Theoretical Computer Science 4 (1977) 245–276.

ON LR(k) GRAMMARS AND LANGUAGES*

Matthew M. GELLER[1] and Michael A. HARRISON
Computer Science Division, University of California at Berkeley, CA 94720, U.S.A.

Communicated by Ronald Book
Received August 1975
Revised August 1976

Abstract. Many different definitions for LR(k) grammars exist in the literature. One of these definitions is chosen and many important implications are drawn from it. In particular, the LR(k) characterization theorem provides valuable information about chains of derivations. The LR(0) languages are then characterized by acceptance by deterministic pushdown automata with a special termination condition, by a condition on the strings in the language, and set theoretically. Important closure properties of the LR(0) languages and a related class of languages are then examined. These are used to examine some decidability questions relating to the class of LR languages. One of these questions is shown to be equivalent to the equality problem for deterministic pushdown automata.

A survey of other LR(k) definitions is given and the exact differences are characterized. On the basis of this analysis, justification for the choice of definition used here is provided.

1. Introduction

LR(k) grammars and languages were introduced about a decade ago [15]. They were claimed to be an exact counterpart to deterministic context free languages [9] and so it was immediately clear that they were a theoretically important family. Moreover, it was claimed that this was the largest class of grammars for which left-to-right bottom up deterministic parsing was possible. Because of this, there has been a great deal of work in this area. These grammars and languages play an important role in Computer Science textbooks in the area. Cf. [1] and its many references and citations to this subject.

The present paper is the first of a series of related papers. In the present paper, we shall compare most of the commonly used definitions of LR(k) grammars and the exact differences will be characterized. The major differences occur when $k = 0$

* Research supported by the National Science Foundation under grant NSF GJ-43332. A preliminary version of some of this research was presented at the 14th Annual Symposium on Switching and Automata Theory, Iowa City, Iowa, 1973. Cf. [6].

[1] Present Address: Department of Computer and Communication Sciences, The University of Michigan, Ann Arbor, MI 48104, U.S.A.

and so the family of LR(0) languages will be characterized in three very different and very striking ways. On the basis of all of the results given here, it will be argued that our definition is the most natural one for LR(k) grammars. The evidence given here for that thesis is convincing and a sequel [7] to this paper which considers the parsers strengthens the argument even more.

The characterizations of LR(0) languages, along with closure results that are proven for LR(0) languages, are used to prove that deciding whether a deterministic language is LR(0) is equivalent to deciding the equivalence of deterministic pushdown automata. It is quite surprising to find this open question occurring in the context of LR(0) testing.

The paper is organized in the following manner. You are now reading Section 1 which will conclude with some of the notation the reader must endure. Section 2 gives our definition of LR(k) grammars and some important consequences of the definition such as unambiguity and the "extended LR(k) theorem". Relations with other definitions of LR(k) are summarized. Section 3 gives three quite different characterizations of the LR(0) languages. In Section 4, closure properties of several families are proved and are used to deal with some decision problems. It is shown that one can decide if a deterministic language is strict deterministic (equivalently prefix free). One can decide whether or not a deterministic language is LR(0) if and only if one can decide if two deterministic context free languages are equal.

Mathematical formalism is needed to deal with strings, sets, and context free grammars and languages. We use the conventional notations and shall not reproduce them here. Cf. [11] for a summary asummary of our conventions. We shall reproduce below a few definitions which are less familiar.

Let $G=(V,\Sigma,P,S)$ be a context free grammar. We define a relation $\Rightarrow\ \subseteq V^*\times V^*$ as follows. For any $\alpha,\beta\in V^*$, $\alpha\overset{\rho}{\Rightarrow}\beta$ if and only if $\alpha=\alpha_1A\alpha_2$, $\beta=\alpha_1\beta_1\alpha_2$ and $A\to\beta_1=\rho\in P$ for some $A\in N$ and $\alpha_1,\alpha_2,\beta_1\in V^*$. In particular, if $\alpha_1\in\Sigma^*$ or $\alpha_2\in\Sigma^*$ we write $\alpha\overset{\rho}{\underset{L}{\Rightarrow}}\beta$ or $\alpha\overset{\rho}{\underset{R}{\Rightarrow}}\beta$ respectively. We may omit the ρ if it is not relevant. Any $\alpha\in V^*$ is called a (*canonical*) *sentential form* if and only if $S\overset{*}{\Rightarrow}\alpha$ ($S\overset{*}{\underset{R}{\Rightarrow}}\alpha$).

We need the formal concept of a canonical derivation. Let $G=(V,\Sigma,P,S)$ be a context free grammar and suppose that

$$S=\alpha_0\overset{\rho_0}{\underset{R}{\Rightarrow}}\alpha_1\overset{\rho_1}{\underset{R}{\Rightarrow}}\cdots\overset{\rho_{n-1}}{\underset{R}{\Rightarrow}}\alpha_n\in\Sigma^*.$$

If for each i, $0\leqslant i<n$, $\alpha_i=\alpha_i'A_ix_i$, $\alpha_{i+1}=\alpha_i'\beta_ix_i$ where $\alpha_i',\beta_i\in V^*$, $x_i\in\Sigma^*$, $A_i\in N$, and $\rho_i=A_i\to\beta_i$ is in P then $\rho=\rho_0\cdots\rho_{n-1}$ is said to be a *canonical derivation*. For $n\geqslant 0$, we may write $S\overset{n}{\underset{R}{\Rightarrow}}\alpha_n$ to indicate the number of steps in the derivation sequence. A context free grammar G is said to be *unambiguous* if each $x\in L(G)$ has exactly one canonical derivation.

We will also need the idea of a "handle".

Definition 1.1. Let $G = (V, \Sigma, P, S)$ be a context free grammar and let $\gamma \in V^*$. A *handle* of γ is an ordered pair (p, i) where $p \in P$ and $i \geq 0$ such that there exist $A \in N$, α, $\beta \in V^*$ and $w \in \Sigma^*$ such that
(i) $S \underset{R}{\overset{*}{\Longrightarrow}} \alpha A w \underset{R}{\Longrightarrow} \alpha\beta w = \gamma$,
(ii) p is $A \to \beta$,
(iii) $i = \lg(\alpha\beta)$.

Some special terminology is needed for dealing with strings. Let $\alpha, \beta \in V^*$ be two strings. Then α is a *prefix* (*suffix*) of β if and only if $\alpha = \beta\gamma$ ($\beta = \gamma\alpha$) for some $\gamma \in V^*$; when $\gamma \neq \Lambda$, α is a *proper prefix* (*proper suffix*) of β. For any $n \geq 0$, define

$$^{(n)}\alpha\,(\alpha^{(n)}) \quad \text{is the prefix (suffix) of } \alpha$$
$$\text{with length } \min(\lg(\alpha), n).$$

We say that a language $L \subseteq \Sigma^*$ is *prefix free* if $\alpha \in L$ and $\alpha\beta \in L$ implies[1] $\beta = \Lambda$.

We wish to perform certain operations on languages. For $L \subseteq \Sigma^*$, we say that

$$\min(L) = \{x \in L \mid \text{there does not exist a } y \in \Sigma^+ \text{ such that } xy \in L\}$$

and

$$\max(L) = \{z \in L \mid \text{there does not exist an } x \in \Sigma^*,\ y \in \Sigma^+ \text{ such that } xy = z\}.$$

Let $X, Y \subseteq \Sigma^*$. Then XY^{-1}, the quotient of X with Y, is defined as follows:

$$XY^{-1} = \{x \in \Sigma^* \mid \text{there exists a } y \in Y \text{ such that } xy \in X\}.$$

It will also be necessary to have the terminology to deal with deterministic pushdown automata, cf. [1, 8, 9, 11].

