1977

Properties of LL-Regular Languages

David A. Poplawski

Report Number:
77-241

PROPERTIES OF LL-REGULAR LANGUAGES

David A. Poplawski
Computer Science Department
Purdue University
W. Lafayette, Ind
47907

CSD TR-241

PROPERTIES OF LL-REGULAR LANGUAGES

David R. Poplawski

August 1, 1977

ABSTRACT: The requirements for a context-free grammar to be LL-regular are relaxed, producing a larger class of grammars than the original definition. Relationships between LL-regular grammars and languages and other classes of grammars and languages are investigated, decidability questions are considered and a linear time parsing algorithm is described.

1. INTRODUCTION

Until recently, efficient deterministic top-down parsing of context-free languages was limited to the class of LL(k) languages [6,8], and some variants [1,7,9]. In addition the class of LL(f) languages [3] was introduced, but due to the arbitrary nature of the function f, efficient parsing was often impossible. Efficient top-down parsing has been achieved by restricting f to the class of functions computable by generalized sequential machines [5], thereby defining the class of LL-regular languages. This paper extends the class of grammars which define LL-regular languages and in addition provides some new results not covered in [5], including a two pass parsing algorithm similar to the parsing algorithm for LR-regular languages [2].

2. LL-REGULAR GRAMMARS AND LANGUAGES

We will use the following notation throughout this paper: A context-free grammar G will be represented by a four-tuple (N,T,P,S), where N is the finite set of non-terminal symbols, T is the finite set of terminal symbols, P is the set of productions of the form $A \rightarrow \beta$, where $A \in N$ and $\beta \in (N \cup T)^*$, and S is the start symbol. We will occasionally identify a production in P by a unique number j by writing $j: A \rightarrow \beta$.

Let $\alpha_1, \alpha_2 \in (N \cup T)^*$. We will write $\alpha_1 A \alpha_2 \Rightarrow_G \alpha_1 \beta \alpha_2$ if there is a production $(A \rightarrow \beta) \in P$ in the grammar G (omitting the subscript G when it is obvious from the context). If $\alpha_1 \in T^*$ then we write $\alpha_1 A \alpha_2 \underset{lm}{\Rightarrow} \alpha_1 \beta \alpha_2$. If $\alpha_2 \in T^*$ then we write $\alpha_1 A \alpha_2 \underset{rm}{\Rightarrow} \alpha_1 \beta \alpha_2$. $\overset{+}{\Rightarrow}$ and $\overset{+}{\underset{lm}{\Rightarrow}}$ are the transitive completions of $\Rightarrow$ and $\underset{lm}{\Rightarrow}$ respectively, $\overset{*}{\Rightarrow}$, $\overset{*}{\underset{lm}{\Rightarrow}}$ and $\overset{*}{\underset{rm}{\Rightarrow}}$ are the reflexive, transitive completions of $\Rightarrow$, $\underset{lm}{\Rightarrow}$ and $\underset{rm}{\Rightarrow}$ respectively. If $p = p_1 p_2 \ldots p_n$ is a sequence of production identifiers, then $\alpha \overset{p}{\underset{lm}{\Rightarrow}} \beta$ represents a leftmost derivation from α to β using in sequence the productions $p_1, p_2, \ldots p_n$. If n is an integer, then we write $\alpha \overset{n}{\underset{lm}{\Rightarrow}} \beta$ if there exists a sequence of productions $p = p_1 p_2 \ldots p_n$ such that $\alpha \overset{p}{\underset{lm}{\Rightarrow}} \beta$. The set $L(G) = \{x \in T^* \mid S \overset{+}{\Rightarrow} x\}$ is the language generated by the context-free grammar G.

Unless otherwise indicated, upper case letters S, A, B will be symbols in N, upper case letter X will be a symbol in $(N \cup T)$, lower case letters a, b, c, d, e will be symbols in T, lower case letters w, x, y, z will be strings of symbols in T^*, and greek letters will be strings of symbols in $(N \cup T)^*$.

If $\alpha \in (N \cup T)^*$, then α^R is the reverse of α. $FIRST_k(x) = x$ if the length of x is less than or equal to k and $FIRST_k(x) = y$ where the length of y equals k and there is a $z \in T^*$ such that $x = yz$.

Let $\pi = (R_1,\ldots,R_n)$ be a collection of regular sets over T. If the R_i are pairwise disjoint and the union of all R_i in π is equal to T*, then π is called a regular partition of T*. For strings $x,y \in T^*$, if $x \in R_i$ and $y \in R_i$ for some i, then we will write $x \equiv y \pmod{\pi}$.

A deterministic finite automata M will be represented by the five-tuple (Q,T,δ,q_0,F), where Q is the set of states, T is the input alphabet, δ is a mapping from $Q\times T \to Q$, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is the set of final states.

DEFINITION 2.1: Let G = (N,T,P,S) be a cfg. Let $\pi = (R_1,\ldots,R_n)$ be a regular partition of T*. G is said to be strong LL-regular for π if, given any two leftmost derivations:

$$S \overset{*}{\underset{lm}{\Rightarrow}} w_1 A\alpha_1 \underset{lm}{\Rightarrow} w_1\beta\alpha_1 \overset{*}{\underset{lm}{\Rightarrow}} w_1 x$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} w_2 A\alpha_2 \underset{lm}{\Rightarrow} w_2\gamma\alpha_2 \overset{*}{\underset{lm}{\Rightarrow}} w_2 y$$

such that $x \equiv y \pmod{\pi}$, then it follows that $\beta = \gamma$.

This definition is equivalent to the definition of an LL-regular grammar given in [5]. Since it is similar in form to the definition of strong LL(k) grammars [6], grammars satisfying it will be called strong LL-regular grammars.

Corresponding to the definition of LL(k) grammars [6] are the LL-regular grammars.

DEFINITION 2.2: Let G = (N,T,P,S) be a cfg. Let $\pi = (R_1,\ldots,R_n)$ be a regular partition of T*. G

is said to be LL-regular for π (written LL(π)) if, given any two leftmost derivations:

$$S \overset{*}{\underset{lm}{\Rightarrow}} \omega A\alpha \underset{lm}{\Rightarrow} \omega\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} \omega x$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} \omega A\alpha \underset{lm}{\Rightarrow} \omega\gamma\alpha \overset{*}{\underset{lm}{\Rightarrow}} \omega y$$

such that $x \equiv y \pmod{\pi}$, then it follows that $\beta = \gamma$.

DEFINITION 2.3: A language L over alphabet T is said to be LL-regular if there exists a grammar G and a regular partition π of T* such that G is LL(π) and L = L(G).

3. GRAMMAR PROPERTIES AND RELATIONSHIPS

The following set of propositions establish the relationship of the class of LL-regular grammars to other grammar classes. The class of LL-regular grammars fall between the LL(k) and the LR-regular [2] class of grammars, and are incomparable with the class of LR(k) grammars.

