March 1989

A Pumping Lemma Scheme for the Control Language Hierarchy

Michael A. Palis
University of Pennsylvania

Sunil M. Shende
University of Pennsylvania

Recommended Citation

Michael A. Palis and Sunil M. Shende, "A Pumping Lemma Scheme for the Control Language Hierarchy", . March 1989.

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-89-24.

A Pumping Lemma Scheme for the Control Language Hierarchy

Abstract

In [9] Weir introduced control grammars as a model for describing the syntactic structure of natural languages. Informally, a control grammar is a pair {*G, C*} where *G* is a context-free grammar whose productions are assigned labels from a finite set of labels II, and *C* (called the control set) is a set of strings over II. A derivation in a control grammar is similar to that in an ordinary context-free grammar except that the control set *C* is used to further constrain the set of valid derivations. In particular, if one views a derivation as a tree, then (in a manner to be described later) each edge in such a tree is given a label from II according to the production of *G* associated with the edge. The derivation tree is considered "valid" if certain paths in the tree correspond to strings which are in the control set *C*. The language generated by the control grammar is then the set of strings having at least one derivation tree in the sense just described.

Keywords

context-free grammars, control grammars, pumping lemma, language hierarchies

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-89-24.

A PUMPING LEMMA SCHEME FOR THE CONTROL LANGUAGE HIERARCHY

Michael A. Palis
and Sunil M. Shende

MS-CIS-89-24
LINC LAB 148

Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104

April 1989

Acknowledgements: This research was supported in part by DARPA grant NOOO14-85-K-0018, NSF grants MCS-8219196-CER, IRI84-10413-AO2, MCS-82-07294, DCR-84-10413, MCS-83-05221 and U.S. Army grants DAA29-84-K-0061, DAA29-84-9-0027.

A Pumping Lemma Scheme for the Control Language Hierarchy

Michael. A. Palis Sunil M. Shende*

22 March 1989

Keywords: Context-free grammars, Control grammars, Pumping Lemma, Language hierarchies.

1 Introduction

In [9] Weir introduced control grammars as a model for describing the syntactic structure of natural languages. Informally, a control grammar is a pair $\{G, C\}$ where G is a context-free grammar whose productions are assigned labels from a finite set of labels Π, and C (called the control set) is a set of strings over Π. A derivation in a control grammar is similar to that in an ordinary context-free grammar except that the control set C is used to further constrain the set of valid derivations. In particular, if one views a derivation as a tree, then (in a manner to be described later) each edge in such a tree is given a label from Π according to the production of G associated with the edge. The derivation tree is considered "valid" if certain paths in the tree correspond to strings which are in the control set C. The language generated by the control grammar is then the set of strings having at least one derivation tree in the sense just described.

Weir defined a hierarchy of language classes called the *Control Language Hierarchy* (**CLH**) as follows: (1) the first class consists of all languages generated by control grammars whose control sets are context-free languages; (2) the k-th class consists of all languages generated by control grammars whose control sets are members of the $(k-1)$-st class. This hierarchy has interesting properties. For instance, Weir has shown that every class in the hierarchy is a full **AFL** [2] and contains only semilinear sets (hence, all members are included among the context-sensitive languages). These classes can also be characterized in terms of automata which are interesting generalizations of nondeterministic pushdown automata [8,9]. Moreover, the first language class in the hierarchy has been shown to be equivalent to the class of languages generated by tree adjoining grammars, a tree rewriting system for natural language recognition studied by Joshi and others [3,8,9]. Finally, it was shown in [5] that every language class in **CLH** is polynomial-time recognizable and that the entire hierarchy is contained in **LOGCFL**, the class of languages log-space reducible to context-free languages [6].

An open problem posed by Weir is whether **CLH** is a strictly separable hierarchy, i.e., whether the k-th language class is strictly contained in the $(k+1)$-st class, for all k. In this paper we resolve this

*Both authors are presently with the Department of Computer & Information Sciences, University of Pennsylvania, Philadelphia, PA 19104. This work was funded in part by ARO grant DAA29-84-9-0027, NSF grants MCS-8219116-CER, MCS-82-07294, DCR-84-10413, MCS-83-05221, and DARPA grant N00014-85-K-0018.

question in the affirmative by proving a pumping lemma for every language class in the hierarchy. As a corollary, we show that the language $L_k = \{{a_1}^n \ldots {a_{2^{(k+1)}+1}}^n \mid n \geq 1\}$ for $k \geq 1$ is not in the k-th language class but is contained in the $(k+1)$-st language class. Our result generalizes the well-known pumping lemma for context-free languages [1,2], as well as the pumping lemma for a restricted hierarchy of language classes due to Khabbaz [4].

2 The Control Language Hierarchy

The Control Language Hierarchy, first introduced in [9], was motivated by the observation that one could obtain non-context-free languages by restricting the derivations of context-free languages [2,7,8].