Definition 1.2. A *deterministic pushdown automaton* (abbreviated *DPDA*) is a 7-tuple

$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$$

where Q is a finite nonempty set, Σ and Γ are two alphabets, $q_0 \in Q$, $Z_0 \in \Gamma$, $F \subseteq Q$ and δ is a partial function

$$\delta : Q \times (\Sigma \cup \{\Lambda\}) \times \Gamma \to_p Q \times \Gamma^*$$

with the property that for any $q \in Q$ and $Z \in \Gamma$, $\delta(q, \Lambda, Z) \neq \emptyset$ implies $\delta(q, a, Z) = \emptyset$ for all $a \in \Sigma$.

Next we must describe how a DPDA moves.

[1] Λ denotes the null string.

Definition 1.3. Let $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$ be a DPDA and let $\mathcal{Q} = Q \times \Sigma^* \times \Gamma^*$. The *yield relation of M*, $\vdash \subseteq \mathcal{Q} \times \mathcal{Q}$ is defined as follows: For any $q, q' \in Q$, $a \in \Sigma \cup \{\Lambda\}$, $w \in \Sigma^*$, $\alpha, \beta \in \Gamma^*$ and $Z \in \Gamma$, $(q, aw, \alpha Z) \vdash (q', w, \alpha\beta)$ if and only if $\delta(q, a, Z) = (q', \beta)$. As in the case of derivations we have $\vdash^*$ for yields in 0 or more steps, $\vdash^+$ for yields in 1 or more steps, and, for $n \geq 0$, $\vdash^n$ for yields in n steps.

We now endow a DPDA with an ability to define, or *accept*, certain languages over its input alphabet.

Definition 1.4. Let $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$. For a given $K \subseteq \Gamma^*$ define the language $T(M, K) \subseteq \Sigma^*$ as follows:

$$T(M, K) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \Lambda, \alpha) \text{ for some } q \in F \text{ and } \alpha \in K\}.$$

In particular let

$$T_0(M) = T(M, \Gamma^*),$$
$$T_1(M) = T(M, \Gamma),$$
$$T_2(M) = T(M, \Lambda).$$

For $i = 0, 1, 2$, let $\Delta_i = \{T_i(M) \mid M \text{ is a DPDA}\}$. By [11] Δ_2 is the family of strict deterministic languages, while Δ_0 is the collection of deterministic languages, cf. [9, 11]. Δ_1 has only been briefly studied in [11]. Δ_2 is a particularly important family; among other reasons, each $L \in \Delta_0$ can be mapped into Δ_2 by "endmarking", i.e. $L \to L\$$ [11, 12, 13].

The reader will soon discover that our definitions and results are quite technical. In order to keep the size of the present paper under control, it has been necessary to delete proofs or merely sketch the arguments. Full proofs may be obtained by writing to the authors or by consulting [4].

2. Definition of LR(k) grammars and some basic consequences

Many different definitions of LR(k) grammars have been given in the literature [1, 15, 17, 18]. We will start this section with our definition. A number of basic implications of the definition are developed. First, it is observed that LR(k) grammars are unambiguous. The definition of LR(k) is extended to certain cross sections of derivation trees and a useful result, called the extended LR(k) theorem, is proven. Our definition is then compared to other definitions which have been given before.

We now present our definition of an LR(k) grammar. Our definition is the same as the one used in [4–7, 11–13] and is quite similar to the one provided in [15].

There are however several differences between the present definition and [15]. In our definition, we have excluded $S \underset{R}{\overset{+}{\Longrightarrow}} S$, and we also have not included endmarkers.

Definition 2.1. Let $k \geq 0$ and $G = (V, \Sigma, P, S)$ be a reduced context free grammar such that $S \underset{R}{\overset{+}{\Longrightarrow}} S$ is impossible in G. G is $LR(k)$ if for each $w, w', x \in \Sigma^*$; $\gamma, \alpha, \alpha', \beta, \beta' \in V^*$; $A, A' \in N$, if

(i) $S \underset{R}{\overset{*}{\Longrightarrow}} \alpha A w \underset{R}{\Longrightarrow} \alpha\beta w = \gamma w$, [that is, γw has handle $(A \to \beta,\ \lg(\alpha\beta))$],

(ii) $S \underset{R}{\overset{*}{\Longrightarrow}} \alpha' A' x \underset{R}{\Longrightarrow} \alpha'\beta' x = \gamma w'$, [that is, $\gamma w'$ has handle $(A' \to \beta',\ \lg(\alpha'\beta'))$],

(iii) ${}^{(k)}w = {}^{(k)}w'$,

then

(iv) $(A \to \beta,\ \lg(\alpha\beta)) = (A' \to \beta',\ \lg(\alpha'\beta'))$.

The conclusion in the definition, that is (iv), has several implications.

(1) By the definition of equality of ordered pairs, we have $A = A'$, $\beta = \beta'$, and $\lg(\alpha\beta) = \lg(\alpha'\beta')$.

(2) $\gamma = {}^{(\lg(\gamma))}\alpha'\beta' = {}^{(\lg(\alpha\beta))}\alpha'\beta' = {}^{(\lg(\alpha'\beta'))}\alpha'\beta' = \alpha'\beta'$. Thus $\gamma = \alpha\beta = \alpha'\beta'$.

(3) Since $\beta = \beta'$, from (2) we have $\alpha = \alpha'$.

(4) $\alpha'\beta' x = \gamma w'$ implies $\alpha'\beta' x = \alpha'\beta' w'$ implies $x = w'$. Note that if G is LR(k), G is LR(k') for all $k' \geq k$.

We shall be comparing a number of definitions which are similar to Definition 2.1. To simplify the presentation of these definitions, let us agree to call the main part of the definitions, parts (i) through (iv) including the quantification, the *body* of Definition 2.1.

One of the properties that we wish a grammatical class to possess, in order that it constitute a useful class of parsing, is unambiguity. We show that the LR(k) grammars are unambiguous. Although this result is claimed in [15, 18], their proofs are incorrect as will be seen later. We begin with two lemmas, and then present the proof.

The first lemma shows that given a sentential form for a reduced context free grammar, if we specify a handle by which to reduce, we uniquely determine the sentential form to which it will be reduced, and conversely, if we specify a sentential form to which it can be reduced, this determines a unique handle. This lemma does not require that the grammar be LR(k).

Lemma 2.2. *Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar. Assume that for $\alpha, \alpha', \beta, \beta' \in V^*$; $w, w' \in \Sigma^+$; $A, A' \in N$*

$$S \underset{R}{\overset{*}{\Longrightarrow}} \alpha A w \underset{R}{\Longrightarrow} \alpha\beta w$$

and

$$S \overset{*}{\underset{R}{\Longrightarrow}} \alpha' A' w' \underset{R}{\Longrightarrow} \alpha'\beta' w' = \alpha\beta w.$$

Then $\alpha A w = \alpha' A' w'$ *if and only if*

$$(A \to \beta, \lg(\alpha\beta)) = (A' \to \beta', \lg(\alpha'\beta')).$$

Proof. The argument is elementary and is omitted. □

The second lemma characterizes unambiguous grammars in terms of handles.

Lemma 2.3. *Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar. Then G is unambiguous if and only if every canonical sentential form has exactly one handle except S, which has none.*

Proof. The argument is omitted. □

Now we shall apply these results to verify that every LR(k) grammar is unambiguous.