PROPOSITION 3.1: Every strong LL-regular grammar is LL-regular.

PROOF: Follows immediately from the definitions, since the strong LL-regular definition is a special case of the LL-regular definition.

PROPOSITION 3.2: There is an LL-regular grammar that is not strong LL-regular.

PROOF: Let G = ({S,A},{b,c,d,e},P,S), where P = {S→Ab,S→Ac,A→d,A→e}, and let π =

({db,ec},{eb,da},T*-{db,ec,eb,dc}). G is not strong LL-regular for π since the two derivations

$$S \overset{*}{\underset{lm}{\Rightarrow}} Ab \underset{lm}{\Rightarrow} db \overset{*}{\underset{lm}{\Rightarrow}} db$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} Ac \underset{lm}{\Rightarrow} ec \overset{*}{\underset{lm}{\Rightarrow}} ec$$

have $db \equiv ec \pmod{\pi}$ but $d \neq e$. Since there are only four derivations posible in G, we have

$$S \overset{*}{\underset{lm}{\Rightarrow}} Ab \underset{lm}{\Rightarrow} db \overset{*}{\underset{lm}{\Rightarrow}} db$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} Ab \underset{lm}{\Rightarrow} eb \overset{*}{\underset{lm}{\Rightarrow}} eb$$

but $db \not\equiv eb \pmod{\pi}$ and

$$S \overset{*}{\underset{lm}{\Rightarrow}} Ac \underset{lm}{\Rightarrow} dc \overset{*}{\underset{lm}{\Rightarrow}} dc$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} Ac \underset{lm}{\Rightarrow} ec \overset{*}{\underset{lm}{\Rightarrow}} ec$$

but $dc \not\equiv ec \pmod{\pi}$ and therefore G is LL-regular for π.

PROPOSITION 3.3: Let $G = (N,T,P,S)$ be a grammar that is $LL(k)$ for some integer $k \geq 1$. Then G is $LL_t(\pi)$ for $\pi = (w_1, w_2, \ldots, u_1T^*, u_2T^*, \ldots)$, where $w_1, w_2, \ldots$ are all the strings over T^* of length less than k and $u_1, u_2, \ldots$ are all the strings over T^* of length exactly k.

PROOF: Assume G is $LL(k)$. Consider the two derivations

$$S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \overset{*}{\underset{lm}{\Rightarrow}} wy$$

and suppose $x \equiv y \pmod{\pi}$. If $x,y = w_i$ then $FIRST_k(x) = FIRST_k(y)$. If $x,y \in u_iT^*$, then $FIRST_k(x) = FIRST_k(y)$ also. Since G is $LL(k)$ we conclude that $\beta = \gamma$. Therefore G is $LL(\pi)$.

PROPOSITION 3.4: The grammar $G = (N,T,P,S)$, where $N = \{S,A,B\}$, $T = \{a,b,c,d,e\}$, and $P = \{S\to Aa,\ S\to Bb,\ A\to cAd,\ A\to e,\ B\to cBd,\ B\to e\}$ is $LL(\pi)$ for the regular partition $\pi = \{cT^*a,\ eT^*a,\ cT^*b,\ eT^*b,\ T^*-(c\cup e)T^*(a\cup b)\}$, but is not $LR(k)$ for any integer k.

PROOF: That G is not $LR(k)$ for any k can be shown by considering the following derivations:

$$S \overset{*}{\underset{rm}{\Rightarrow}} c^nAd^na \Rightarrow c^ned^na$$

$$S \overset{*}{\underset{rm}{\Rightarrow}} c^nBd^nb \Rightarrow c^ned^nb \qquad \text{for } n\geq 0.$$

For any fixed integer k there is always an $n\geq k$ such that $FIRST_k(ed^na) = FIRST_k(ed^nb)$. But $c^nAd^na \neq c^nBd^na$ so G is not $LR(k)$ for any integer k.

To show that G is $LL(\pi)$ it is sufficient to show that the contrapositive of the definition is satisfied, that is, instead of showing that $x\equiv y \pmod{\pi}$ implies that $\alpha = \beta$, show that $\alpha \neq \beta$ implies that $x\not\equiv y \pmod{\pi}$. To this end, consider the following three places where the definition may be not satisfied: Case 1 - using productions $S\to Aa$ and $S\to Bb$.

$$S \overset{*}{\underset{lm}{\Rightarrow}} S \underset{lm}{\Rightarrow} Aa \overset{*}{\underset{lm}{\Rightarrow}} c^ned^na$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} S \underset{lm}{\Rightarrow} Bb \overset{*}{\underset{lm}{\Rightarrow}} c^ned^nb \qquad n\geq 0$$

In this case, $Aa \neq Bb$, and for all n, $c^ned^na \not\equiv c^ned^nb \pmod{\pi}$. Case 2 - using productions $A\to cAd$ and $A\to e$.

$$S \overset{*}{\underset{lm}{\Rightarrow}} c^mAd^ma \underset{lm}{\Rightarrow} c^mcAdd^ma \overset{*}{\underset{lm}{\Rightarrow}} c^mc^ned^nd^ma$$

$S \overset{m}{\Rightarrow} c^mBd^ma \underset{m}{\Rightarrow} c^med^ma \overset{m}{\Rightarrow} c^med^ma \quad m\geq 0, n\geq 1$

In this case, $cBd \neq e$, and for all n and m, $c^ned^mc^ma \neq ed^ma \pmod{\pi}$. Case 3 - using productions $B \rightarrow cBd$ and $B \rightarrow e$.'

$S \overset{m}{\Rightarrow} c^mBd^mb \underset{m}{\Rightarrow} c^mcBdd^mb \overset{m}{\Rightarrow} c^nc^ned^nd^mb$

$S \overset{m}{\Rightarrow} c^mBd^mb \underset{m}{\Rightarrow} c^med^mb \overset{m}{\Rightarrow} c^med^mb \quad m\geq 0, n\geq 1$

In this case, $cBd \neq e$, and for all n and m, $c^ned^mc^mb \neq ed^mb \pmod{\pi}$. So in all cases, the definition is satisfied and therefore G is LL(π).

The following theorem, due to [3], is useful in establishing two properties of LL-regular grammars.

THEOREM 3.5: If there exists a distinctive function f for a grammar G, then G is unambiguous and has no left-recursive nonterminal symbols.

COROLLARY 3.6: LL-regular grammars are unambiguous and have no left-recursive nonterminals.

PROOF: The function f, such that $f(x) = i$ if $x \in R_i$, where R_i is a member of the regular partition π, is a distinctive function.

PROPOSITION 3.7: The grammar $G = (N,T,P,S)$, where $N = \{S\}$, $T = \{a\}$, $P = \{S \rightarrow Sa, S \rightarrow a\}$ is LR(k) but not LL(π) for any regular partition π of T*.

PROOF: That G is LR(1) is obvious, and since S is left-recursive, G cannot be LL-regular.