Definition 2.1 A *Control Grammar* (henceforth **CG**) $\mathcal{G}$, is a pair $\{G, C\}$, where $G = (V, \Sigma, \Pi, Z, P, Label)$ and $C \subseteq \Pi^+$. The first component, G, of the control grammar is, by itself, called a Labeled, Distinguished Context-free grammar (or LDCFG) in [9]. V and Σ are, respectively, finite sets of *nonterminals* and *terminals* of the LDCFG G, with $Z \in V$ the *start symbol* of G. The set of *grammar symbols*, $V \cup \Sigma$, is denoted by V_Σ. P is a finite set of *distinguished* productions of the form $(X \rightarrow X_1 \ldots X_n,\ i)$, where $X \rightarrow X_1 \ldots X_n$ can be viewed as a standard context-free production with $X \in V$ and the right-hand side $X_1 \ldots X_n$ belongs to ${V_\Sigma}^*$. In addition, i is an integer (with $1 \leq i \leq n$) that identifies exactly one symbol X_i on the right-hand side as being *distinguished.* Π is a finite set of *production labels* and *Label* is a one-to-one function from P to Π, which assigns a unique label to every production. For the sake of clarity, we will write a distinguished production $p = (X \rightarrow X_1 \ldots X_n,\ i)$ with $Label(p) = l$ as

$$p = l : X \rightarrow X_1 \ldots \check{X}_i \ldots X_n$$

The set $C \subseteq \Pi^+$ is called the *control set* of the grammar $\mathcal{G}$; each string in Π^+ is referred to as a *control string.* We say that grammar G is *controlled* by control set C. Note that by definition, C does not include the *empty string* ϵ.

It may be observed that our definition generalizes a related formalism described by Khabbaz in [4]. The two definitions coincide when the underlying context-free grammar for G is restricted. [1]

Derivations and derivation trees of control grammars are very similar to those of standard context-free grammars, and are defined inductively as follows. $A \overset{0}{\underset{G}{\Longrightarrow}} A$ is a derivation in 0 steps of G, for every nonterminal $A \in V_N$. The set of derivation trees corresponding to this derivation is the singleton consisting of a tree with a single node labeled A, and is denoted by $TreeSet(A \overset{0}{\underset{G}{\Longrightarrow}} A)$.

Inductively, let $Y \overset{k}{\underset{G}{\Longrightarrow}} \alpha X \beta$ denote a derivation of G in k or fewer steps, with its associated derivation trees, $T = TreeSet(Y \overset{k}{\underset{G}{\Longrightarrow}} \alpha X\ \beta)$. Then for every production $p = l : X \rightarrow X_1 \ldots \check{X}_i \ldots X_n$ of G, we say that $Y \overset{k+1}{\underset{G}{\Longrightarrow}} \alpha X_1 \ldots X_i \ldots X_n \beta$. For every tree $\Delta \in T$, let δ be the the leaf node labeled X which corresponds to the instance of X used on the left-hand

[1] In Khabbaz's definition, for every labeled production $p = l : X \rightarrow X_1 \ldots \check{X}_i \ldots X_n$, the *underlying* context-free production $X \rightarrow X_1 \ldots X_i \ldots X_n$ is a *linear* production; furthermore, either $X_1 \ldots X_i \ldots X_n \in \Sigma^*$, or $X_1 \ldots X_{i-1} X_{i+1} \ldots X_n \in \Sigma^*$ and $X_i \in V$.

side of the production. Let Δ_1 be the tree obtained from Δ by adding new leaf nodes labeled $X_1, \ldots, X_i, \ldots, X_n$, and new undirected edges from δ to all the new leaf nodes *except* the one labeled X_i. For this node, we add an directed edge to it from δ, and label the edge with the production label l. All such trees Δ_1 are included in $TreeSet(Y \overset{k+1}{\underset{G}{\Longrightarrow}} \alpha X_1 \ldots X_i \ldots X_n\ \beta)$.

Following standard terminology, we say that $A \overset{*}{\underset{G}{\Longrightarrow}} \alpha$, if $A \overset{k}{\underset{G}{\Longrightarrow}} \alpha$ for some finite $k \geq 0$. Likewise, $TreeSet(A \overset{*}{\underset{G}{\Longrightarrow}} \alpha)$ is the set of all derivation trees for derivations $A \overset{*}{\underset{G}{\Longrightarrow}} \alpha$. Let Γ be any derivation tree in $TreeSet(A \overset{*}{\underset{G}{\Longrightarrow}} \alpha)$. Then α is said to be the *yield* of Γ; equivalently, Γ *yields* α.

The set of *source nodes* of Γ consists of the root node and internal nodes of Γ which are connected to their respective parent nodes by an *undirected edge*. Clearly, every source node of Γ begins a unique directed, labeled path which ends at some leaf node of Γ; we denote such a path as a *c-path*. The corresponding string of labels along a c-path, read top to bottom, defines a *control word*. We abbreviate the collection of control words of Γ by $Words(\Gamma)$. Γ is defined to be a *valid derivation tree* for a terminal string $w \in \Sigma^*$ just in case that $\Gamma \in TreeSet(Z \overset{*}{\underset{G}{\Longrightarrow}} w)$ and $Words(\Gamma) \subseteq C$.

