. Since H is a descent, *base* contains v_0 and v_m, and every node in *base* has the same stack. Call i, j base-adjacent iff $i < j$, i and j are in *base*, and no node between i and j is in *base*. If i, j are base-adjacent, then every node between i and j shares the stack of i and j, with one or more additional symbols on top. Call the interval from i to j a hill iff i, j are base-adjacent. Call a hill a split hill iff it contains at least one split.

Take any base-adjacent i, j such that the hill between i and j has at least as many splits as any hill between 0 and m. Let x be the number of splits between i and j. We will show $x \leq 2 + f(|W| - 1)$. Suppose $j = i + 1$. Then $x \leq 1$. Suppose $j = i + 2$. Then $x \leq 2$. So say $j > i + 2$. Let $i' = i + 1$ and $j' = j - 1$. Then $0 < i' < j' < m$, and since H is controlled, $\eta(i') = \eta(j') = 1 + \eta(0)$ and j' is in $\beta(i')$.

Suppose there are no splits between 0 and i' or between j' and m. Then there are n splits between i' and j'. Let $m' = j' - i'$. Then the limited controlled descent $(v_{i'}, \ldots, v_{j'})$ respects W and has n splits. But $m' < m$, a contradiction, since by the construction of m, there is no such m'.

So there is such a split. Now, since m is in $\beta(0)$, $(\sigma(0), \sigma(m), \tau(0))$ is in W. Let $W' = W - (\sigma(0), \sigma(m), \tau(0))$. Suppose there are i'', j'' such that $i' \leq i'' < j'' \leq j'$, j'' is in $\beta(i'')$, $\sigma(i'') = \sigma(0)$, $\sigma(j'') = \sigma(m)$, and $\tau(i'') = \tau(0)$. Let $b_1 = 0$, $t_1 = i''$, $t_2 = j''$, and $b_2 = m$. Since there is a split between 0 and i' or between j' and m, there is a split between 0 and i'' or between j'' and m, hence between b_1 and t_1 or between t_2 and b_2. But then H is not limited, a contradiction. So there are no such i'', j''. Hence the limited controlled descent $(v_{i'}, \ldots, v_{j'})$ respects W'. Since $|W'| < |W|$, by the induction hypothesis there are at most $f(|W'|)$ splits between i' and j'. Then allowing a split between i and i' and a split between j' and j, $x \leq 2 + f(|W'|)$.

Now, the list *base* is a flat descent. If *base* was not limited, then H would not be limited. So *base* is limited. Then by Lemma 2, it has at most k splits. Then there are at most k split hills in H. Recall that $k \geq 3$. Each split hill has at most x splits. Therefore

$$\begin{aligned} n &\leq kx \\ n &\leq k(2 + f(|W| - 1)) \\ n &\leq 2k + k(k^{|W|-1+1} - 2k) \\ n &\leq 2k + k^{|W|+1} - 2k^2 \\ n &\leq f(|W|), \end{aligned}$$

completing the induction. Therefore any limited controlled descent which respects W has at most $(|N| \cdot 3^{|N|})^{|W|+1}$ splits.

Now we can complete the proof of Lemma 1.

Lemma 1. *Let $H = (v_0, \ldots, v_m)$ be a path in a derivation tree D from the root to a leaf (excluding the leaf) with more than $(|N| \cdot 3^{|N|})^{|N|^2 \cdot 3^{|N|}+1}$ branch nodes. Then there are $0 \leq b_1 < t_1 < t_2 \leq b_2 \leq m$ such that*

- *$\sigma(b_1) = \sigma(t_1)$ and $\sigma(t_2) = \sigma(b_2)$,*
- *b_2 is in $\beta(b_1)$ and t_2 is in $\beta(t_1)$,*
- *$\tau(b_1) = \tau(t_1)$, and*
- *there is a branch node between b_1 and t_1 or between t_2 and b_2.*

Proof. Since H is a path from the root to a leaf, H is a controlled descent. Suppose H is limited. Let $W = N \times N \times R$. Clearly H respects W. Then by Lemma 3, H has at most $(|N| \cdot 3^{|N|})^{|W|+1} = (|N| \cdot 3^{|N|})^{|N|^2 \cdot 3^{|N|}+1}$ splits. Then since by definition, every branch node is immediately followed by a split, H has at most $(|N| \cdot 3^{|N|})^{|N|^2 \cdot 3^{|N|}+1}$ branch nodes, a contradiction. So H is not limited and the lemma holds.