LR-regular grammars are ones that satisfy the following definition.

DEFINITION 3.1: Let $G = (N,T,P,S)$ be a cfg. Let $\pi = (R_1,\ldots,R_n)$ be a regular partition of T^*. G is said to be LR-regular for π (LR(π)) if, given any two rightmost derivations:

$$S \overset{*}{\underset{rm}{\Rightarrow}} \alpha A w \Rightarrow \alpha\beta w$$

$$S \overset{*}{\underset{rm}{\Rightarrow}} \phi B x \Rightarrow \alpha\beta y$$

such that $w \equiv y \pmod{\pi}$, then it follows that $\alpha A y = \phi B x$ ($\alpha=\phi$, $A=B$, $y=x$).

The class of LR-regular grammars has been shown to properly include the LR(k) graammars [2]. In order to show that the class of LR-regular also includes the class of LL-regular grammers, the following conditions on a grammar will be shown to be sufficient to guarantee that it is LR-regular.

PROPOSITION 3.8: Let $G = (N,T,P,S)$ be a cfg. Let $\pi = (R_1,\ldots,R_n)$ be a regular partition of T^*. If G is unambiguous and the three derivations:

$$S \overset{*}{\Rightarrow} \alpha A w$$

$$A \overset{*}{\Rightarrow} \beta$$

$$S \overset{*}{\Rightarrow} \alpha\beta y$$

where $w \equiv y \pmod{\pi}$ imply that $S \overset{*}{\Rightarrow} \alpha A y$, then G is LR($\pi$).

PROOF: Let G satisfy the conditions above. Suppose we have the following derivations in G:

$$S \overset{*}{\underset{rm}{\Rightarrow}} \alpha Aw \Rightarrow \alpha\beta w$$

$$S \overset{*}{\underset{rm}{\Rightarrow}} \phi Bx \Rightarrow \alpha\beta y$$

and suppose $w \equiv y \pmod{\pi}$. Then rewriting, we get

$$S \overset{*}{\Rightarrow} \alpha Aw$$

$$A \Rightarrow \beta$$

$$S \overset{*}{\Rightarrow} \alpha\beta y$$

and that $w \equiv y \pmod{\pi}$. So from the second definition we get $S \overset{*}{\Rightarrow} \alpha Ay$. Since G is unambiguous,

$$S \overset{*}{\Rightarrow} \alpha Ay$$

$$S \overset{*}{\Rightarrow} \alpha\beta y$$

and $A \Rightarrow \beta$

imply that $\alpha Ay \Rightarrow \alpha\beta y$. Now we have the derivations

$$S \overset{*}{\underset{rm}{\Rightarrow}} \alpha Ay \Rightarrow \alpha\beta y$$

$$S \overset{*}{\underset{rm}{\Rightarrow}} \phi Bx \Rightarrow \alpha\beta y$$

and since G is unambiguous, there can be at most one rightmost derivation from S to $\alpha\beta y$. Therefore we have $\phi Bx = \alpha Ay$ and G is LR(π).

It can now be shown that every LL-regular grammar satisfies these conditions and is therefore LR-regular. To accomplish this we require that the regular partition π be a left congruence. It is well known that any regular partition π of T^* has a refinement which is a left congruence [4], and it is easily seen that any grammar that is LL(π_1) is LL(π_2) for any refinement π_2 of π_1. In what

follows we will assume without loss of generality that any regular partition of T^* is a left congruence.

PROPOSITION 3.9: If a grammar $G = (N,T,P,S)$ is $LL(\pi)$ for some left congruent regular partition $\pi = (R_1,\ldots,R_n)$ of T^*, then G is $LR(\pi)$.

PROOF: From corollary 3.6 we know that every LL-regular grammar is unambiguous. Now assume that for the $LL(\pi)$ grammar G there are derivations

$$S \overset{*}{\Rightarrow} uAw$$

$$A \overset{*}{\Rightarrow} v$$

$$S \overset{*}{\Rightarrow} uvw'$$

and that $w \equiv w' \pmod{\pi}$. If $w = w'$ then $S \overset{*}{\Rightarrow} uAw'$ and G satisfies the alternate definition of $LR(\pi)$. If not, let $xB\alpha$ be the last intermediate string before the leftmost derivations of uvw and uvw' diverge. Then we have

$$S \overset{*}{\underset{lm}{\Rightarrow}} xB\alpha$$

$$B \Rightarrow b \overset{*}{\Rightarrow} \gamma$$

$$\alpha \overset{*}{\Rightarrow} y$$

$$x\gamma y = uvw$$

$$B \Rightarrow b' \overset{*}{\Rightarrow} \gamma'$$

$$\alpha \overset{*}{\Rightarrow} y'$$

$$x\gamma' y' = uvw'$$

If $uv = xz$ for some $z \in T^*$, then $\gamma y \equiv \gamma' y' \pmod{\pi}$ and the $LL(\pi)$ property is violated. Conversly, if

$x=uvz$ for some $z\in T^*$, then the leftmost derivations of uvw and uvw' do not diverge until after the generation of v from A. Therefore $S \overset{*}{\Rightarrow} uAw'$, G is LR($\pi$) and the proof is complete.

4. LANGUAGE PROPERTIES AND RELATIONSHIPS

In this section the class of LL-regular languages are shown to properly include the class of LL(k) languages, to be properly included in the class of LR-regular languages, and to be incomparable with the class of deterministic languages.

PROPOSITION 4.1: The class of LL-regular languages properly includes the class of of LL(k) languages.

PROOF: Every LL(k) language is LL-regular since every LL(k) grammar is LL-regular. In [5] it is shown that the language $L = \{a^nb^n \mid n\geq 1\}\cup\{a^nc^n \mid n\geq 1\}$ is strong LL-regular but not LL(k). Since every strong LL-regular grammar is LL-regular, L is LL-regular but not LL(k); therefore the inclusion is proper.

PROPOSITION 4.2: The language $L = \{a^nb^nc \mid n\geq 1\} \cup \{a^nb^{2n}d \mid n\geq 1\}$ is LL-regular but not deterministic.

PROOF: The grammar $G = (N,T,P,S)$, where $N = \{S,A,B\}$, $T = \{a,b,c,d\}$, and $P = \{S\rightarrow Ac,\ S\rightarrow Bd,\ A\rightarrow aAb,\ A\rightarrow ab,\ B\rightarrow aBbb,\ B\rightarrow abb\}$, generates L and is LL($\pi$) for the regular partition $\pi = \{aaT^*c,\ abT^*c,\ aaT^*d,\ abT^*d,$

$T^*-(a \vee ab)T^*(c \vee d))$. To show that L is not deterministic, we will show that $L' = L/\{c,d\} = \{a^n b^n \mid n \geq 1\} \cup \{a^n b^{2n} \mid n \geq 1\}$ is not deterministic. Since L' is the quotient of L by the regular set {c,d}, L will be deterministic only if L' is [4]. It is easy to show that L' is not accepted by any deterministic PDA, therefore L' is not deterministic and neither is L.