Definition 2.2 The Control Language $L(\mathcal{G})$, generated by CG $\mathcal{G} = \{G, C\}$, with start symbol Z of G, is

$$L(\mathcal{G}) = \{w \in \Sigma^* \mid \text{there is a valid derivation tree for } w \text{ in } \mathcal{G}\}$$

Let $\mathcal{C}$ be any family of languages over a finite alphabet. We say that a language L is *controlled in family* $\mathcal{C}$ iff there is a control grammar $\mathcal{G} = \{G, C\}$ such that $L = L(\mathcal{G})$ and $C \in \mathcal{C}$.

Following [9], we define a countable hierarchy of families of languages such that the 0-th family in the hierarchy is exactly the family of context-free languages, and every language in the $(i+1)$-th family is controlled in the i-th family.

Definition 2.3 The Control Language Hierarchy (**CLH**) is defined as follows:

- $\mathbf{CLH_0} = \{L \mid L = L(G)$, where G is a standard context-free grammar $\}$; i.e. $\mathbf{CLH_0} = \mathbf{CFL}$, the family of context-free languages.
- for all $k \geq 1$,
  $\mathbf{CLH_k} = \{L \mid$ there exists a sequence of grammars $G_0,\ G_1,\ \ldots,\ G_k$ such that
  1. G_0 is a context-free grammar, and for all $1 \leq j \leq k$, G_j is an LDCFG.
  2. $C_0 = L(G_0)$,
  3. for all $1 \leq j < k$, $C_j = L(\{G_j, C_{j-1}\})$, and
  4. $L = L(\{G_k, C_{k-1}\})\}$.
- $\mathbf{CLH} = \cup_k \mathbf{CLH_k}$, for all countable $k \geq 0$.

For all $k \geq 0$, if $L \in \mathbf{CLH_k}$, then we abbreviate the first two conditions in the definition by saying that *the grammar sequence* $\mathcal{G} = G_0, G_1, \ldots, G_k$ *generates* L, or $L = L(\mathcal{G})$.

Clearly, for any k, the family $\mathbf{CLH_k}$ is contained in the family $\mathbf{CLH_{k+1}}$; in the sequel, we resolve an open problem by showing that this containment is *proper*. In [4], Khabbaz proved a pumping lemma to establish strict separation of his hierarchy. However, the schema cannot be used for our purposes, since families at any level $k \geq 1$ in the Khabbaz hierarchy are *not closed under concatenation* unlike their counterparts $\mathbf{CLH_k}$. Consequently, at each level $k \geq 1$, the family $\mathbf{CLH_k}$ *properly contains* the corresponding Khabbaz family. [2]

3 A Pumping Lemma Scheme for CLH

Let G be an arbitrary LDCFG. Productions in G of the form $l : X \rightarrow \check{\epsilon}$ and $l : X \rightarrow \check{Y}$ for nonterminals X, Y are respectively called ϵ-productions and *chain*-productions of G. The following result was obtained in [5]:

Lemma 3.1 *For any $k \geq 0$, let $L = L(\{G, C\})$ be a control language in $\mathbf{CLH_{k+1}}$, where C is in $\mathbf{CLH_k}$. Then there is a control grammar $\mathcal{H} = \{H, D\}$ such that $L = L(\mathcal{H})$, D is in $\mathbf{CLH_k}$, and the underlying grammar of LDCFG H has no ϵ - or chain-productions.*

Given a grammar $\mathcal{H} = \{H, D\}$ for L as above, we construct an equivalent grammar $\mathcal{G} = \{G, C\}$ for L. Let M_H be the deterministic finite-state automaton corresponding to LDCFG H as follows. For every grammar symbol X of H, there is a state q_X in M_H. For every production $l : X \rightarrow X_1 \ldots \check{X}_i \ldots X_n$ of H, there is a transition from state q_X to state q_{X_i} labeled l. All states of M_H corresponding to nonterminal symbols of H are *initial* states of M_H; the remaining states are designated as the *final* states. It should be easy to see that the grammar $\mathcal{G} = \{G, C\}$ with $G = H$ and $C = D \cap L(M_H)$, also generates the language $L = L(\mathcal{H})$. We shall say that the grammar $\mathcal{G}$ is a *reduced* control grammar for L. Since $\mathbf{CLH_k}$ for arbitrary k is closed under intersection with regular languages, the following result is obvious.

Lemma 3.2 *For any k and any language $L \in \mathbf{CLH_k}$, there is a grammar sequence $G_0, G_1, \ldots, G_k$ generating L with the properties that:*

- *$G_0 = (V_0, \Sigma_0, P_0, Z_0)$ is a standard context-free grammar free of ϵ - and chain-productions,*
- *for all $1 \leq i \leq k$, $G_i = (V_i, \Sigma_i, \Pi_i, Z_i, P_i, Label_i)$ is an LDCFG free of ϵ - and chain-productions,*
- *if, for all $0 \leq i \leq k$, L_i is the language generated by the grammar sequence $G_0, G_1, \ldots, G_i$, then $\{G_i, L_{i-1}\}$ is a reduced control grammar for L_i.*

The proof is obtained by a straightforward induction and is omitted here.