3.3. Pumping lemma for indexed languages

We are now ready to prove our pumping lemma for indexed languages (Theorem 1), which generalizes the pumping lemma for ET0L languages of Rabkin [18]. As noted, this lemma allows arbitrary positions in a word to be designated as "marked", and then provides guarantees about the marked positions during the pumping operation. The only difference between our pumping operation and that of Theorem 15 of Rabkin [18] is that in the latter, there are guaranteed to be at least two marked positions in the $v_{i,j}$ of part 4, whereas in our lemma, this $v_{i,j}$ might not contain any marked positions and could even be an empty string (which nonetheless maps under ϕ to u_i, which does contain a marked position).

Theorem 1. *Let L be an indexed language. Then there is an $l \geq 0$ (which we will call a threshold for L) such that for any $w \in L$ 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. $\{(i,j) \mid 1 \leq i \leq n \text{ and } 1 \leq j \leq n_i\}$, by I);*
2. *there is a map $\phi : I \to \{1, \ldots, n\}$ 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. *for each $(i,j) \in I$, if $v_{i,j}$ contains a marked position then so does $u_{\phi(i,j)}$;*
4. *there is an $(i,j) \in I$ such that $\phi(i,j) = i$, and there is at least one marked position in u_i but outside of $v_{i,j}$.*

Proof (Idea). To aid the reader's understanding, we first give a sketch, followed below by the full proof. We take a grounded indexed grammar $G = (N, T, F, P, S)$ with language L and set the threshold l using the bound from Lemma 1 together with some properties of the productions of G. Then we take any $w \in L$ with at least l marked positions and take a derivation tree D for w. Some path in D from the root to a leaf then has enough branch nodes to give us the b_1, t_1, t_2, and b_2 from Lemma 1. We then need to construct the map ϕ and the factors u_i and $v_{i,j}$. To do this, we use the nodes in $\beta(b_1)$ and $\beta(t_1)$. The nodes in $\beta(b_1)$ will correspond to $v_{i,j}$s and those in $\beta(t_1)$ will correspond to u_is. The operation ϕ will then map each node in $\beta(b_1)$ to a node in $\beta(t_1)$ with the same σ, and which is marked if the node being mapped is marked. This is possible because $\tau(b_1) = \tau(t_1)$. The justification for this construction is that in D, between b_1 and t_1 the stack grows from x to yx for some $x, y \in F^*$, and then shrinks back to x between $\beta(t_1)$ and $\beta(b_1)$. Since $\sigma(b_1) = \sigma(t_1)$, the steps between b_1 and t_1 can be repeated, growing the stack from x to yx to yyx to $yyyx$, and so on. Then the ys can be popped back off by repeating the steps between the nodes in $\beta(t_1)$ and $\beta(b_1)$. This construction gives us parts 1, 2, and 3 of the theorem. Part 4 follows from the fact that there is a branch node between b_1 and t_1 or between t_2 and b_2. In the former case, the u_i in part 4 corresponds to the yield produced between b_1 and t_1 involving the branch node, and the $v_{i,j}$ is specially constructed as an empty factor which maps to u_i. In the latter case, since t_2 is in $\beta(t_1)$, b_2 is in $\beta(b_1)$, and $\sigma(t_2) = \sigma(b_2)$, the u_i in part 4 corresponds to t_2 and the $v_{i,j}$ corresponds to b_2.

Proof. We now give a full proof of the theorem. Let $G = (N, T, F, P, S)$ be a grounded indexed grammar such that $L(G) = L$. Let d be the highest i such that there is a production in P with i nonterminals on the righthand side. Let e be the highest i such that there is a production in P with i terminals on the righthand side. Let $z = (|N| \cdot 3^{|N|})^{|N|^2 \cdot 3^{|N|}+1}$. Let $l = ed^z + 1$.

If L is finite, then trivially the theorem holds. So say L is infinite. Let w be a word in L with at least l marked positions. Take any derivation tree D of w. Suppose no path in D from the root to a leaf has more than z branch nodes. The maximum outdegree of D is at most d. Then by Lemma 14 of Rabkin [18], D has at most d^z marked leaves. Recall that each leaf has a label in T^* (a terminal string) and marked leaves are those whose label contains a marked position of w. Each marked leaf has a label of length at most e. But then w has at most ed^z marked positions, a contradiction.