PROPOSITION 4.3: The language $L = \{a^n b^m \mid n \geq m > 0\}$ is deterministic but not LL-regular.

PROOF: L is generated by the grammar G = ({S,A}, {a,b}, {S→aS, S→A, A→aAb, A→ab}, S) which is LR(1) and therefore deterministic. Any unambiguous grammar generating an $x \in L$ by a leftmost derivation must do so in a manner equivalent to one of two ways: generate an equal number of a's and b's followed

by extra a's or generate some a's followed by an equal number of a's and b's. In either case, for the grammar to be LL-regular the decision to switch between generating an equal number of a's and b's and generating extra a's must be made when the lookahead information consists of an arbitrary number of a's followed by b's. In the first case the decision must be made after generating exactly as many a's as there are b's, but regular lookahead cannot tell how many b's are in the lookahead. In the second case the decision must be made when there are an equal number of a's and b's in the lookahead, but regular lookahead connot detect that situation either. Therefore L is not LL-regular.

PROPOSITION 4.4: The class of LR-regular languages properly includes the class of LL-regular languages.

PROOF: Every LL-regular language is LR-regular since every LL-regular grammar is LR-regular. Since the class of LR-regular languages properly includes the class of deterministic languages [2], and since $\{a^n b^m | n \geq m > 0\}$ is a deterministic language that is not LL-regular, the inclusion is proper.

5. DECIDABILITY

This section is concerned with the problem of establishing that grammars have the LL-regular property. It will be shown to be decidable whether a grammar is LL(π) for a specific regular partition π, although it is not decidable whether there exists a regular partition π for which the grammar is LL(π).

The next proposition establishes a procedure for deciding whether a grammar is LL-regular or not. Without loss of generality, we will assume that the grammar is reduced.

PROPOSITION 5.1: Let $G = (N,T,P,S)$ be a cfg. Let $\pi = (R_1,\ldots,R_n)$ be a regular partition of T^*. Let $L_{ij} = \{\alpha \mid S \overset{*}{\underset{lm}{\Rightarrow}} \omega A\alpha \underset{lm}{\Rightarrow} \omega\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} \omega x,\ x\in R_i,\ j:A \to \beta \text{ in } P\}$. G is LL($\pi$) if and only if $L_{ij} \cap L_{ik} = \Phi$ for all R_i in π and productions $j:A \to \beta$ and $k:A \to \gamma$ in P such that $\beta \neq \gamma$.

PROOF: (if) Consider the derivations

$$S \overset{*}{\underset{lm}{\Rightarrow}} \omega A\alpha \underset{lm}{\Rightarrow} \omega\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} \omega x \quad (1)$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} \omega A\alpha \underset{lm}{\Rightarrow} \omega\gamma\alpha \overset{*}{\underset{lm}{\Rightarrow}} \omega y \quad (2)$$

where $j:A \to \beta$, $k:A \to \gamma$, $x \equiv y \pmod{\pi}$ (assume $x,y \in R_i$). Derivation (1) implies that $\alpha \in L_{ij}$, while derivation (2) implies that $\alpha \in L_{ik}$. Since $L_{ij} \cap L_{ik} = \Phi$ when $\beta \neq \gamma$, we must have $\beta = \gamma$. Therefore G is LL(π).

(only if) Assume $L_{ij} \cap L_{ik} \neq \Phi$. Let α be any

element of $L_{ij} \cap L_{ik}$ $(j \neq k)$. Since $\alpha \in L_{ij}$, there must be a derivation using $j:A \rightarrow \beta$

$$S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx,$$

where $x \in R_i$. The similar derivation

$$S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \overset{*}{\underset{lm}{\Rightarrow}} wy$$

using $k:A \rightarrow \gamma$ must also exist, with $\alpha \in L_{ik}$ assuring that $y \in R_i$. Consequently $x \equiv y \pmod{\pi}$. Since G is $LL(\pi)$, $\beta = \gamma$ $(j=k)$. Therefore, $L_{ij} \cap L_{ik} = \Phi$ (for $j \neq k$) can only be true when $\beta \neq \gamma$.

In order to satisfy the conditions of the proposition above, we must construct the sets L_{ij}. To that end, consider the following sets (where $G = (N,T,P,S)$ and $\pi = (R_1,\ldots,R_n)$ is a regular partition of T^*):

For each production $j:A \rightarrow \beta$ in P define the following cfg's:

$$G_j = (N_j,T,P_j,S_j)$$

where $N_j = N \cup \{X' \mid X \in N\} \cup \{S_j\}$,

$P_j = P \cup \{A' \rightarrow \beta \mid (j:A \rightarrow \beta) \in P\}$

$\cup \{X' \rightarrow X_i' X_{i+1} \ldots X_n \mid (X \rightarrow X_1 X_2 \ldots X_n) \in P$ for all $1 \leq i \leq n$, $X_i \in N\}$.

PROPOSITION 5.2: $L(G_j) = L_1$ where $L_1 = \{x \in T^* \mid S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx$ is a derivation in G using production $j:A \rightarrow \beta\}$.

PROOF: $L_1 \subseteq L(G_j)$. It will be shown first by induction on the length of a derivation that if $S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha$ then

there exists a derivation $S_j \overset{*}{\underset{lm}{\Rightarrow}}_G A'\alpha$.

(0 steps) S derives S in 0 steps in G; $S_j \overset{*}{\underset{lm}{\Rightarrow}}_G S'$.

(assume if $S \overset{n}{\underset{lm}{\Rightarrow}}_G wA\alpha$ then there exists a derivation $S_j \overset{*}{\underset{lm}{\Rightarrow}}_G A'\alpha$) Let $S \overset{*}{\underset{lm}{\Rightarrow}}_G wB\alpha \underset{lm}{\Rightarrow}_G ww_1Aa_1\alpha$. Then $S_j \overset{*}{\underset{lm}{\Rightarrow}}_G B'\alpha$ by the induction hypothesis and $B'\alpha \underset{lm}{\Rightarrow}_G A'\alpha_1\alpha$ using the production $B'\rightarrow A'\alpha_1$ in P' that was obtained in the construction of G_j from the production $B\rightarrow w_1Aa_1$ in P.

Since we now have $S \overset{*}{\underset{lm}{\Rightarrow}}_G wA\alpha$ implies $S_j \overset{*}{\underset{lm}{\Rightarrow}}_G A'\alpha$, we obtain $S \overset{*}{\underset{lm}{\Rightarrow}}_G wA\alpha \underset{lm}{\Rightarrow}_G w\beta\alpha$ implies $S_j \overset{*}{\underset{lm}{\Rightarrow}}_G A'\alpha \underset{lm}{\Rightarrow}_G \beta x$ by using the production $A'\rightarrow\beta$ in P' obtained from $A\rightarrow\beta$ in P.