Theorem 2.4. *Let $G = (V, \Sigma, P, S)$ be an LR(k) grammar, $k \geqslant 0$. Then G is unambiguous.*

Proof. The argument is a straightforward application of Lemma 2.3 and is omitted. □

The following lemma tells us when a grammar is not LR(k). Consequently the lemma is often useful in proofs by contradiction.

Lemma 2.5. *Let $k \geqslant 0$ and $G = (V, \Sigma, P, S)$ be a reduced context free grammar such that $S \Rightarrow_R^+ S$ is impossible in G. G is not LR(k) if and only if there exist $w, w', x \in \Sigma^*$; $A, A' \in N$; $\gamma', \gamma, \alpha, \alpha', \beta, \beta' \in V^*$ such that*

(i) $S \Rightarrow_R^* \alpha A w \Rightarrow_R \alpha\beta w = \gamma w$,

(ii) $S \Rightarrow_R^* \alpha' A' x \Rightarrow_R \alpha'\beta' x = \gamma' x = \gamma w'$,

(iii) ${}^{(k)}w = {}^{(k)}w'$, *and*

(iv) $(A \to \beta, \lg(\alpha\beta)) \neq (A' \to \beta', \lg(\alpha'\beta'))$ *with*

(v) $\lg(\alpha'\beta') \geqslant \lg(\alpha\beta)$.

Proof. Simply negate the definition. If (v) is not satisfied then the (i) and (ii) can be reversed so that it is satisfied. □

The following theorem will be extremely useful in studying the class of LR(0) languages. It is an inductive version of the definition of an LR(k) grammar.

Theorem 2.6. (*Extended LR(k) theorem*). *Suppose $G = (V, \Sigma, P, S)$ is an LR(k) grammar and there exist $\alpha \in V^*$; $x_1, x_2, w \in \Sigma^*$ such that*

(i) $S \Rightarrow_R^* \alpha x_1 \Rightarrow_R^+ wx_1$,

(ii) $S \Rightarrow_R^+ wx_2$,

(iii) ${}^{(k)}x_1 = {}^{(k)}x_2$,

(iv) $x_1 \neq \Lambda$, $k > 0$, *or $k = 0$ and there exists no $x \in \Sigma^*$ such that Sx is a sentential form of G with a handle whose second component is 1,*[2]

then

(v) $S \Rightarrow_R^* \alpha x_2 \Rightarrow_R^+ wx_2$.

Proof. We assume for the sake of contradiction that (i), (ii), (iii), and (iv) hold, but not (v). Suppose $\alpha x_1 \Rightarrow_R^+ wx_1$ is a derivation of n steps, where $n \geqslant 1$, by the (unique) derivation

$$\alpha x_1 = \alpha_n x_1 \underset{R}{\Rightarrow} \alpha_{n-1} x_1 \underset{R}{\Rightarrow} \cdots \underset{R}{\Rightarrow} \alpha_1 x_1 = wx_1$$

with $\alpha_i \in V^*$, for $1 \leqslant i \leqslant n$. Let m be the number of steps in the derivation $S \Rightarrow_R^+ wx_2$, and let $r = \min(m, n)$.

Now, suppose that the last r steps of the derivation $S \Rightarrow_R^+ wx_2$ are

$$\alpha'_r \underset{R}{\Rightarrow} \alpha'_{r-1} \underset{R}{\Rightarrow} \cdots \underset{R}{\Rightarrow} \alpha'_1 = wx_2$$

for some $\alpha_i \in V^*$, for $1 \leqslant i \leqslant r$.

Claim. *There exists some $l \leqslant r$ such that $\alpha'_l \neq \alpha_l x_2$.*

Proof. By contradiction. Suppose $\alpha'_l = \alpha_l x_2$ for all $l \leqslant r$.

Case 1. $r < n$. Then $\alpha'_r = \alpha_r x_2 = S$. Thus $\alpha_r = S$ and $x_2 = \Lambda$. Since ${}^{(k)}x_1 = {}^{(k)}x_2$ we must have $k = 0$ or $x_1 = \Lambda$. If $x_1 = \Lambda$ then $S \Rightarrow_R^+ S$ which contradicts the fact that G is LR(k). Therefore, $k = 0$. However, we know that $\alpha_{r+1} x_1 \Rightarrow_R \alpha_r x_1 = Sx_1$. The handle of Sx_1 has second component 1, contradicting (iv).

Case 2. $r = n$. Again $\alpha'_r = \alpha_r x_2$. We have $S \Rightarrow_R^* \alpha'_r = \alpha_r x_2 = \alpha_n x_2 = \alpha x_2 \Rightarrow_R^+ wx_2$. But this is (v), which is assumed to be false. Thus, we have a contradiction and the Claim is established.

Now let m be the smallest positive integer satisfying our claim. Clearly $m > 1$, since $\alpha'_1 = wx_2 = \alpha_1 x_2$.

Now, we know that there exist $\bar{\alpha}, \bar{\alpha}', \beta, \bar{\beta} \in V^*$; $\bar{A}, \bar{A}' \in N$, and $y, z \in \Sigma^*$ such that

(i) $S \Rightarrow_R^* \alpha_m x_1 = \bar{\alpha}\bar{A}yx_1 \Rightarrow_R \bar{\alpha}\bar{\beta}yx_1 = \alpha_{m-1}x_1$,

(ii) $S \Rightarrow_R^* \alpha'_m = \bar{\alpha}'\bar{A}'z \Rightarrow_R \bar{\alpha}'\bar{\beta}'z = \alpha'_{m-1} = \alpha_{m-1}x_2 = \bar{\alpha}\bar{\beta}yx_2$,

using the fact that $\alpha'_{m-1} = \alpha_{m-1}x_2$ from our minimality assumption about m.

[2] In most cases, we will use the fact that $x_2 \neq \Lambda$. The other possibilities are useful in studying other definitions of LR(k) grammars.

Now let $\gamma = \bar{\alpha}\bar{\beta}$. We get

(i) $S \overset{*}{\underset{R}{\Rightarrow}} \bar{\alpha}\bar{A}yx_1 \underset{R}{\Rightarrow} \bar{\alpha}\bar{\beta}yx_1 = \gamma yx_1$,

(ii) $S \overset{*}{\underset{R}{\Rightarrow}} \bar{\alpha}'\bar{A}'z \underset{R}{\Rightarrow} \bar{\alpha}'\bar{\beta}'z = \gamma yx_2$.

Now ${}^{(k)}x_1 = {}^{(k)}x_2$ implies ${}^{(k)}yx_1 = {}^{(k)}yx_2$. Since G is LR(k), we have

$$(\bar{A} \to \bar{\beta}, \lg(\bar{\alpha}\bar{\beta})) = (\bar{A}' \to \bar{\beta}', \lg(\bar{\alpha}'\bar{\beta}')).$$

From (ii) $\alpha'_m = \bar{\alpha}'\bar{A}'z$, and using the equality of handles, we have $\bar{\beta} = \bar{\beta}'$, $\bar{A} = \bar{A}'$, and thus $\bar{\alpha}' = \bar{\alpha}$ and $z = yx_2$. Thus $\alpha'_m = \bar{\alpha}\bar{A}yx_2 = \alpha_m x_2$. But this contradicts our assumption that $\alpha'_m \neq \alpha_m x_2$. □

Now we examine other definitions for LR(k) grammars which have been used in the literature. There are two definitions which entail extending the original grammar by adding an "initial production". The first is the definition used in [1]. The second involves adding an endmarker [15]. Finally, we examine a definition given in [17] that has been used in work on topdown parsing. For each of these three definitions, we shall examine the classes of grammars and languages generated by these definitions. It turns out that these investigations usually require that we discuss the cases $k > 0$ and $k = 0$ separately. We shall conclude this section with a chart of the relationships between the various classes of grammars and languages.