So some path $H = (v_0, \ldots, v_m)$ in D from the root to a leaf (excluding the leaf) has more than z branch nodes. Then by Lemma 1, there are $0 \leq b_1 < t_1 < t_2 \leq b_2 \leq m$ such that $\sigma(b_1) = \sigma(t_1)$, $\sigma(t_2) = \sigma(b_2)$, b_2 is in $\beta(b_1)$, t_2 is in $\beta(t_1)$, $\tau(b_1) = \tau(t_1)$, and there is a branch node between b_1 and t_1 or between t_2 and b_2.

We specify the subscripts of I by defining n and each n_i. Let $n = 4 + |\beta(t_1)|$. Let $n_1 = n_n = 1$. We will give names to the nodes in $\beta(b_1)$ and $\beta(t_1)$, as follows. Let n_2 be the number of nodes in $\beta(b_1)$ to the left of t_1, plus one. From left to right, call these nodes $N_{2,2}, N_{2,3}, \ldots, N_{2,n_2}$. Call the nodes in $\beta(t_1)$ from left to right $N_3, N_4, \ldots, N_{n-2}$. For each i from 3 to $n-2$, let n_i be the number of nodes in $\beta(b_1)$ which are reachable from N_i. From left to right, call these nodes $N_{i,1}, N_{i,2}, \ldots, N_{i,n_i}$. Let n_{n-1} be the number of nodes in $\beta(b_1)$ to the right of t_1, plus one. From left to right, call these nodes $N_{n-1,1}, N_{n-1,2}, \ldots, N_{n-1,n_{n-1}-1}$.

We now define each u_i and each $v_{i,j}$, as well as the map ϕ which will be used in the replacement operation. Let $v_{1,1}$ be the yield of D to the left of b_1, and let $\phi(1,1) = 1$. Let $v_{2,1} = \lambda$ and let $\phi(2,1) = 2$. Let $v_{n-1,n_{n-1}} = \lambda$ and let $\phi(n-1, n_{n-1}) = n-1$. Let $v_{n,1}$ be the yield of D to the right of b_1, and let $\phi(n,1) = n$. For all other i, j for which there is a node $N_{i,j}$, proceed as follows. Set $v_{i,j} = \text{yield}(N_{i,j})$. If $N_{i,j}$ is b_2, then set $\phi(i,j) = i$. (Notice that $N_i = t_2$, so $\sigma(N_i) = \sigma(N_{i,j})$.) Otherwise, if yield$(N_{i,j})$ contains a marked position, then $\sigma(N_{i,j})$ is in $\tau(b_1)[3]$. Since $\tau(b_1) = \tau(t_1)$, there is an N_k such that $\sigma(N_k) = \sigma(N_{i,j})$ and yield(N_k) contains a marked position. Set $\phi(i,j) = k$. Otherwise, yield$(N_{i,j})$ does not contain a marked position, so $\sigma(N_{i,j})$ is in $\tau(b_1)[2]$. Then there is an N_k such that $\sigma(N_k) = \sigma(N_{i,j})$. Set $\phi(i,j) = k$. Finally, for i from 1 to n, let $u_i = v_{i,1} \cdots v_{i,n_i}$.

We now have that each u_i is the yield of the nodes in some part of D. In particular, u_1 is the yield to the left of b_1, u_2 is the yield under b_1 to the left of t_1, $u_3 \cdots u_{n-2}$ is the yield under t_1, u_{n-1} is the yield under b_1 to the right of t_1, and u_n is the yield to the right of b_1. Thus $u_1 \cdots u_n$ is the yield of D, namely w. This gives us part 1 of the theorem.