Following [1] (pp. 186), we introduce some auxiliary definitions which simplify the discussion. A *j-factorization* Φ of a string w is a j-tuple of strings $(u_1, \ldots, u_j)$ such that $w = u_1 \ldots u_j$, i.e. w is obtained by concatenating components of its factorization. If the length of w is n, then any integer i, $1 \leq i \leq n$, is called a *position of* w. Informally, a position i of w refers to the i^{th} symbol in string w. Hence, specifying a set of positions of w can also be described as *marking* the corresponding symbols of w; the reader should consider both these phrases as being equivalent.

[2] For example, for every language L at level $k \geq 1$ in the Khabbaz hierarchy, the language $L\dot{L}$ obtained by concatenating L to itself, is in $\mathbf{CLH_k}$ but not in the corresponding Khabbaz family.

Given a set of positions, F, of w, any j-factorization Φ of w induces a partition of F into j components F_i, $1 \leq i \leq j$, such that F_i is the subset of positions in F which belong to the substring u_i in the factorization Φ. More formally, if $\Phi = (u_1, \ldots, u_j)$, then $F/\Phi = (F_1, F_2, \ldots, F_j)$ is defined such that for all $1 \leq i \leq j$,

$$F_i = \{m \in F \mid lg(u_1 \ldots u_{i-1}) < m \leq lg(u_1 \ldots u_i)\}$$

We also define the sequence of numbers, e_i, such that for all integers $i \geq 0$, $e_i = 2^{(i+2)} + 1$. Observe that $e_{i+1} = 2e_i - 1$.

We are now ready to state our main result which is a pumping lemma scheme for the entire **CLH** hierarchy; Ogden's pumping lemma for context-free grammars [cf. [1]] turns out to be a special case of our scheme.

Theorem 3.3 (Pumping Lemma Scheme) *For any $k \geq 0$, let $L = L(\mathcal{G})$ be a language in* $\mathbf{CLH_k}$ *generated by the grammar sequence $\mathcal{G} = G_0, G_1, \ldots, G_k$ satisfying the conditions of Lemma 3.2.*

Then there is a constant $n(\mathcal{G})$ such that for each $w \in L$, and any set of positions F in w, if $\mid F \mid \geq n(\mathcal{G})$ then there is an e_k–factorization $\Phi = (v_1, v_2, \ldots, v_{e_k})$ of w such that

1. *at least one triple $(F_{2j-1}, F_{2j}, F_{2j+1})$ of positions among all $1 \leq j \leq e_{k-1} - 1$ has the property that $F_{2j-1}, F_{2j}, F_{2j+1}$ are all non-empty.*
2. *$\mid F_2 \cup F_3 \cup \ldots \cup F_{e_k - 1} \mid \leq n(\mathcal{G})$, and*
3. *for all $m \geq 0$, the string $w^{[m]}$ with factorization*

$$\Phi^{[m]} = (u_1, u_2, \ldots, u_{e_k})$$

also belongs to L, where the strings u_i, $1 \leq i \leq e_k$ are defined by $u_i = v_i$ if i is odd, and $u_i = {v_i}^m$ otherwise.

Proof (of Theorem 3.3): It should be clear that Theorem 3.3 for $k = 0$ is simply a restatement of Ogden's Lemma for CFLs. We use this as the basis for our inductive proof. To avoid confusing the reader, we shall describe the proof in detail for the case when $k = 1$. Extensions to higher levels in the hierarchy are then obtained by changing some of the constants appropriately in our proof.

At the outset, we make some simple observations which will be used to complete the proof.

Claim 3.1 Let G be an arbitrary LDCFG with N distinct nonterminal symbols. Then given any $m \geq 1$, and a sequence of nonterminals $(X_1, X_2, \ldots, X_{m(N+1)})$ of G, there exist m *distinct pairs* of integers (p_i, q_i), $1 \leq i \leq m$, such that $(i-1)(N+1) < p_i < q_i \leq i(N+1)$ and $X_{p_i} = X_{q_i}$.

The claim is easily proved by applying the pigeonhole principle m times to contiguous segments of length $(N+1)$ of the sequence, i.e. to the segment $(X_1, \ldots, X_{N+1})$, the segment $(X_{N+2}, \ldots, X_{2(N+1)})$ etc.

Now suppose $\{G, C\}$ is a reduced control grammar with Z as the start nonterminal of LDCFG G. Let Γ be a derivation tree in $TreeSet(Z \overset{*}{\underset{G}{\Longrightarrow}} w)$ for some *terminal* string w as shown in Figure 1. Then a tree such as Γ_0 shown in Figure 1, is called a *recursive subtree* of Γ if and only if both its root node and the unique internal node of Γ (denoted as the *foot node* of Γ_0) which is at the frontier of Γ_0, are both labeled by the same nonterminal symbol A; in addition, both are also required to be *source nodes* in Γ. As shown in the figure, the recursive subtree Γ_0 induces a 5–factorization $\Phi = (u, v, x, y, z)$ of the string w. Note that the string vy is non-empty by Lemma 3.1.