The original definition of LR(k) grammars [15] differed from Definition 2.1 in not excluding derivations of the form $S \overset{+}{\underset{R}{\Rightarrow}} S$. That definition allowed ambiguous grammars like

$$S \to S \mid a$$

to be called LR(0). Salomaa [18] noted this and excluded $S \to S$ as a rule from his definition of LR(k) grammars. But as Graham pointed out, that did not solve the problem as grammars like

$$S \to A, \qquad A \to S \mid a,$$

satisfied the new definition and were still ambiguous. Clearly the ambiguity problem can be disposed of forever by excluding all derivations of the form

$$S \overset{+}{\underset{R}{\Rightarrow}} S.$$

LR(k) grammars are defined in [1] by adding a production $S' \to S$ to the original grammar. The purpose of adding a production $S' \to S$ to the grammar was to simplify the termination condition of the parsers for grammars in this class, and to insure unambiguity. By using this definition, the parser will halt in an accept state if and only if this reduction to S' is performed. The same effect might have been achieved by not allowing an S on the right hand side of any production rule in an LR(k) grammar. In this way, a reduction to S would signify an accept state or an error condition. We shall show in the sequel [7] that by slightly altering the termination condition, these restrictions are not necessary.

We now present the definition from [1].

Definition 2.7. Let $k \geqslant 0$ and $G = (V, \Sigma, P, S)$ be a reduced context free grammar. Define the *augmented grammar* $G' = (V', \Sigma, P', S')$ where $V' = V \cup \{S'\}$ and $P' = P \cup \{S' \to S\}$, where S, a symbol not in V, is our new starting symbol. G is said to be $ALR(k)$ (augmented LR(k)) if and only if G' satisfies the body of Definition 2.1.

We will show that the class of LR(k) grammars is at least as large as the class of A LR(k) grammars. Later, we shall show that the inclusion of classes is proper.

Lemma 2.8. *Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar. For each $k \geqslant 0$, if G is $ALR(k)$ then G is $LR(k)$.*

Proof. The argument is straightforward and is omitted. □

In the next result, we show the converse of Lemma 2.8 for all $k \geqslant 1$.

Lemma 2.9. *Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar. If G is $LR(k)$ for some $k \geqslant 1$ then G is $ALR(k)$.*

Proof. The proof is a tedious but simple case analysis in which the canonical sentential forms are examined. Details are omitted. □

Combining the results for grammars leads to

Lemma 2.10. *The classes of $LR(k)$ and $ALR(k)$ grammars are co-extensive for all $k \geqslant 1$.*

Our results show that the class of ALR(k) grammars is contained in the class of LR(k) grammars for $k \geqslant 0$ and we have equality for $k \geqslant 1$. But there remains the possibility that the definition of ALR(0) grammars is more restrictive than the LR(0) definition. This turns out to be the case. We now characterize the ALR(0) grammars in terms of LR(0) grammars.

Lemma 2.11. *If $G = (V, \Sigma, P, S)$ is $LR(0)$ and $S \overset{+}{\underset{R}{\Longrightarrow}} Sw$ is impossible in G for any $w \in \Sigma^+$ then G is $ALR(0)$.*

Proof. Again, the argument is a case study on canonical sentential forms and is omitted. □

The following lemma shows that ALR(0) grammars cannot be left recursive on S.

Lemma 2.12. *Let $G=(V,\Sigma,P,S)$ be an $ALR(0)$ grammar. For any $w\in\Sigma^*$, $S\overset{+}{\underset{R}{\Longrightarrow}}Sw$ is impossible in G.*

Proof. The argument is straightforward. □

The following theorem characterizes the class of ALR(k) grammars and is a summary of the previous lemmas.

Theorem 2.13. *Let $G=(V,\Sigma,P,S)$ be a reduced context free grammar.*

(i) *For $k>0$, G is an $LR(k)$ grammar if and only if G is an $ALR(k)$ grammar.*

(ii) *G is an $ALR(0)$ grammar if and only if G is an $LR(0)$ grammar and $S\overset{+}{\underset{R}{\Longrightarrow}}Sw$ is impossible in G for any $w\in\Sigma^+$.*

Proof. The result follows directly from Lemma 2.10, Lemma 2.11 and Lemma 2.12. □

A concrete example of an LR(0) grammar which is not ALR(0) is

$$S\rightarrow Sa\mid a.$$

We now study the class of ALR(0) languages.

Theorem 2.14. *$L\subseteq\Sigma^*$ is an $ALR(0)$ language if and only if L is strict deterministic.*

Proof. Assume that L is an ALR(0) language. Thus there exists an ALR(0) grammar $G=(V,\Sigma,P,S)$ with $L=L(G)$. Assume for the sake of contradiction that L is not strict deterministic. Since L is deterministic, it must fail to be prefix free. Thus there exist $x\in\Sigma^*$, $y\in\Sigma^+$ such that

(i) $S\overset{*}{\underset{R}{\Longrightarrow}}S\overset{+}{\underset{R}{\Longrightarrow}}x$,

(ii) $S\overset{+}{\underset{R}{\Longrightarrow}}xy$,

and

(iii) ${}^0(\Lambda)={}^0(y)=\Lambda$.

Since G is ALR(0), G is LR(0) by Lemma 2.8. By the extended LR(k) theorem (Theorem 2.6), we have

$$S\overset{+}{\underset{R}{\Longrightarrow}}Sy.$$

But this contradicts Theorem 2.13. Thus L is strict deterministic.

Conversely, assume that L is a strict deterministic language. Then $L=L(G)$, where $G=(V,\Sigma,P,S)$ is a strict deterministic grammar. By [12] G is LR(0). By [11] G cannot be left recursive. By Theorem 2.13, G is ALR(0) so that L is an ALR(0) language. □

Corollary. *The class of ALR(0) languages is properly contained in the class of LR(0) languages.*

Proof. This is a direct result of the theorem and the fact that there are LR(0) languages (like a^+) which are not prefix free. □

Instead of extending our grammar with the production $S' \rightarrow S$, we might extend our grammar with the production $S' \rightarrow S\$$, where \$ is an endmarker, a symbol not in our original grammar. For $k > 0$, this corresponds to the definition of an LR(k) grammar in [15]. The only difference is that [15] allows for k endmarkers, whereas we only allow one. We have eliminated the necessity for k endmarkers by also defining ${}^{(k)}x$ for $\lg(x) < k$.

The addition of an endmarker makes the termination configuration for the parser even simpler than using the condition from [1]. By adding an endmarker, the termination condition becomes the reading of the endmarker.

We now define \$LR(k) grammars.

Definition 2.15. Let $k \geqslant 0$ and $G = (V, \Sigma, P, S)$ be a reduced context free grammar. Define the *\$-augmented grammar* $G' = (V', \Sigma', P', S')$ where $V' = V \cup \{S', \$\}$, $\Sigma' = \Sigma \cup \{\$\}$, $P' = P \cup \{S' \rightarrow S\$\}$, where S' and \$ are new symbols not in V. G is said to be *\$LR(k)* if and only if G' satisfies the body of Definition 2.1.

For $k \geqslant 1$, the relationship between \$LR(k) and LR(k) grammars is simple.

Theorem 2.16. *Let $G = (V, \Sigma, P, S)$ be a context free grammar. For each $k \geqslant 1$, G is \$LR(k) if and only if G is LR(k).*

Corollary. *For any $k \geqslant 1$, L is a \$LR(k) language if and only if L is an LR(k) language.*

We now show, as in the ALR(0) case, that the class of \$LR(0) grammars is properly contained in the class of LR(0) grammars.