To establish part 2, we will argue that the derivation tree D of s can be "pumped" to produce new derivation trees which yield strings in accordance with the replacement operation. Let $x \in F^*$ be the stack at node b_1. Since b_2 is in $\beta(b_1)$, and b_2 is a descendant of t_1, the stack at node t_1 has the form yx for some $y \in F^*$. Thus in D, the stack grows from x to yx between b_1 and t_1, remains at or above yx until t_2, and then shrinks back to x at b_2. We will construct a new derivation tree D' in which the stack grows from x to yx to yyx, then shrinks from yyx to yx to x. The construction is as follows. Initialize D' to a copy of D. Next, make a copy C of the subtree $D(b_1)$. In C, for every ancestor of any node in $\beta(b_1)$, put y on top of its stack. For each node $N_{i,j}$ in $\beta(b_1)$, proceed as follows. Notice that $N_{i,j}$ has stack x in D and hence stack yx in C, while $N_{\phi(i,j)}$ has stack yx in D. Further, $\sigma(N_{i,j}) = \sigma(N_{\phi(i,j)})$. So in C, replace the subtree $C(N_{i,j})$ with the subtree $D(N_{\phi(i,j)})$. Finally, in D', replace the subtree $D'(t_1)$ with C. The resulting derivation tree D' now obeys the rules of G and has a yield equal to the result of performing the replacement operation of part 2 on w. This procedure can be repeated to produce a new tree D'' in which the stack grows to $yyyx$, with a yield corresponding to two iterations of the replacement operation of part 2, and so on. This gives us part 2 of the theorem. Part 3 follows from the construction of the ϕ operation.

We now establish part 4 of the theorem. Recall that there is a branch node in H between b_1 and t_1 or between t_2 and b_2. If there is a branch node between b_1 and t_1, then by definition, this branch node has at least two marked children. Take any one of these marked children which is not on the path from node b_1 to node t_1. This child is either to the left of t_1 or to the right of t_1. If it is to the left of t_1, then some marked $N_{2,i}$ is reachable from it for some $2 \leq i \leq n_2$. Since $N_{2,i}$ is marked, yield$(N_{2,i})$ contains a marked position. So $v_{2,i}$ contains a marked position. Then since $\phi(2,1) = 2$ and u_2 contains $v_{2,1}$ and $v_{2,i}$, (2,1) satisfies part 4. Similarly, if the marked child is to the right of t_1, then some marked $N_{n-1,i}$ is reachable from it for some $1 \leq i \leq n_{n-1} - 1$. Since $N_{n-1,i}$ is marked, yield$(N_{n-1,i})$ contains a marked position. So $v_{n-1,i}$ contains a marked position. Then since $\phi(n-1, n_{n-1}) = n-1$ and u_{n-1} contains $v_{n-1,i}$ and $v_{n-1,n-1}$, $(n-1, n-1)$ satisfies part 4. Otherwise, if there is no branch node in H between b_1 and t_1, then there is a branch node between t_2 and b_2. We have $t_2 = N_i$ and $b_2 = N_{i,j}$ for some $3 \leq i \leq n-2$ and $1 \leq j \leq n_i$. By definition, the branch node between t_2 and b_2 has at least two marked children. Take any one of these marked children which is not on the path from node t_2 to node b_2. Some marked $N_{i,k}$ is reachable from this child where $k \neq j$. So yield$(N_{i,k})$ contains a marked position, hence $v_{i,k}$ contains a marked position. Since $N_{i,j}$ is b_2, $\phi(i,j) = i$ by construction. Then since u_i contains both $v_{i,j}$ and $v_{i,k}$, (i,j) satisifies part 4. This establishes part 4 of the theorem, which completes the proof.

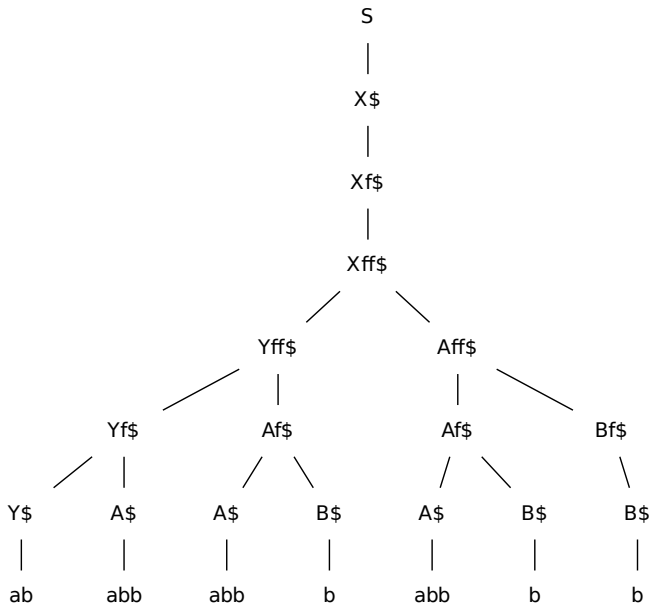

Figure 1: A derivation tree for the string $ab^1ab^2ab^3ab^4$.