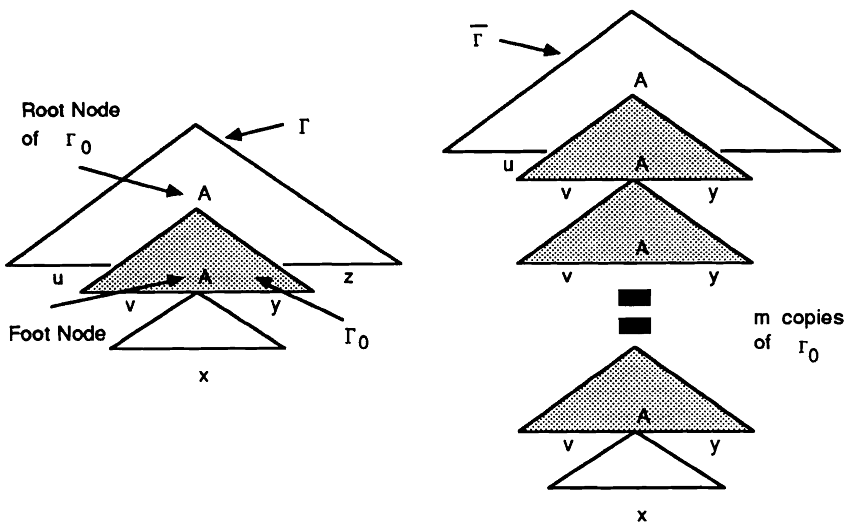

Figure 1: A Recursive Subtree of Γ

Claim 3.2 Let Γ_0 be a recursive subtree of a valid derivation tree Γ for $w \in L(\{G, C\})$, i.e. $Words(\Gamma) \subseteq C$. Then the tree $\bar{\Gamma}$ obtained from Γ by replacing Γ_0 by a stack of $m \geq 0$ *identical copies* of Γ_0 (see Figure 1), is also a valid derivation tree for a string in $L(\{G, C\})$. In particular, if Γ derives the string $w = uvxyz$ as shown then $\bar{\Gamma}$ derives the string $uv^m xy^m z$.

Proof (of Claim 3.2): Observe that since the root and the foot nodes of Γ_0 are also source nodes in Γ, every c-path of Γ either passes through nodes *entirely inside* Γ_0 or through nodes *entirely outside* Γ_0 (the c-path which begins at the foot node of Γ_0 belongs to the latter category). But $Words(\Gamma) \subseteq C$; hence, all control words which label c-paths in Γ_0 are all *in the control set* C. Therefore, replacing Γ_0 by a stack of m of its identical copies within Γ produces derivation tree $\bar{\Gamma}$ which is also valid, i.e. with $Words(\bar{\Gamma}) \subseteq C$, for the replacement simply produces copies of the c-paths already in Γ_0 without extending any of the c-paths originally in Γ. Note that if (u, v, x, y, z) is the factorization of w induced by Γ_0, then the substrings v and y on the frontier of Γ are replaced by v^m and y^m in $\bar{\Gamma}$ as shown in Figure 1..

We now proceed to prove Theorem 3.3 for the case $k = 1$. Let $\mathcal{G} = \{G_0, G_1\}$ be a control grammar for $L \in CLH_1$ as in Lemma 3.2. Let N_1 be the number of nonterminals of G_1, and let n_0 be the context-free pumping lemma constant for G_0. Then we claim that the corresponding constant $n_1 \equiv n(\mathcal{G})$ is given by $n_1 = d_1{}^{2n_0(4N_1+3)}$ where d_1 is the maximum length of right hand sides of productions in G_1.

Some preliminary definitions and observations are needed next. Let w be a string in $L(\mathcal{G})$ and let F be a set of positions of w with $| F |> n_1$. Then for any derivation tree, Δ, for w, we can define the following subsets of the set of nodes of Δ [3]. The set of $D-$nodes is the set of *ancestors* of some position in F. The set of $B-$nodes is a subset of the set of $D-$nodes with the following property. Any $B-$node has at least *two* immediate sons in Δ which are both $D-$nodes. Stated somewhat differently, any node which is on the path from the root node to some position in F belongs to the