To characterize the class of \$LR(0) grammars, we must first define the notion of a pathological production.

Definition 2.17. Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar. A production $p \in P$ is *pathological* if

(i) there exists a $T \in N$, $w \in \Sigma^*$ such that

$$p = (T \rightarrow S)$$

and

$$S \overset{+}{\underset{R}{\Rightarrow}} Tw \underset{R}{\Rightarrow} Sw$$

or (ii) there exists an $A \in N$, $w \in \Sigma^*$ such that

$$p = (A \to \Lambda)$$

and

$$S \overset{+}{\underset{R}{\Rightarrow}} SAw \underset{R}{\Rightarrow} Sw.$$

Pathological productions can be characterized in the following manner.

Lemma 2.18. *Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar. Then G has no pathological productions if and only if there exists no $w \in \Sigma^*$ such that Sw is a sentential form of G with a handle whose second component is* 1.

Proof. The argument is quite easy and is omitted. □

Now pathological productions can be related to \$LR(0) grammars.

Lemma 2.19. *Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar. Then G is \$LR(0) if and only if it is LR(0) and has no pathological productions.*

Proof. The argument is a more-or-less straightforward application of earlier lemmas and techniques. □

The following theorem characterizes the class of \$LR grammars.

Theorem 2.20. *Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar.*

(i) *For $k > 0$, G is a \$LR(k) grammar if and only if G is an LR(k) grammar.*

(ii) *G is a \$LR(0) grammar if and only if G is an LR(0) grammar and has no pathological production.*

Proof. Follows directly from Theorem 2.16 and Lemma 2.19. □

A concrete example of an LR(0) grammar which is not a \$LR(0) grammar is

$$S \to Ab,$$

$$A \to S \mid b.$$

The following theorem characterizes the class of \$LR(0) languages.

Theorem 2.21. *$L \subseteq \Sigma^*$ is a \$LR(0) language if and only if L is an LR(0) language.*

Proof. If L is a \$LR(0) language it follows easily that L is LR(0) as well.
In the reverse direction, some of the results of Section 3 as well as those in [11]

are used together with a lemma from the present section. The details are omitted. □

It is possible to prove the following result which is stated without proof here. Full details are given in [4]. The result indicates why these special productions are called pathological.

Theorem 2.22. *An LR(0) grammar can have at most one pathological production.*

Finally, we discuss a definition for LR(k) grammars which originated in [17]. The definition is reminiscent of the extended LR(0) theorem, in that it considers derivations of arbitrary length, whereas other definitions of LR(k) are basically concerned with single steps in a derivation sequence. The definition is also unusual in that it considers sentential forms which are not canonical.

An incorrect proof is presented in [16] that the new definition is equivalent to the LR(k) definition of [15]. That is immediately false because of ambiguity considerations but other problems exist as well. We shall further show that even if we eliminate ambiguous grammars from the definition of [15], namely, if we use our LR(k) definition, that the LR(k) and LLR(k) classes of grammars still do not correspond.

We begin by giving the definition from [17].

Definition 2.23. Let $G = (V, \Sigma, P, S)$ be a reduced context free grammar. Then G is *LLR*(k) if and only if

(a) G is unambiguous and,

(b) for all $w_1, w_2, w_3, w_3' \in \Sigma^*$, $A \in N$, if

(i) $S \Longrightarrow^* w_1Aw_3 \Longrightarrow^* w_1w_2w_3$,

(ii) $S \Longrightarrow^* w_1w_2w_3'$,

and

(iii) ${}^{(k)}w_3 = {}^{(k)}w_3'$,

then

(iv) $S \Longrightarrow^* w_1Aw_3'$.

We first show that if a grammar is \$LR(k), it is also LLR(k).

Theorem 2.24. *Let $G = (V, \Sigma, P, S)$ be a \$LR(k) grammar for some $k \geq 0$. Then G is LLR(k).*

Proof. We assume that G is an LR(k) grammar where $k \geq 0$. Thus G is unambiguous. We assume that for all $w_1, w_2, w_3, w_3' \in \Sigma^*$, $A \in N$:

(i) $S \Longrightarrow^* w_1Aw_3 \Longrightarrow^* w_1w_2w_3$,

(ii) $S \Longrightarrow^* w_1w_2w_3'$,

(iii) ${}^{(k)}w_3 = {}^{(k)}w_3'$.

By Lemma 2.18 and Lemma 2.19 we have

(iv) (a) $k \geqslant 1$ or

(b) $k = 0$ and there exists no $x \in \Sigma^*$ such that S is a sentential form in G whose second component is 1.

By (i) there exists an $\alpha \in V^*$ such that

(i′) $S \xRightarrow{*}_R \alpha A w_3 \xRightarrow{+}_R w_1 w_2 w_3$,

where $\alpha \xRightarrow{*}_R w_1$ and $A \xRightarrow{+}_R w_2$, (i′), (ii), (iii), (iv) and the extended LR(k) theorem give us

(v) $S \xRightarrow{*}_R \alpha A w_3'$.

(i′) gives us $\alpha A w_3' \xRightarrow{*}_R w_1 A w_3'$, so (i′) and (v) give us $S \Rightarrow^* w_1 A w_3'$. Thus G is LLR(k). □

We cannot show that every LR(0) grammar is LLR(0), in fact, we can later give a counterexample to this statement. However, it follows immediately from Theorem 2.23 that every LR(0) grammar is LLR(1).

Corollary 2.24.1. *Let $G = (V, \Sigma, P, S)$ be an LR(0) grammar. Then G is LLR(1).*

Proof. The proof is immediate. □

For $k \geqslant 1$, we can show that any LR(k) grammar is an LLR(k) grammar.

Corollary 2.24.2. *Let $G = (V, \Sigma, P, S)$ be an LR(k) grammar where $k \geqslant 1$. Then G is an LLR(k) grammar.*

Proof. Since G is an LR(k) grammar, where $k \geqslant 1$, by Theorem 2.20 G is a \$LR(k) grammar. It now follows from Theorem 2.24 that G is an LLR(k) grammar. □

It is also false that all LLR(0) grammars are LR(0) grammars. We can, in fact, show that for any $k \geqslant 0$, there exists a grammar which is LLR(0) but not LR(k).

Theorem 2.25. *For any $k \geqslant 0$, there exists a grammar which is LLR(0) but not LR(k).*

Proof. Consider the grammar

$$S \to aAb^k c$$

$$S \to aAAb^k d$$

$$A \to \Lambda$$

where $k \geqslant 0$. This grammar is clearly not LR(k) but is LLR(0). □

If we limit ourselves to Λ-free grammars, the class of \$LR(k) grammars is the same as the class of LLR(k) grammars for any $k \geq 0$.

Theorem 2.26. *Let $G = (V, \Sigma, P, S)$ be a reduced context free Λ-free grammar. Then G is \$LR(k) if and only if G is LLR(k).*

Proof. Assume that G is \$LR(k). Then by Theorem 2.24, G is LLR(k). □

Conversely, assume that G is LLR(k). We first show that G is LR(k). Since G is unambiguous, $S \Rightarrow_R^+ S$ is impossible in G. Assume for the sake of contradiction that G is not LR(k). By Lemma 2.5, we have

There exist $w_3, w_3', x \in \Sigma^*$, $\gamma, \alpha, \alpha', \beta, \beta' \in V^*$, $A, A' \in N$ such that:

(i) $S \Rightarrow_R^* \alpha A w_3 \Rightarrow_R \alpha\beta w_3 = \gamma w_3$,

(ii) $S \Rightarrow_R^* \alpha' A' x \Rightarrow_R \alpha'\beta' x = \gamma w_3'$,

(iii) ${}^{(k)}w_3 = {}^{(k)}w_3'$,

(iv) $(A \to \beta, \lg(\alpha\beta)) \neq (A' \to \beta', \lg(\alpha'\beta'))$,

(v) $\lg(\alpha'\beta') \geq \lg(\alpha\beta)$.