Finally $w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}}_G wx$ implies $\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}}_G x$ by using the same sequence of productions in G' as are used in G.

Consequently, if $S \overset{*}{\underset{lm}{\Rightarrow}}_G wA\alpha \underset{lm}{\Rightarrow}_G w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}}_G wx$ then $S_j \overset{*}{\underset{lm}{\Rightarrow}}_G A'\alpha \underset{lm}{\Rightarrow}_G \beta\alpha \overset{*}{\underset{lm}{\Rightarrow}}_G x$. Letting x be any element of L_1, we have then a derivatin of x in G' and therefore $x \in L(G_j)$. So $L_1 \subseteq L(G_j)$.

$L(G_j) \subseteq L_1$. By induction on the length of a derivation it will be shown that if $S_j \overset{*}{\underset{lm}{\Rightarrow}}_G A'\alpha$ then there exists a derivation $S \overset{*}{\underset{lm}{\Rightarrow}}_G wA\alpha$ for some w.

(1 step) $S_j \underset{lm}{\Rightarrow} S'$; S derives S in 0 steps in G.

(assume if $S_j \overset{n}{\underset{lm}{\Rightarrow}}_G A'\alpha$ then there exists a derivation $S \overset{*}{\underset{lm}{\Rightarrow}}_G wA\alpha$ for some w) Let $S_j \overset{n}{\underset{lm}{\Rightarrow}}_G B'\alpha$

$\Rightarrow_G$ $A'\alpha_1\alpha$. Then $S \overset{*}{\Rightarrow}_G wB\alpha$ for some w by the induction hypothesis and $wB\alpha \Rightarrow_G ww_1A\alpha_1\alpha$ where $B\rightarrow w_1A\alpha_1$ is a production in P that the production $B'\rightarrow A'\alpha_1$ in P' was obtained from.

Given that $S_j \overset{*}{\Rightarrow}_G A'\alpha$ implies that $S \overset{*}{\Rightarrow}_G wA\alpha$ for some w, we obtain $S_j \overset{*}{\Rightarrow}_G A'\alpha \Rightarrow_G \beta\alpha \overset{*}{\Rightarrow}_G x$ implies $S \overset{*}{\Rightarrow}_G wA\alpha \Rightarrow_G w\beta\alpha \overset{*}{\Rightarrow}_G wx$ by first using production $A\rightarrow\beta$ (from which $A'\rightarrow\beta$ in P' was obtained) followed by equivalent derivations of x from $\beta\alpha$ in both G and G'.

Letting x be any element of $L(G_j)$, we then have a derivation of wx in G; consequently $x \in L_1$. So $L(G_j) \subseteq L_1$.

Together $L_1 \subseteq L(G_j)$ and $L(G_j) \subseteq L_1$ imply $L(G_j) = L_1$.

Let $M_i = (Q,T,\delta,q_0,F)$ be a deterministic finite automaton that accepts the language R_i. For each cfg G_j from above and for each R_i in π define the following cfg's:

$$G_{ij} = (N_{ij},T,P_{ij},S_{ij})$$

where $N_{ij} = \{S_{ij}\} \cup \{[p,X,q] \mid X \in N_j \text{ and } p,q \in Q\}$,

$$P_{ij} = \{S_{ij} \rightarrow [q_0,S_j,p] \mid p \in F\}$$
$$\cup \{[p,X,q] \rightarrow [p,X_1,q_1]\ldots[q_{n-1},X_n,q] \mid (X \rightarrow X_1X_2\ldots X_n) \in P_j \text{ and } p,q_1,\ldots,q_{n-1} \in Q\}$$
$$\cup \{[p,A,p] \rightarrow \varepsilon \mid (A \rightarrow \varepsilon) \in P_j\}$$
$$\cup \{[p,a,q] \rightarrow a \mid a \in T \text{ and } \delta(q,a) = p\}.$$

The grammars G_{ij} then are put into reduced form by eliminating useless productions and non-terminals.

PROPOSITION 5.3: $L(G_{ij}) = L_2$ where $L_2 = \{x \in R_i \mid S \overset{*}{\underset{lm}{\Rightarrow}} \omega A\alpha \underset{lm}{\Rightarrow} \omega\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} \omega x$ is a derivation in G using production $j: A \to \beta\}$.

PROOF: $L(G_{ij}) \subseteq L_2$. Let $x \in L(G_{ij})$. Define homomorphism $h: Q \times (N_{ij} \cup T_{ij}) \times Q \to N_{ij} \cup T_{ij}$ as $h([p,X,q]) = X$. If $S_{ij} \underset{lm}{\Rightarrow} [q_0,S_j,p] \overset{*}{\underset{lm}{\Rightarrow}} z[q,X,q']\alpha \overset{*}{\underset{lm}{\Rightarrow}} x$ is a derivation of x in G_{ij} then $S_j \overset{*}{\underset{lm}{\Rightarrow}} zXh(\alpha) \overset{*}{\underset{lm}{\Rightarrow}} x$ is a derivation of x in G_j. Consequently $x \in L(G_j)$ and $x \in L_1$ so there is a derivation in G: $S \overset{*}{\underset{lm}{\Rightarrow}} \omega A\alpha \underset{lm}{\Rightarrow} \omega\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} \omega x$. Consider the grammar G_{ij}' consisting of the grammar G_{ij} with the productions $[p,a,q] \to a$ replaced by $[p,a,q] \to aq$ and with $S_{ij} \to [q_0,S_j,p]$ replaced by $S_{ij} \to q_0[q_0,S_j,p]$. If $x = a_1a_2\ldots a_n$ then elements of $L(G_{ij}')$ are of the form $q_0a_1q_1\ldots a_nq_n$ where $q_n \in F$. Then $x = a_1a_2\ldots a_n \in L(G_{ij})$, $\delta(q_i,a_{i+1}) = q_{i+1}$ for $0 \le i \le n-1$ and $x \in R_i$ since M_i accepts R_i. Therefore, if $x \in L(G_{ij})$ then $x \in L_2$.

$L_2 \subseteq L(G_{ij})$. It is easy to show by induction on the length of a derivation that if $B \overset{*}{\Rightarrow} y$ in G_j and $\delta(p,y) = q$ then there is a derivation Let $x \in L_2$, that is $S \overset{*}{\underset{lm}{\Rightarrow}} \omega x$ in G and $x \in R_i$. Suppose $\delta(q_0,x) = p$, where $p \in F$. Since $x \in L_2$ so also $x \in L_1$ and there is a derivation $S_j \overset{*}{\underset{lm}{\Rightarrow}} x$ in G_j.

Then by the induction above there is a derivation $S_{ij} \Rightarrow [q_0,S_j,p] \overset{*}{\Rightarrow} x$ in G_{ij} and consequently $x \in L(G_{ij})$.