Example 1. We now give an example of how our pumping operation works on a derivation tree of an indexed grammar. Let the nonterminal alphabet N be $\{S, X, Y, A, B\}$, the terminal alphabet T be $\{a, b\}$, the stack alphabet F be $\{\$, f\}$, and the set of productions P be $\{S \to X\$,\ X \to Xf,\ X \to YA,\ Yf \to YA,\ Y\$ \to ab,\ Af \to AB,\ A\$ \to abb,\ Bf \to B,\ B\$ \to b\}$. Let G be the indexed grammar (N, T, F, P, S). Notice that G is grounded and that $L(G)$ determines the infinite word $ab^1ab^2ab^3\cdots$. Figure 1 depicts a derivation tree of G with terminal yield $ab^1ab^2ab^3ab^4$. Notice how stacks are copied as the derivation proceeds; for example, the production $X \to YA$ applied to $Xff\$$ copies X's stack $ff\$$ to both Y and A, yielding $Yff\$\ Aff\$$.

Take every position in the terminal yield of the tree to be marked. Let b_1 and t_1 be the nodes labelled $X\$$ and $Xf\$$, respectively, and let t_2 and b_2 be the nodes labelled $Af\$$ and $A\$$, respectively, in the $Yff\$$ subtree. We have $\sigma(b_1) = \sigma(t_1) = X$, $\sigma(t_2) = \sigma(b_2) = A$, and $\tau(b_1) = \tau(t_1) = [\{S, X\}, \{\}, \{Y, A, B\}]$. Additionally, b_2 is in $\beta(b_1)$, which consists of all the nodes labelled $A\$$, $B\$$, or $Y\$$, and t_2 is in $\beta(t_1)$, which consists of all the nodes labelled $Af\$$, $Bf\$$, or $Yf\$$. Also, there is a branch node between t_2 and b_2, namely t_2 itself. This satisfies the conditions of Lemma 1. We break up the string as $\overline{\overline{ab}\,\overline{abb}}\ \overline{\overline{abb}\,\overline{b}}\ \overline{\overline{\overline{abb}\,\overline{b}}\ \overline{\overline{b}}}$, where the outer brackets delimit the u_is and the inner brackets delimit the $v_{i,j}$s.[1] Set $\phi(1,1) = 1$, $\phi(1,2) = 2$, $\phi(2,1) = 2$, $\phi(2,2) = 4$, $\phi(3,1) = 2$, $\phi(3,2) = 4$, and $\phi(4,1) = 4$. Applying the pumping operation yields $\overline{\overline{ab}\,\overline{abb}}\ \overline{\overline{abb}\,\overline{b}}\ \overline{\overline{abb}\,\overline{b}}\ \overline{\overline{b}}\ \overline{\overline{\overline{abb}\,\overline{b}}\ \overline{\overline{b}}}\ \overline{\overline{b}}\ \overline{\overline{b}} = ab^1ab^2ab^3ab^4ab^5$, which is indeed in $L(G)$. Applying it again yields $ab^1ab^2ab^3ab^4ab^5ab^6$, and so on.

We now follow Rabkin [18] in giving a more formal description of the replacement operation in part 2 of Theorem 1. This operation produces the words $w^{(t)}$ for all $t \geq 0$, where

$$\begin{aligned} v_{i,j}^{(0)} &= v_{i,j} \\ u_i^{(t)} &= v_{i,1}^{(t)} v_{i,2}^{(t)} \cdots v_{i,n_i}^{(t)} \\ v_{i,j}^{(t+1)} &= u_{\phi(i,j)}^{(t)} \\ w^{(t)} &= u_1^{(t)} u_2^{(t)} \cdots u_n^{(t)} \end{aligned}$$

[1] We omit the empty u_is which the proof of Theorem 1 would produce at the beginning and end of the string, since they do not affect the pumping operation in this example.

Notice that $w^{(0)} = w$. The following lemma states that the number of marked symbols tends to infinity as the replacement operation is repeatedly applied.