Since G is reduced and Λ-free, for some $w_1 \in \Sigma^*$, $w_2 \in \Sigma^+$,

(vi) $\alpha \Rightarrow_R^* w_1$ and $\beta \Rightarrow_R^* w_2$.

Thus $\gamma \Rightarrow^* w_1 w_2$. From (i) it follows that

(i') $S \Rightarrow^* w_1 A w_3 \Rightarrow^* w_1 w_2 w_3$.

From (ii) it follows that

(ii') $S \Rightarrow^* w_1 w_2 w_3'$.

Since G is LLR(k), (i), (i'), (ii') and (iii) give us

$$S \overset{*}{\Rightarrow} w_1 A w_3' \overset{*}{\Rightarrow} w_1 w_2 w_3'.$$

It follows that for some $\alpha'' \in V^*$, $\beta'' \in V^+$

$$S \underset{R}{\overset{*}{\Rightarrow}} \alpha'' A w_3' \underset{R}{\Rightarrow} \alpha''\beta'' w_3' \underset{R}{\overset{*}{\Rightarrow}} \alpha'' w_2 w_3' \underset{R}{\overset{*}{\Rightarrow}} w_1 w_2 w_3'. \tag{1}$$

We now consider this derivation and derivation (ii), namely

$$S \underset{R}{\overset{*}{\Rightarrow}} \alpha' A' x \underset{R}{\Rightarrow} \alpha'\beta' x = \alpha\beta w_3' \underset{R}{\overset{*}{\Rightarrow}} \alpha w_2 w_3' \underset{R}{\overset{*}{\Rightarrow}} w_1 w_2 w_3'. \tag{2}$$

Since G is unambiguous, each step in derivations (1) and (2) must correspond.

We now consider three cases, corresponding to when $\alpha\beta w_3' = \alpha''\beta'' w_3'$ when $\alpha\beta w_3'$ precedes $\alpha''\beta'' w_3'$ in the unique derivation of $w_1 w_2 w_3'$ and when $\alpha''\beta'' w_3'$ precedes $\alpha\beta w_3'$ in this derivation.

Case 1. $\alpha\beta w_3' = \alpha''\beta'' w_3'$. It follows immediately that $\alpha\beta = \alpha''\beta''$. We shall show that $\alpha = \alpha''$ and $\beta = \beta''$. Assume for the sake of contradiction that $\alpha \neq \alpha''$. Without loss of generality, assume that α is a proper prefix of α''. Thus, for some $\bar{\alpha} \in V^+$, $\alpha'' = \alpha\bar{\alpha}$. Now, we have rightmost sentential forms $\alpha''\beta'' w_3' = \alpha\bar{\alpha}\beta'' w_3'$.

Now, consider the unique tree in which $\alpha\bar{\alpha}\beta'' w_3' \Rightarrow^* w_1 w_2 w_3$ (see Fig. 1).

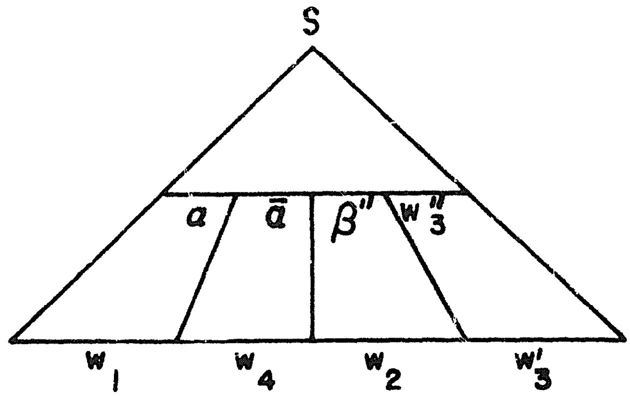

Fig. 1. A derivation tree for case 1.

By eq. 2, we have $\alpha \Rightarrow^* w_1$, and by eq. 1 $\beta'' \Rightarrow^* w_2$. Since $\bar{\alpha} \neq \Lambda$, and G is Λ-free, for some $w_4 \in \Sigma^+$, we have $\bar{\alpha} \Rightarrow^* w_4$ in this unique tree. Therefore $w_1 w_4 w_2 w_3' = w_1 w_2 w_3$ which is clearly false. Therefore $\alpha = \alpha''$ and $\beta = \beta''$. Since $\alpha\beta w_3' = \alpha''\beta'' w_3'$, the predecessors of these canonical sentential forms must be equal, therefore $\alpha'' A w_3' = \alpha' A' x$. Since A and A' are the rightmost variables in these equal canonical sentential forms, respectively, we have $\alpha'' = \alpha'$, $A = A'$, and $x = w_3'$. Thus $\alpha = \alpha'' = \alpha'$. Since $\alpha\beta = \alpha'\beta'$, we have $\alpha'\beta = \alpha'\beta'$, thus $\beta = \beta'$. Therefore $(A \to \beta, \lg(\alpha\beta)) = (A' \to \beta', \lg(\alpha'\beta'))$, contradicting (iv).

Case 2. $\alpha\beta w_3'$ precedes $\alpha''\beta'' w_3'$. That is

$$S \overset{*}{\underset{R}{\Rightarrow}} \alpha\beta w_3' \overset{*}{\underset{R}{\Rightarrow}} \alpha'' A w_3' \underset{R}{\Rightarrow} \alpha''\beta'' w_3' \overset{*}{\underset{R}{\Rightarrow}} \alpha'' w_2 w_3' \overset{*}{\underset{R}{\Rightarrow}} w_1 w_2 w_3'. \quad (3)$$

Since we have no Λ-rules, $\beta \neq \Lambda$. Since A is the rightmost variable in the sentential form $\alpha'' A w_3'$, we must have $\beta^{(1)} \in N$. Since $\beta^{(1)} \in N$, α cannot be changed in the production sequence $\alpha\beta w_3' \Rightarrow_R^* \alpha''\beta'' w_3'$, since A is the rightmost non-terminal in the canonical sentential form $\alpha'' A w_3'$, and thus must derive from $\beta^{(1)}$. Therefore, for some $\beta''' \in V^+$, we can write

$$\alpha''\beta'' = \alpha\beta'''.$$

By an argument identical to Case 1, we can show that $\alpha = \alpha''$. Therefore from eq. 3

$$S \overset{*}{\underset{R}{\Rightarrow}} \alpha A w_3' \underset{R}{\Rightarrow} \alpha\beta'' w_3' \overset{*}{\underset{R}{\Rightarrow}} w_1 w_2 w_3'.$$

Using (i) and (vi) and the above derivation in G we can have

$$S \overset{*}{\underset{R}{\Rightarrow}} \alpha A w_3' \underset{R}{\Rightarrow} \alpha\beta w_3' \overset{*}{\underset{R}{\Rightarrow}} \alpha w_2 w_3' \overset{*}{\underset{R}{\Rightarrow}} w_1 w_2 w_3'.$$

Since G is unambiguous, $\beta = \beta''$. It follows from (3) that

$$S \overset{*}{\Longrightarrow} \alpha\beta w_3' \underset{R}{\overset{+}{\Longrightarrow}} \alpha\beta w_3' \underset{R}{\overset{*}{\Longrightarrow}} w_1 w_2 w_3'.$$

This contradicts the fact that G is unambiguous.