Define the homomorphisms h_{ij} from N_{ij} to N_j where $h_{ij}([p,X,q]) = X$ (extended to N_{ij}^* in the standard way). For each cfg G_{ij} above define the following left-linear grammars:

$$H_{ij} = (N_{ij},T,P_{ij}',S_{ij})$$

where $P_{ij}' = \{[p_1,X,p_{n+1}] \rightarrow [p_1,X_1,p_2]X_2\ldots X_n \mid ([p_1,X,p_{n+1}] \rightarrow [p_1,X_1,p_2]\ldots[p_n,X_n,p_{n+1}]) \in P_{ij}\}$

$\cup \{[p,A',q] \rightarrow \epsilon \mid ([p,A',q] \rightarrow \beta) \in P_{ij}$ and $A' \rightarrow h_{ij}(\beta)$ is the production in P_j obtained from $(j:A \rightarrow \beta) \in P\}$.

PROPOSITION 5.4: $L(H_{ij}) = L_3$ where $L_3 = \{\alpha \mid S \overset{*}{\Rightarrow} wA\alpha \Rightarrow w\beta\alpha \overset{*}{\Rightarrow} wx$ is a derivation in G using production $j:A \rightarrow \beta$, and $x \in R_j\}$.

PROOF: $L(H_{ij}) \subseteq L_3$. Let $S_{ij} \overset{*}{\Rightarrow} [p,X,q]h_{ij}(\alpha_1) \overset{*}{\Rightarrow} [p',A',q']h_{ij}(\alpha_2\alpha_1) \Rightarrow h_{ij}(\alpha_2\alpha_1)$ be a derivation in G_{ij} where $\alpha_1,\alpha_2 \in N_{ij}$. Let $\alpha = h_{ij}(\alpha_2\alpha_1) \in L(H_{ij})$. Then there must be a derivation $S_{ij} \overset{*}{\Rightarrow} [p,S,q]\alpha_1 \overset{*}{\Rightarrow} [p',A',q']\alpha_2\alpha_1 \overset{*}{\Rightarrow} x$ in G_{ij}. Therefore $x \in L_2$ so $x \in R_j$ and there is a derivation $S \overset{*}{\Rightarrow} wA\alpha \Rightarrow w\beta\alpha \overset{*}{\Rightarrow} wx$. Therefore $\alpha \in L_3$.

$L_3 \subseteq L(H_{ij})$. Let $\alpha \in L_3$. Then there is a derivation $S \overset{*}{\Rightarrow} wA\alpha \Rightarrow w\beta\alpha \overset{*}{\Rightarrow} wx$ where $x \in R_j$, so

$x \in L_2$ and also $x \in L(G_{ij})$. Therefore there is a derivation $S_{ij} \overset{*}{\underset{lm}{\Rightarrow}} [p,A',q]\alpha' \overset{*}{\underset{lm}{\Rightarrow}} x$ in G_{ij}, where $\alpha = h_{ij}(\alpha')$. Consequently there is a derivation $S_{ij} \overset{*}{\underset{lm}{\Rightarrow}} [p,A',q]h_{ij}(\alpha') \overset{*}{\Rightarrow} h_{ij}(\alpha') = \alpha$ in H_{ij}, so $\alpha \in L(H_{ij})$.

Combining the two inclusions above results in $L(H_{ij}) = L_3$.

The decidability result can now be stated.

PROPOSITION 5.5: Let $G = (N,T,P,S)$ be a cfg and let $\pi = (R_1,\ldots,R_n)$ be a regular partition of T^*. It is decidable whether G is $LL(\pi)$ or not.

PROOF: The construction of the left-linear grammars H_{ij} above define the regular sets $L_{ij} = L(H_{ij})$. Since there are a finite number of regular sets L_{ij}, there are a finite number of intersections $L_{ij} \cap L_{ik}$. The L_{ij} are regular so it is decidable whether the intersection is empty or not. Therefore the decision procedure for $LL(\pi)$-ness consists of testing $L_{ij} \cap L_{ik}$ for emptiness for all i,j,k such that $j:A \to \beta$, $k:A \to \gamma$ and $\beta \neq \gamma$. By proposition 5.1, if any intersection is non-empty, G is not $LL(\pi)$; if all intersections are empty, G is $LL(\pi)$.

Although it is decidable whether a grammar is LL-regular for a given regular partition π, it is not decidable whether

a grammar has a regular partition for which it is LL-regular. The following lemma will help establish this fact.

LEMMA 5.6: If $G = (N,T,P,S)$ is LL-regular for the regular partition $\pi = (R_1,\ldots,R_n)$ then for any two productions $A\to\beta$ and $A\to\gamma$ in P $(\beta\neq\gamma)$, there exists a regular set R such that $\{x \mid S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx\} = L_1 \subseteq R$ and $\{y \mid S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \overset{*}{\underset{lm}{\Rightarrow}} wy\} \cap R = \Phi$.

PROOF: Define the sets $M = \{i \mid L_1 \cap R_i \neq \Phi\}$ and $R = \bigcup_{i\in M} R_i$. R is the union of all sets R_i in π that intersect the language L_1, so $L_1 \subseteq R$. Suppose $R \cap L_2 \neq \Phi$, and let $y \in R \cap L_2$. Then $y \in R_i$ for some $i \in M$. Since $i \in M$ there must be some $x \in R_i$ such that $x \in L_1$. Consider the derivations

$$S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx$$

$$S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\gamma\alpha \overset{*}{\underset{lm}{\Rightarrow}} wy.$$

$x \in R_i$, $y \in R_i$ so $x \equiv y \pmod{\pi}$. Since G is $LL(\pi)$, $\beta = \gamma$. Consequently $y \in L_1$. Since G is unambiguous, y is not in L_2. Therefore there is no $y \in R \cap L_2$, that is $R \cap L_2 = \Phi$.

PROPOSITION 5.7: It is not decidable whether or not a context-free grammar $G = (N,T,P,S)$ is LL-regular.

PROOF: Let $G_1 = (N_1,T_1,P_1,S_1)$ and $G_2 = (N_2,T_2,P_2,S_2)$ be any two cfg's where $N_1 \cap N_2 = \Phi$. Let $G = (N_1 \cup N_2 \cup \{S\}, T_1 \cup T_2, P_1 \cup P_2 \cup \{S\to S_1, S\to S_2\}, S)$. By the previous lemma, if G has a regular partition π such that G

is $LL(\pi)$ then there exists a regular set R such that for productions $S \to S_1$ and $S \to S_2$, $\{x \mid S \overset{*}{\Rightarrow} S \Rightarrow S_1 \overset{*}{\Rightarrow} x\} = L(G_1) = L_1 \subseteq R$ and $\{y \mid S \overset{*}{\Rightarrow} S \Rightarrow S_2 \overset{*}{\Rightarrow} y\} = L(G_2) = L_2 \cap R = \phi$. Therefore, deciding whether or not there exists a π for which G is $LL(\pi)$ implies deciding whether or not there is a regular separating set for the cfg's L_1 and L_2. This problem has been shown to be undecidable. Consequently, it is undecidable whether π exists or not.