[3]The reader may observe that we are following notation from [1], pp.187

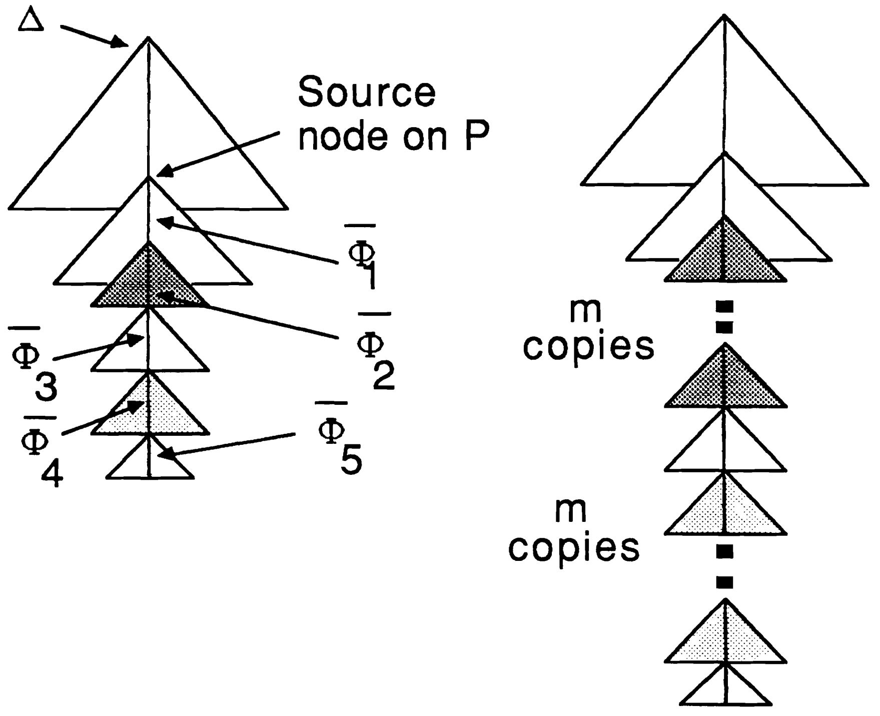

Figure 2: Pumping a c-path from a source node on P

set of $D-$nodes. The $B-$nodes are simply those which belong to at least two such paths. Given these definitions, it is easy to show ([1], pp.187) that

Claim 3.3 For every tree Δ, if w is the string of terminal symbols at the leaves of Δ, F is a set of marked positions of w, and every root-to-leaf path in Δ has at most i $B-$nodes, then the number of positions of w in F is at most ${d_1}^i$.

Recall that we chose $| F |$ to be greater than n_1. Consequently, by the contrapositive of claim 3.3, there is a path in Δ with at least $2n_0(4N_1 + 3)$ $B-$nodes; without loss of generality, let P be the path with the maximum number of $B-$nodes over all such paths. P begins at the root node of Δ and ends at some leaf node labeled, say, by the l^{th} symbol of w, denoted as w_l.

Consider, $\bar{P}$, which is the smallest contiguous part of the path P such that it contains the leaf node labeled w_l and has exactly $2n_0(4N_1 + 3)$ $B-$nodes, i.e. $\bar{P}$ starts at some $B-$node, denoted $\bar{\gamma}$ and contains the "lowest" $2n_0(4N_1 + 3)$ $B-$nodes of P (see Figure 2). We denote the set of $B-$nodes in $\bar{P}$ by $B_{\bar{P}}$, and the subtree of Δ rooted at $\bar{\gamma}$ by Γ. It is easy to see that every path in Γ has at most $2n_0(4N_1 + 3)$ $B-$nodes; hence, by claim 3.3, the number of positions on the frontier of Γ is at most n_1. Since all the strings that will be "pumped" in the rest of the proof are contained within Γ, condition (2) of Theorem 3.3 directly follows from this observation.

Let S be the set of source nodes on $\bar{P}$. For any $B-$node in S, it must be an ancestor of some position from F in w in subtree Γ, such that the position either lies to the left or to the right of the leaf node labeled w_l. We shall denote the set of $B-$nodes in S with marked descendents to the left of w_l as B_l; the set B_r is defined analogously. The reader can quickly verify that $B_l \cup B_r = B_{\bar{P}}$. Since

$B_{\bar{P}}$ contains exactly $2n_0(4N_1+3)$ nodes, either B_l or B_r must contain at least half, i.e. $n_0(4N_1+3)$, $B-$nodes. We shall make use of this principle extensively in the rest of the argument.

We now have three cases depending on the number of *source nodes* on path $\bar{P}$ (recall that source nodes begin c-paths in tree Δ); Either $\mid S \mid$ equals 0, or $\mid S \mid$ is between 1 and $(4N_1+3)$ (inclusive), or $\mid S \mid$ is at least $4(N_1+1)$.

1. $\mid S \mid = 0$: Since none of the internal nodes on $\bar{P}$ are source nodes, it follows from our definitions that $\bar{P}$ forms the "tail" of the c-path beginning at some source node on path P (see Figure 2).

   Without loss of generality, suppose that B_l contains at least half of the nodes in $B_{\bar{P}}$. For every node in B_l, we *mark* the label on the directed edge out of the node (note that every $B-$node has such a directed edge out of it which lies on $\bar{P}$). These marked labels now serve as "positions" on the control word. We denote the set of these positions by K; the size of K equals that of B_l, and is clearly greater than n_0. Hence, by Ogden's lemma (or Theorem 3.3 with $k = 0$), we can find a $5-$factorization, $\bar{\Phi}$, of the control word, such that either $K_1,\ K_2,\ K_3$ or $K_3,\ K_4,\ K_5$ are all non-empty with respect to $\bar{\Phi}$. But the factorization $\bar{\Phi}$ of the control word induces a $9-$factorization Φ of w as shown in Figure 3. Statement (1) in Theorem 3.3 is now immediate, where either $F_1,\ F_2,\ F_3$ or, respectively, $F_3,\ F_4,\ F_5$ are all non-empty with respect to Φ.