Case 3. $\alpha''\beta''w_3'$ precedes $\alpha\beta w_3'$. Here, techniques similar to those used in the first two case are used to reach a contradiction. The details are omitted.

Since Cases 1, 2 and 3 were contradicted, our assumption that G is not LR(k) is contradicted. Thus, G must be LR(k).

We now show that G is \$LR(k). If $k > 0$ then G is \$LR(k) by Theorem 2.20. Suppose $k = 0$. Suppose G is not \$LR(0). Since G is LR(0), G is unambiguous. By Lemma 2.19, G has a pathological production. We have two cases.

Case 1. There exists a $T \in N$, $S \neq T$, $w \in \Sigma^+$ such that $\rho = (T \to S)$ and $S \Longrightarrow_R^+ Tw \Longrightarrow_R Sw$. Choose any $w' \in L(G)$. Then

(i) $S \Longrightarrow_R^+ Tw \Longrightarrow_R Sw \Longrightarrow_R^+ w'w$.

Also

(ii) $S \Longrightarrow_R^* w'$,

and

(iii) ${}^{(0)}w = {}^{(0)}\Lambda = \Lambda$.

Therefore, since G is LLR(0), we have

(iv) $S \Longrightarrow^* T \Longrightarrow^* w'$.

It follows from (iv) that $S \Longrightarrow^+ T$ and from (i) that $T \Longrightarrow^+ S$. Therefore $S \Longrightarrow^+ S$ in G.

This, however, gives us that G is ambiguous, which is a contradiction.

Case 2. There exists an $A \in N$, $w \in \Sigma^+$ such that $\rho = (A \to \Lambda)$ and $S \Longrightarrow_R^+ SAw \Longrightarrow_R Sw$. Again, choose any $w' \in L(G)$. Then

(i) $S \Longrightarrow_R^+ SAw \Longrightarrow_R Sw \Longrightarrow_R^+ w'w$.

Also

(ii) $S \Longrightarrow_R^+ w'$,

and

(iii) ${}^0(w) = {}^{(0)}w' = \Lambda$.

Therefore, since G is LLR(0), we have

(iv) $S \Longrightarrow^* SA \Longrightarrow^* w'$.

Clearly from (iv), $S \Longrightarrow^+ SA \Longrightarrow S$. Therefore $S \Longrightarrow^+ S$ in G, which contradicts the unambiguity of G. $\square$

We can now show that there exist grammars which are LR(0) but not LLR(0).

Corollary. *There exists a grammar which is LR(0) but not LLR(0).*

Proof. Consider the grammar with productions

$$S \to Aa \mid a,$$

$$A \to S.$$

This grammar is LR(0) and Λ-free, but not \$LR(0) by Theorem 2.20 since it has pathological production $A \rightarrow S$. Therefore, by Theorem 2.26, G is not LLR(0). □

We now study the class of LLR(k) languages. We have shown that without Λ-rules, if a grammar is LLR(k) then it is LR(k). We shall show the equality of language classes by providing a transformation for eliminating Λ-rules from LLR(k) grammars.

Lemma 2.27. *Let $G = (V, \Sigma, P, S)$ be an LLR(k) grammar, $k \geq 0$. Then there exists an LLR(k), Λ-free grammar G' such that $L(G') = L(G) - \{\Lambda\}$.*

Proof. Let $G = (V, \Sigma, P, S)$ be an LLR(k) grammar, $k \geq 0$. We use Theorem 1.8.1 of [8] for eliminating Λ-rules from a grammar. Let the resultant grammar be $G' = (V', \Sigma, P', S')$. The following claim follows from the construction.

Claim. *$\alpha \in V'^*$ is a sentential form in G' if and only if α is a sentential form in G and each of the non-terminals in α produces something other than Λ in G.*

By [8], $L(G') = L(G) - \{\Lambda\}$. We now show that G' is LLR(k).

(a) By [8], G' is unambiguous since G is unambiguous.

(b) Assume that in G' for $w_1, w_3, w_3' \in \Sigma^*$, $w_2 \in \Sigma^+$, $A \in N'$

(i) $S \Rightarrow^* w_1 A w_3 \Rightarrow^* w_1 w_2 w_3$,

(ii) $S \Rightarrow^* w_1 w_2 w_3'$,

(iii) ${}^{(k)}w_3 = {}^{(k)}w_3'$.

By our claim, in G we also have (i), (ii) and (iii). Since G is LLR(k), we have

(iv) $S \Rightarrow^* w_1 A w_3'$ in G.

Since in G, $A \Rightarrow^* w_2$, by our claim $w_1 A w_3'$ is a sentential form in G'. Therefore in G', $S \Rightarrow^* w_1 A w_3'$. Thus, G' is LLR(k). □

The lemma now helps us show that by eliminating Λ-rules, we can get an equivalent LR(1) grammar for any LLR(k) grammar.

Lemma 2.28. *Let $G = (V, \Sigma, P, S)$ be an LLR(k) grammar, $k \geq 0$. Then there exists an LR(1) grammar G' such that $L(G) = L(G')$.*

Proof. Let G be an LLR(k) grammar, $k \geq 0$. By Lemma 2.27, there exists an LLR(k), Λ-free grammar G''' such that $L(G''') = L(G) - \{\Lambda\}$. By Theorem 2.26, G''' is LR(k). By [11], there exists some LR(1) grammar $G'' = (V, \Sigma, P, S)$ such that $L(G'') = L(G''')$. If $\Lambda \notin L(G'')$, we let $G' = G''$. Otherwise, we let $G' = (V \cup \{S'\}, \Sigma, P \cup \{S' \rightarrow S, S' \rightarrow \Lambda\}, S')$ where S' is a new symbol not in V. Clearly $L(G') = L(G)$, and we can easily show that G' is an LR(1) grammar. □

It follows directly from this lemma that all LLR languages are deterministic.

Theorem 2.29. *$L \subseteq \Sigma^*$ is a deterministic language if and only if L is an LLR language.*

Proof. The result follows directly from Theorem 2.24, Lemma 2.28 and the fact that the class of LR(1) grammars generates the deterministic languages. Cf. [12]. □

Finally we study the class of LLR(0) languages.

Lemma 2.30. *If $L \subseteq \Sigma^*$ is an LLR(0) language, then L is an LR(0) language.*

Proof. Assume that $L = L(G)$, where $G = (V, \Sigma, P, S)$ is an LLR(0) grammar. By Theorem 2.29, L is a deterministic language. We next assume for some $x \in \Sigma^+$, $w, y \in \Sigma^*$, $w \in L$, $wx \in L$, $y \in L$ that we have

(i) $S \Rightarrow^* S \Rightarrow^* w$,

(ii) $S \Rightarrow^* wx$,

and

(iii) ${}^{(0)}(\Lambda) = {}^{(0)}x = \Lambda$.

Since G is LLR(0), (i)–(iii) give us

(iv) $S \Rightarrow^* Sx$.