Lemma 4. *If L is an indexed language with threshold l, and $w \in L$ has at least l marked symbols, then for all $t \geq 0$, $w^{(t)}$ has at least t marked symbols.*

PROOF. Call each $v_{i,j}$ a v-word and call a v-word marked if it contains a marked position. We will show by induction on t that for all $t \geq 0$, $w^{(t)}$ contains at least t occurrences of marked v-words. Obviously the statement holds for $t = 0$. So say $t \geq 1$ and suppose for induction that $w^{(t-1)}$ contains at least $t-1$ occurrences of marked v-words. By part 3 of Theorem 1, for every marked $v_{i,j}$ in $w^{(t-1)}$, $v_{i,j}^{(1)}$ contains a marked position. Then $w^{(t)}$ contains at least $t-1$ occurrences of marked v-words. Now by part 4 of the theorem, there is an $(i,j) \in I$ such that $\phi(i,j) = i$ and there is at least one marked position in u_i but outside of $v_{i,j}$. Then $v_{i,j}^{(1)} = u_i$ contains at least one more occurrence of a marked v-word than $v_{i,j}$. Now since w contains $v_{i,j}$, $w^{(t-1)}$ contains $v_{i,j}$. Then $w^{(t)}$ contains at least one more occurrence of a marked v-word than $w^{(t-1)}$. So $w^{(t)}$ contains at least t occurrences of marked v-words, completing the induction. Thus for all $t \geq 0$, $w^{(t)}$ contains at least t occurrences of marked v-words, hence $w^{(t)}$ contains at least t marked positions.

4. Applications

In this section we give some applications of our pumping lemma for indexed languages (Theorem 1). We prove for IL a theorem about frequent and rare symbols which was proved by Ehrenfeucht and Rozenberg for ET0L languages [8] (see also Rabkin [18], Theorem 22). Then we characterize the infinite words determined by indexed languages.

4.1. Frequent and rare symbols

Let L be a language over an alphabet A, and $B \subseteq A$. B is **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**. B is called **rare** if for every $k \geq 1$, there is an $n_k \geq 1$ such that for all $w \in L$, if $\#_B(w) \geq n_k$ then the distance between any two appearances in w of symbols from B is at least k.

Theorem 2. *Let L be an indexed language over an alphabet A, and $B \subseteq A$. If B is rare in L, then B is nonfrequent in L.*

PROOF. Suppose B is rare and frequent in L. By Theorem 1, L has a threshold $l \geq 0$. Since B is frequent in L, there is a $w \in L$ with more than l symbols from B. If we mark all of the symbols from B in w, parts 1 to 4 of the theorem apply. By part 3 of the theorem, if $v_{i,j}$ contains a marked position then so does $u_{\phi(i,j)}$, and by part 4 of the theorem, there is an $(i,j) \in I$ such that $\phi(i,j) = i$ and there is at least one marked position in u_i but outside of $v_{i,j}$. Then $u_i^{(1)}$ contains at least two marked positions. So take any two marked positions in $u_i^{(1)}$ and let d be the distance between them. Let $k = d + 1$. Since B is rare in L, there is an $n_k \geq 1$ such that for all $w' \in L$, if $\#_B(w') \geq n_k$ then the distance between any two appearances in w' of symbols from B is at least k. By Lemma 4, $w^{(n_k)}$ contains at least n_k marked symbols, so $\#_B(w^{(n_k)}) \geq n_k$. Then the distance between any two appearances in $w^{(n_k)}$ of symbols from B is at least k. Since u_i appears in w and $u_i^{(1)}$ contains u_i, $u_i^{(1)}$ appears in $w^{(t)}$ for all $t \geq 1$. Then $u_i^{(1)}$ appears in $w^{(n_k)}$ and contains two symbols from B separated by $d < k$, a contradiction. So if B is rare in L, then B is nonfrequent in L.

Theorem 2 gives us an alternative proof of the result of Hayashi [12] and Gilman [11] that the language L below is not indexed.

Corollary 1 ([12] Theorem 5.3; [11] Corollary 4). *The language*

$$L = \{(\mathtt{ab}^n)^n \mid n \geq 1\}$$

is not indexed.

PROOF. The subset $\{\mathtt{a}\}$ of $\{\mathtt{a}, \mathtt{b}\}$ is rare and frequent in L. So by Theorem 2, L is not indexed.