6. CONVERTING LL-REGULAR GRAMMARS TO STRONG LL-REGULAR GRAMMARS

Although the class of LL-regular grammars properly includes the class of strong LL-regular grammars as was shown in section 3, the class of LL-regular languages is equivalent to the class of strong LL-regular languages. This equivalence follows from the fact that, for any LL-regular grammar, a strong LL-grammar that generates the same language can be constructed.

In this section a method for constructing a strong LL-regular grammar $G' = (N',T,P',S')$ from an LL-regular grammar $G = (N,T,P,S)$ will be given. G' will be "structurally equivalent" to G in the following sense: 1. $L(G) = L(G')$. 2. There exists a homomorphism h from P' to P such that for

any leftmost derivation p in G', h(p) is a leftmost derivation in G such that $S' \underset{lm}{\overset{p}{\Rightarrow}}_{G'} x$ and $S \underset{lm}{\overset{h(p)}{\Rightarrow}}_{G} x$, where $x \in T^*$. This means that h can be used to recover a derivation in G of any $x \in L(G)$ from a derivation of x in G'.

In order to construct the new grammar G', we first need to construct a deterministic finite automata M that is the cross product of the automata $M_{i,j}$ accepting the languages $L(H_{i,j})$ reversed for the grammars $H_{i,j}$ defined in section 5. Formally, let $M_{i,j} = (Q_{i,j}, T, \delta_{i,j}, q_{i,j}', F_{i,j})$ be a deterministic finite automata accepting $L(H_{i,j})$ reversed. Let $M = (Q, T, \delta, q_0, \phi)$, where $Q = \{[q_{1,1}, q_{1,2}, \ldots, q_{n,k}] \mid q_{i,j} \in Q_{i,j}\}$, $q_0 = [q_{1,1}', q_{1,2}', \ldots, q_{n,k}']$, and $\delta([q_{1,1}, q_{1,2}, \ldots, q_{n,k}], a) = [\delta_{1,1}(q_{1,1}, a), \delta_{1,2}(q_{1,2}, a), \ldots, \delta_{n,k}(q_{n,k}, a)]$ for all q in Q and a in T.

The construction of G' now follows:

CONSTRUCTION 6.1: Let G = (N,T,P,S) be a cfg. Let M = (Q,T,δ,q_0,F) be the deterministic finite automata constructed above. Then G' = (N',T,P',S'), where N' = $\{[X,q] \mid X \in (N \cup T), q \in Q\}$, S' = $[S, q_0]$, and P' = $\{[X,q] \rightarrow [X_1,q_1][X_2,q_2]\ldots[X_n,q_n] \mid (X \rightarrow X_1X_2\ldots X_n) \in P, q_1, q_2, \ldots, q_n \in Q, q_n = q, q_{i-1} = \delta(q_i, X_i)\} \cup \{[A,q] \rightarrow \epsilon \mid (A \rightarrow \epsilon) \in P\} \cup \{[a,q] \rightarrow a \mid a \in T, q \in Q\}$.

PROPOSITION 6.1: The grammar G' constructed above from G is structurally equivalent to G.

PROOF: Straightforward induction on the lengths of

derivations results in showing that $\{x \mid S \overset{*}{\underset{lm}{\Rightarrow}}_G x\}$ $= \{x \mid [S,q_0] \overset{*}{\underset{lm}{\Rightarrow}}_{G'} x\}$. The homomorphism h defined as $h([X,q]\rightarrow[X_1,q_1]\ldots[X_n,q_n]) = (X\rightarrow X_1\ldots X_n)$ and $h([a,q]\rightarrow a) = \lambda$ produces a derivation h(p) in G from a derivation p in G′ of any $x \in L(G) = L(G')$.

PROPOSITION 6.2: If the grammar G is LL-regular for π, then the grammar G′ constructed above is strong LL-regular for π.

PROOF: For the purposes of this proof, the strings α, α_1, α_2, β, γ are obtained from the strings α', α_1', α_2', β', γ' by deleting the state component from each symbol in the string.

We first notice that by the construction of G′ that if $[X,q] \Rightarrow [X_1,q_1]\ldots[X_n,q_n]$ then $q_1 = \delta(q_n,X_n\ldots X_2)$ and straightforward induction on the length of a derivation shows that if $[S,q_0] \overset{*}{\underset{lm}{\Rightarrow}}_G w[A,q]\alpha'$ then $q = \delta(q_0,\alpha^R)$. We also notice that if $[A,q]\alpha' \underset{lm}{\Rightarrow}_G \beta'\alpha' \overset{*}{\underset{lm}{\Rightarrow}}_G x$ and $x \in R_i$ then the component fsa of q corresponding to M_{ij} will be in a final state, that is $q_{ij} \in F_{ij}$ in M_{ij} so that $\alpha \in L(H_{ij})$.

Now consider the derivations

$$[S,q_0] \overset{*}{\underset{lm}{\Rightarrow}} w_1[A,q]\alpha_1' \underset{lm}{\Rightarrow} w_1\beta'\alpha_1' \overset{*}{\underset{lm}{\Rightarrow}} w_1x$$

$$[S,q_0] \overset{*}{\underset{lm}{\Rightarrow}} w_2[A,q]\alpha_2' \underset{lm}{\Rightarrow} w_2\gamma'\alpha_2' \overset{*}{\underset{lm}{\Rightarrow}} w_2y$$

in G′ and let $x,y \in R_i$ (that is $x \equiv y \pmod{\pi}$). We want to show that $\beta' = \gamma'$. Consider the

corresponding derivations in G:

$$S \underset{lm}{\Rightarrow} w_1 A\alpha_1 \underset{rm}{\Rightarrow} w_1 \beta\alpha_1 \underset{lm}{\Rightarrow} w_1 x$$

$$S \underset{lm}{\Rightarrow} w_2 A\alpha_2 \underset{rm}{\Rightarrow} w_2 \gamma\alpha_2 \underset{lm}{\Rightarrow} w_2 y$$

where $j:A\rightarrow\beta$, $k:A\rightarrow\gamma$ and $x,y \in R_i$. We know that $\alpha_1 \in L(H_{ij})$, $\alpha_2 \in L(H_{ik})$ and since G is LL(π), $L(H_{ij}) \cap L(H_{ik}) = \phi$ unless $j = k$. But the derivations in G' insure that state q is identical in both [A,q]'s. The only way for this to occur is for $j=k$. Therefore $\beta=\gamma$. The construction then insures that $\beta'=\gamma'$ since the state components are uniquely determined once q has been fixed. Consequently, G' is strong LL-regular for π

7. AN LL-REGULAR PARSING ALGORITHM

An efficient two pass parsing algorithm can be constructed for LL-regular grammars which is analogous to the two pass parsing algorithm for LR-regular grammars described in [2]. First the reverse of the input string is gsm-mapped into another string which has information about the regular-partition-lookahead attached to each symbol in the original string. This new string is then parsed by an LL(k)-type parser where instead of having k-symbol lookahead, the parser uses the regular-partition-lookahead information which has been attached to each input symbol.