   Furthermore, by Theorem 3.2, the substrings $\bar{\Phi}_2$ and $\bar{\Phi}_4$ can be "pumped"; this corresponds to "pumping" strings $\Phi_2,\ \Phi_4,\ \Phi_6$, and Φ_8 thus proving statement (3) of the theorem. Statement (2) follows from the remark made above, i.e. from claim 3.3. Note that if we substitute the set B_r for B_l in the above discussion, then a similar argument provides the other two symmetric cases in statement (1) of the theorem.

2. $1 \leq \mid S \mid \leq (4N_1+3)$: An easy counting argument confirms that there is at least one source node on $\bar{P}$ whose corresponding c-path passes through at least $2n_0$ $B-$nodes on $\bar{P}$. Let all the $B-$nodes associated with the above c-path be denoted by set B'; define the sets B_l' and B_r' for B' analogous to B_l and B_r respectively for $B_{\bar{P}}$. Without loss of generality, we may assume that the set B_l' is at least as large as B_r'. Clearly, $B_l' \cup B_r' = B'$, and hence B_l' contains at least n_0 nodes. The reader may note that if the labels on the directed edges out of these $B-$nodes are now marked, then an argument along the same lines as the case above (i.e. $\mid S \mid = 0$) suffices to prove the theorem (see Figure 2).

   Observe that the subtree rooted at this source node is also a subtree of Γ and hence contains no more than n_1 positions in w; statement (2), therefore, follows.

3. $\mid S \mid \geq 4(N_1+1)$: If any of the c-paths associated with the source nodes in S contains a minimum of $2n_0$ $B-$nodes from $B_{\bar{P}}$, then this case reduces to the previous one. Otherwise, there must be at least $4(N_1+1)$ source nodes, such that the parts of their c-paths along $\bar{P}$ contain at least one $B-$node from $B_{\bar{P}}$. If we let $\bar{B}$ to be the set of such $B-$nodes, and define $\bar{B}_l$ and $\bar{B}_r$ analogous to B_l and B_r respectively (for $B_{\bar{P}}$), then $\bar{B} = \bar{B}_l \cup \bar{B}_r$. So, without loss of generality, let $\bar{B}_l$ be the larger set. Then it is not difficult to see that there are at least $2(N_1+1)$ source nodes on $\bar{P}$ such that the parts of their c-paths (along $\bar{P}$) contain at least one node in $\bar{B}_l$. Choose $2(N_1+1)$ such source nodes, labeled $(X_1, X_2, \ldots, X_{2(N_1+1)})$ in sequence with the property that for all $0 \leq i < 2N_1$, the source node labeled X_i is an ancestor of the one