4.2. CD0L and morphic words

Next, we turn to characterizing the infinite words determined by indexed languages. We show that every infinite indexed language has an infinite CD0L subset, which then implies that $\omega(\text{IL})$ contains exactly the morphic words. This is a new result which we were not able to obtain using the pumping lemmas of Hayashi [12] or Gilman [11].

Theorem 3. *Let L be an infinite indexed language. Then L has an infinite CD0L subset.*

Proof. By Theorem 1, L has a threshold $l \geq 0$. Take any $w \in L$ such that $|w| \geq l$, and mark every position in w. Then parts 1 to 4 of the theorem apply. Now for each $(i, j) \in I$, create a new symbol $x_{i,j}$. Let X be the set of these symbols. For i from 1 to n, let $x_i = x_{i,1}x_{i,2}\cdots x_{i,n_i}$. Let $x = x_1x_2\cdots x_n$. Let h be a morphism such that $h(x_{i,j}) = x_{\phi(i,j)}$ for all $(i, j) \in I$. Let g be a morphism such that $g(x_{i,j}) = v_{i,j}$ for all $(i, j) \in I$. Let A be the alphabet of L. Let G be the HD0L system $(X \cup A, h, x, g)$. Then for all $t \geq 0$, $g(h^t(x)) = w^{(t)}$. By Lemma 4, for all $t \geq 0, |w^{(t)}| \geq t$. Then $L(G)$ is an infinite HD0L subset of L. By Theorem 18 of Smith [20], every infinite HD0L language has an infinite CD0L subset. Therefore L has an infinite CD0L subset.

Theorem 4. *ω(IL) contains exactly the morphic words.*

Proof. For any infinite word $\alpha \in \omega(\text{IL})$, some $L \in \text{IL}$ determines α. Then L is an infinite indexed language, so by Theorem 3, L has an infinite CD0L subset L'. Then L' determines α, so α is in $\omega(\text{CD0L})$. Then by Theorem 23 of Smith [20], α is morphic. For the other direction, by Theorem 23 of Smith [20], every morphic word is in $\omega(\text{CD0L})$, so since $\text{CD0L} \subset \text{IL}$, every morphic word is in $\omega(\text{IL})$.

Theorem 4 lets us use existing results about morphic words to show that certain languages are not indexed, as the following example shows.

Corollary 2. *Let $L = \{$`0`, `0:1`, `0:1:01`, `0:1:01:11`, `0:1:01:11:001`, $\ldots\}$, the language containing for each $n \geq 0$ a word with the natural numbers up to n written in backwards binary and colon-separated. Then L is not indexed.*

Proof. L determines the infinite word $\alpha =$ `0:1:01:11:001:101:011:111:`$\cdots$. By Theorem 3 of Culik and Karhumäki [7], α is not morphic. Then by our Theorem 4, α is not in $\omega(\text{IL})$, so no language in IL determines α, hence L is not indexed.

5. Conclusion

In this paper we have characterized the infinite words determined by indexed languages, showing that they are exactly the morphic words. In doing so, we proved a new pumping lemma for the indexed languages, which may be of independent interest and which we hope will have further applications. One direction for future work is to look for more connections between formal languages and infinite words via the notion of prefix languages. It would be interesting to see what other language classes determine the morphic words, and what language classes are required to determine infinite words that are not morphic. More generally, for any language class, we can ask what class of infinite words it determines, and for any infinite word, we can ask in what language classes it can be determined, yielding many opportunities for future research. It is hoped that work in this area will help to build up a theory of the complexity of infinite words with respect to what language classes can determine them.

Acknowledgments.

I want to thank my advisor, Rajmohan Rajaraman, for supporting this work, encouraging me, and offering many helpful comments and suggestions. I would also like to thank the anonymous reviewers for their valuable input.

References

[1] Aho, A. V. (1968). Indexed grammars—an extension of context-free grammars. *J. ACM*, *15*, 647–671. doi:10.1145/321479.321488.

[2] Aho, A. V. (1969). Nested stack automata. *J. ACM*, *16*, 383–406. doi:10.1145/321526.321529.

[3] Allouche, J.-P., & Shallit, J. (2003). *Automatic Sequences: Theory, Applications, Generalizations*. New York, NY, USA: Cambridge University Press.