The first step in constructing a parser for an LL-regular grammar $G = (N,T,P,S)$ with regular partition $\pi = (R_1,\ldots,R_n)$ of T^* is to construct the gsm that maps the input string. Let $g:T^* \rightarrow T^* \times \{1,\ldots,n\}$ be defined as follows: $g(xa) = g(x)[a,i]$ such that $xa \in R_i$ reversed and $g(\epsilon) = \epsilon$. Thus for any string $x = a_1 a_2 \ldots a_k \in T^*$, $g(a_k a_{k-1} \ldots a_1) = [a_k,i_k]\ldots[a_1,i_1]$ where $a_j \ldots a_1 \in R_{i_j}$ for all $1 \leq j \leq k$, that is, the symbol i_j attached to each symbol a_j of x indicates which regular set of the partition π the remaining input string is in. Therefore, by preprocessing the input string, the lookahead information will be attached to the input string at the point where it will be needed when parsing.

The second step in constructing a parser is to construct a parsing function M.

DEFINITION 7.1: Let $G = (N,T,P,S)$ be a cfg and $\pi = (R_1,\ldots,R_n)$ be a regular partition of T^*. $M:N \times \{1,\ldots,n\} \rightarrow$ subsets of $((N \cup T)^*)$ where $\beta \in M(A,i)$ if there is a derivation $S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx$ such that $x \in R_i$.

M can be constructed from the grammars G_{ij} defined in section 5. Recall that $L(G_{ij}) = \{x \in R_i \mid S \overset{*}{\underset{lm}{\Rightarrow}} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \overset{*}{\underset{lm}{\Rightarrow}} wx\}$; it is straightforward to show that $\beta \in M(A,i)$ whenever $G_{ij} \neq \emptyset$, where $(j:A \rightarrow \beta) \in P$.

The parsing function M for LL-regular grammars is unfortunately not single valued. This is because the

definition requires for specific R_i, A and α that $\beta = \gamma$. For $\alpha' \neq \alpha$ but the same R_i and A it does not require that $\beta = \gamma$ and as a result M will not generally be single valued. Fortunately, M is assured to be single valued for strong LL-regular grammars.

PROPOSITION 7.1: Let G = (N,T,P,S) be a strong LL-regular grammar for the regular partition $\pi = (R_1,\ldots,R_n)$ of T*. Then the function M defined above is single valued.

PROOF: Suppose $\beta \in M(A,i)$ and $\gamma \in M(A,i)$: Then there must be derivations

$$S \underset{lm}{\Rightarrow} wA\alpha \underset{lm}{\Rightarrow} w\beta\alpha \underset{lm}{\Rightarrow} wx$$

$$S \underset{lm}{\Rightarrow} w'A\alpha' \underset{lm}{\Rightarrow} w'\gamma\alpha' \underset{lm}{\Rightarrow} w'y$$

where $x \in R_i$ and $y \in R_i$, that is $x \equiv y \pmod{\pi}$. But since G is a strong LL-regular grammar for π, $\beta=\gamma$. Therefore M is single valued.

Consequently construction of single valued M requires that the grammar be strong LL-regular. In the previuos section we showed that any LL-regular grammar can be transformed into a structurally equivalent strong LL-regular grammar. We can construct the parsing function M for the strong LL-regular grammar and parse in it; since it is structurally equivalent to the original grammar, we can recover a parse in the LL-regular grammar from the parse in the strong LL-regular grammar.

The parsing algorithm for LL-regular grammars uses M and a stack to parse a gsm mapped input string $x = [a_1,i_1]\ldots[a_k,i_k]\$$ using:

ALGORITHM 7.1: LL-regular parser.

1. Initialize stack to S$, where $ is a new symbol not in the terminal alphabet of the language to be parsed.
2. If the symbol on top of the stack is $ and the next input symbol is $, halt and accept the input. If the symbol on top of the stack is $a \in T$ and the next input symbol is $[a,i]$ then pop a from the stack and advance the input to the next symbol; repeat step 2. If the symbol on top of the stack is $a \in T$ and the next input symbol is not $[a,i]$ for any i then halt and announce an error. If the symbol on top of the stack is $A \in N$ and the next input symbol is $[a,i]$ then replace A on the stack by $M(A,i)$; repeat step 2.

In summary, an LL-regular parser for a grammar $G = (N,T,P,S)$ and a regular partition $\pi = (R_1,\ldots,R_n)$ of T^* consists of a gsm g and a parsing function M. To parse a string x, first map x into x′ using the gsm g. Second, append $ to the end of x′ giving x′$. Third, use the parsing algorithm with function M to obtain the leftmost parse of x. Finally, if the original grammar was LL-regular, map the derivation obtained into a derivation in

the original grammar.

8. SUMMARY

In this paper we introduced a logical generalization of the class of LL-regular grammars. This larger class of grammars was shown to properly include the LL(k) grammars, to be properly included in the LR-regular grammars and to be incomparable with the LR(k) grammars. A decision procedure was described which allowed one to test for LL-regularness for a given grammar and regular partition; it was shown that no decision procedure exists when only the grammar is given.

A two-pass parsing algorithm for LL-regular grammars was described. This algorithm produces the left-most parse of an input string in linear time. Since LL-regular grammars are a larger class of grammars than the LL(k) grammars, the class of linear-time parsable top-down grammars has been extended.

REFERENCES

1. M. E. Conway, Design of a separable transition-diagram compiler, Communications ACM 6:7 (1963) 396-408.

2. K. Culik and R. Cohen, LR-regular grammars - an extension of LR(k) grammars, JCSS 7 (1973) 66-96.

3. K. Culik, Contribution to deterministic top-down analysis of context-free languages, Kybernetika 4:5 (1968) 422-431.

4. J. E. Hopcroft and J. D. Ullman, "Formal languages and their relation to automata", Addison Wesley, Reading, MA, 1969.

5. S. Jarzabek and T. Krawczyk, LL-regular grammars, Information Processing Letters 4:2 (1975) 31-37.

6. P. M. Lewis and R. E. Stearns, Syntax directed transduction, Journal ACM 15:3 (1969) 465-488.

7. D. J. Rosenkrantz and P. M. Lewis, Deterministic left corner parsing, IEEE Conf. Record 11th Annual Symposium on Switching and Automata Theory (1970) 139-152.

8. D. J. Rosenkrantz and R. E. Stearns, Properties of deterministic top-down grammars, Information and Control 17:3 (1970) 226-256.

9. D. Wood, The theory of left factored languages, Computer Journal 12:4 (1969) 349-356 and 13:1 (1970) 55-62.