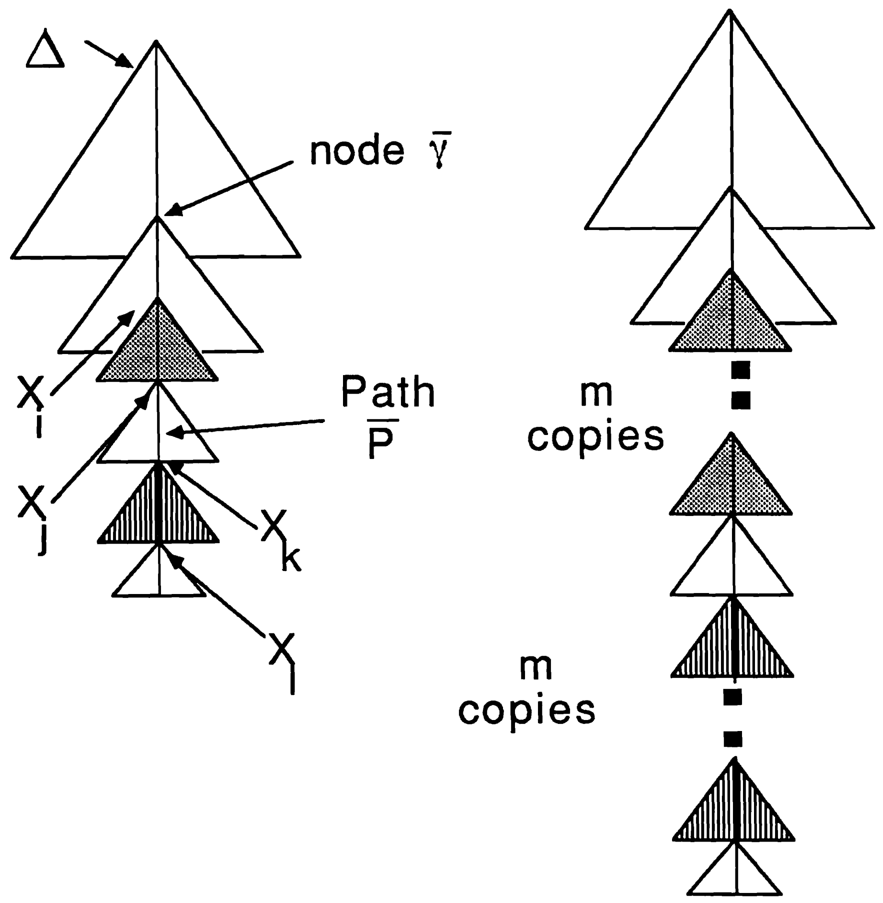

Figure 3: Pumping two recursive subtrees below $\bar{P}$

labeled X_{i+1} on path $\bar{P}$. By claim 3.1 (with $m = 2$), there must be two pairs of nodes, labeled (X_i, X_j) and (X_k, X_l), with $1 \leq i < j < k < l \leq 2(N_1+1)$, such that $X_i = X_j$ and $X_k = X_l$. These pairs respectively define two ***recursive subtrees*** of Δ (see the shaded trees in Figure 3), thereby inducing a e_1–factorization of w which satisfies F_1, F_2, F_3, F_4, F_5 all non-empty (note that, in this case, we have a stronger condition than (1) in theorem 3.3). Moreover, by claim 3.2, it is possible to "pump" both these recursive subtrees m times independently to obtain a valid derivation tree for $w^{[m]}$. Condition (2) is satisfied as before, by observing in Figure 3 that all "pumped" portions lie in the subtree Γ, which contains at most n_1 positions from F.

This concludes the proof of the theorem for the case $k = 1$. It can be easily extended to levels $k > 1$ in the following way. Let N_k be the number of nonterminals of G_k, and inductively let n_{k-1} be the iteration theorem constant for the control set of L generated by the sequence of grammars $G_0, G_1, \ldots, G_{k-1}$. Then the corresponding constant $n_k \equiv n(\mathcal{G})$ is given by $n_k = {d_k}^{2n_{k-1}(2^{k+1}[N_k+1]-1)}$ where d_k is the maximum length of right hand sides of productions in G_k. The proof then follows along exactly the same lines as above, except that we use claim 3.1

with $m = 2^k$. .

Theorem 3.3 has some nontrivial consequences. It is now possible to show that the language $\{a_1{}^n \ldots a_{2^{(k+1)}+1}{}^n \mid n \geq 1\}$ for $k \geq 1$ is not in the family $\mathbf{CLH_k}$ (the proof is obtained by generalizing the well-known proof [2] that the language $\{a^n b^n c^n \mid n \geq 1\}$ is not context-free). This language can, however, be easily shown to be in $\mathbf{CLH_{k+1}}$. It was shown in [5] that the hierarchy **CLH** is strictly contained in the complexity class **LOGCFL** [6]. By the result in this paper, we have the following:

Theorem 3.4 **CLH** *is an infinite, strictly separable hierarchy properly contained in* **LOGCFL**, *i.e.*

$$\mathbf{CFL} = \mathbf{CLH_0} \subset \mathbf{CLH_1} \subset \mathbf{CLH_2} \subset \ldots \mathbf{CLH} \subset \mathbf{LOGCFL}$$

4 Conclusion

We have exhibited a scheme of pumping lemmas for every language class in Weir's progression. This generalizes the pumping lemma for context-free languages and those for a hierarchy of language classes due to Khabbaz. As a consequence, it is shown that Weir's progression forms a strictly separable hierarchy of language classes which are contained LOGCFL (the class of languages log-space reducible to context-free languages).

References

[1] M. A. Harrison. *Introduction to Formal Language Theory.* Addison-Wesley, Reading, MA, 1978.

[2] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[3] A. K. Joshi. An introduction to tree adjoining grammars. In A. Manaster-Ramer, editor, *Mathematics of Language*, John Benjamins, Amsterdam, 1987.

[4] N. A. Khabbaz. A geometric hierarchy of languages. *J. Comput. Syst. Sci.*, 8:142–157, 1974.

[5] M. A. Palis and S. Shende. *Upper Bounds on Recognition of a Hierarchy of Non-Context-Free Languages.* Technical Report, Dept. of Comp. and Info. Sciences, University of Pennsylvania, 1988.

[6] I. H. Sudborough. On the tape complexity of deterministic context-free languages. *J. ACM*, 25(3):405–414, July 1978.

[7] J. W. Thatcher. Tree automata: An informal survey. In A. V. Aho, editor, *Currents in the Theory of Computing*, pages 143–172, Prentice Hall Inc., Englewood Cliffs, NJ, 1973.

[8] K. Vijay-Shanker. *A Study of Tree Adjoining Grammars.* PhD thesis, University of Pennsylvania, Philadelphia, Pa, 1987.

[9] D. J. Weir. *Context-Free Grammars to Tree Adjoining Grammars and Beyond.* Technical Report, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, 1987.