[4] Berstel, J. (1989). Properties of infinite words : Recent results. In *STACS 89* (pp. 36–46). Springer Berlin Heidelberg volume 349 of *Lecture Notes in Computer Science*. doi:10.1007/BFb0028971.

[5] Book, R. V. (1977). On languages with a certain prefix property. *Mathematical Systems Theory*, *10*, 229–237.

[6] Braud, L., & Carayol, A. (2010). Linear orders in the pushdown hierarchy. In *ICALP 2010 Part II* (pp. 88–99). Springer Berlin Heidelberg volume 6199 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-642-14162-1_8.

[7] Culik, K., & Karhumäki, J. (1994). Iterative devices generating infinite words. *Int. J. Found. Comput. Sci.*, *5*, 69–97.

[8] Ehrenfeucht, A., & Rozenberg, G. (1976). On proving that certain languages are not ETOL. *Acta Informatica*, *6*, 407–415. doi:10.1007/BF00268142.

[9] Ehrenfeucht, A., Rozenberg, G., & Skyum, S. (1976). A relationship between ET0L and EDT0L languages. *Theoretical Computer Science*, *1*, 325–330.

[10] Gazdar, G. (1988). Applicability of indexed grammars to natural languages. In U. Reyle, & C. Rohrer (Eds.), *Natural Language Parsing and Linguistic Theories* (pp. 69–94). Springer Netherlands volume 35 of *Studies in Linguistics and Philosophy*. doi:10.1007/978-94-009-1337-0_3.

[11] Gilman, R. H. (1996). A shrinking lemma for indexed languages. *Theor. Comput. Sci.*, *163*, 277–281. doi:10.1016/0304-3975(96)00244-7.

[12] Hayashi, T. (1973). On derivation trees of indexed grammars: an extension of the *uvwxy*-theorem. *Publications of The Research Institute for Mathematical Sciences*, *9*, 61–92. doi:10.2977/prims/1195192738.

[13] Hopcroft, J., & Ullman, J. (1979). *Introduction to automata theory, languages, and computation*. Addison-Wesley series in computer science. Addison-Wesley.

[14] Kari, L., Rozenberg, G., & Salomaa, A. (1997). L systems. In G. Rozenberg, & A. Salomaa (Eds.), *Handbook of Formal Languages, Vol. 1* (pp. 253–328). New York, NY, USA: Springer-Verlag New York, Inc.

[15] Latteux, M. (1978). Une note sur la propriété de prefixe. *Mathematical Systems Theory*, *11*, 235–238.

[16] Maslov, A. N. (1976). Multilevel stack automata. *Problems of Information Transmission*, *12*, 38–43.

[17] Parys, P. (2012). A Pumping Lemma for Pushdown Graphs of Any Level. In C. Dürr, & T. Wilke (Eds.), *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012)* (pp. 54–65). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik volume 14 of *Leibniz International Proceedings in Informatics (LIPIcs)*. doi:http://dx.doi.org/10.4230/LIPIcs.STACS.2012.54.

[18] Rabkin, M. (2012). Ogden's lemma for ET0L languages. In *Proceedings of the 6th International Conference on Language and Automata Theory and Applications* LATA'12 (pp. 458–467). Berlin, Heidelberg: Springer-Verlag. doi:10.1007/978-3-642-28332-1_39.

[19] Rozenberg, G., & Salomaa, A. (1980). *Mathematical Theory of L Systems*. Orlando, FL, USA: Academic Press, Inc.

[20] Smith, T. (2013). On infinite words determined by L systems. In J. Karhumäki, A. Lepistö, & L. Zamboni (Eds.), *Combinatorics on Words* (pp. 238–249). Springer Berlin Heidelberg volume 8079 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-642-40579-2_25.

[21] Smith, T. (2013). On Infinite Words Determined by Stack Automata. In *FSTTCS 2013* (pp. 413–424). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik volume 24 of *Leibniz International Proceedings in Informatics (LIPIcs)*. doi:10.4230/LIPIcs.FSTTCS.2013.413.

[22] Smith, T. (2014). On infinite words determined by indexed languages. In E. Csuhaj-Varjú, M. Dietzfelbinger, & Z. Ésik (Eds.), *Mathematical Foundations of Computer Science 2014* (pp. 511–522). Springer Berlin Heidelberg volume 8634 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-662-44522-8_43.