Thus $S \Rightarrow^* Sx \Rightarrow^* yx$. Therefore $yx \in L$. Thus L satisfies condition (b) for LR(0) languages of the LR(0) characterization theorem. Thus L is LR(0). □

Theorem 2.31. *$L \subseteq \Sigma^*$ is an LR(0) language if and only if L is an LLR(0) language.*

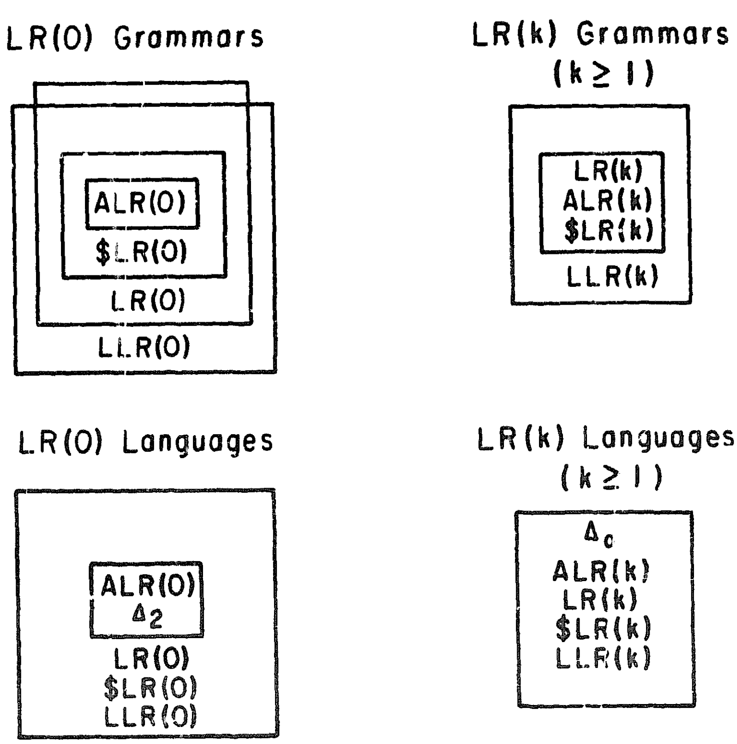

Fig. 2. Comparison of classes of languages and grammars.

Proof. Let $L \subseteq \Sigma^*$ be an LR(0) language. By Theorem 2.21, L is a \$LR(0) languge. It follows from Theorem 2.24 that L is an LLR(0) language.

The converse is Lemma 2.30. □

One may summarize the results about comparison of these classes of grammars graphically. This is done in Fig. 2.

3. Properties of LR(0) languages

In this section, we shall study the class of LR(0) languages. We begin with the main theorem of the section, the LR(0) language characterization theorem. This theorem gives a string characterization, a machine characterization, and a set-theoretic characterization of the class of LR(0) languages. This theorem has proved to be a very valuable result. Not only is it used extensively in later proofs, but condition (b) allows us to assert that certain sets are LR(0) languages by inspection. We conclude the section by showing that a well-known language is LR(0) using the characterization theorem.

Theorem 3.1. (LR(0) language characterization theorem). *Let $L \subseteq \Sigma^*$. The following four statements are equivalent.*

(a) *L is an $LR(0)$ language.*

(b) *$L \subseteq \Sigma^*$ is a deterministic context free language and for all $x \in \Sigma^+$, $w, y \in \Sigma^*$ if $w \in L$, $wx \in L$, and $y \in L$ then $yx \in L$.*

(c) *There exists a DPDA $A = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$ where $F = \{q_f\}$ and there exists $Z_f \in \Gamma$ such that*

$$L = T(A, Z_f) = T(A, \Gamma) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \overset{*}{\vdash} (q_f, \Lambda, Z_f)\}.$$

(d) *There exist strict deterministic languages L_0 and L_1 such that $L = L_0 L_1^*$.*

Proof. We first prove that (a) implies (b). We assume that $G = (V, \Sigma, P, S)$ is an LR(0) grammar, and $L = L(G)$. We assume that for $w \in \Sigma^*$, $x \in \Sigma^+$, we have $w \in L$ and $wx \in L$. Thus, we have in G the derivations

(i) $S \underset{R}{\overset{*}{\Longrightarrow}} S \underset{R}{\overset{+}{\Longrightarrow}} w$,

(ii) $S \underset{R}{\overset{*}{\Longrightarrow}} wx$,

(iii) ${}^{(0)}\Lambda = {}^{(0)}x = \Lambda$,

(iv) $x \neq \Lambda$.

By the Extended LR(0) Theorem (2.6), we get

(v) $S \underset{R}{\overset{*}{\Longrightarrow}} Sx \underset{R}{\overset{+}{\Longrightarrow}} wx$.

Now we assume that for some $y \in \Sigma^*$, we have $y \in L$. Thus, we have $S \underset{R}{\overset{+}{\Longrightarrow}} y$. (v) gives us

$$S \underset{R}{\overset{*}{\Rightarrow}} Sx \underset{R}{\overset{+}{\Rightarrow}} yx.$$

Thus $yx \in L$ which completes the proof that (a) implies (b).

We now prove that (b) implies (c). This the most involved part of the proof of this theorem, since several machine constructions are involved. We begin by considering the degenerate[3] languages that obey (b), namely $\emptyset$ and $\{\Lambda\}$. We then consider the prefix free languages that satisfy (b). We then consider the languages that satisfy (b) that are not prefix free.

If either $L = \emptyset$ or $L = \{\Lambda\}$, it is easy to construct a DPDA A such that $L = T(A, Z_f)$ for some stack symbol Z_f.

Now, we shall operate under the assumption that L is not degenerate. We know that there exists a DPDA $A = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$ such that $L = T(A)$, since L is deterministic. We wish to modify A so that we obtain a new DPDA such that L is accepted by a new machine with only Z_f on the pushdown. Our first step is to add a new "bottom of stack" marker to our machine. We let

$$A' = \langle Q', \Sigma, \Gamma', \delta', q_0', Z_b, F'\rangle$$

have a new start state which adds a new bottom of stack marker to the pushdown and then simulates A.

We now choose any $x \in \Sigma^*$ such that $x \in \min(L)$. Since $L \neq \emptyset$, clearly $\min(L) \neq \emptyset$. We shall now consider two cases. The first case will correspond to the strict deterministic languages. Observe carefully, however, that this does not follow directly from the statement of this case. Our construction will be much simpler in this case, than in case two, where our language is not prefix free.

Case 1. Choose any $x \in \Sigma^*$ such that $x \in \min(L)$. For all $y \in \Sigma^+$, $xy \notin L$.

Claim. *L is strict deterministic.*

Proof. Assume for the sake of contradiction that L is not strict deterministic. Then there exist $w \in \Sigma^*$, $y \in \Sigma^+$ such that $w \in L$ and $wy \in L$ since L is not prefix free. Since $x \in L$, $xy \in L$ by (b). But this contradicts the supposition for Case 1, giving us a contradiction. Thus L is strict deterministic.

We shall construct a machine A'' that emulates the machine A' until a final state of A' is reached. It then erases the stack until a new bottom of stack marker is reached, and then goes to a special final state and puts the special accept symbol on the pushdown. It is not difficult to construct A'' such that

$$T(A'', \Gamma'') = T(A'', Z_f) = T(A') = T(A) = L.$$

This completes the proof of Case 1.

Case 2. Let $x \in \Sigma^*$ be any string such that $x \in \min(L)$ and for some $y \in \Sigma^+$,

[3] A language $L \subseteq \Sigma^*$ is *degenerate* if $L = \emptyset$ or $L = \{\Lambda\}$.

$xy \in L$. In this case, our machine A' cannot go to a dead configuration after reaching an accepting configuration. We shall construct a new machine A'', which, after accepting any string by $T(A'', Z_f)$, will pretend that it is in fact x that it has just accepted. Our first claim shows us how our machine will pretend that it has just accepted x. Our claim concerns the behavior of A' under the assumption of Case 2.

Claim. *There exists a $\bar{q} \in Q'$, $\bar{\alpha} \in (\Gamma')^*$, $\bar{Z} \in \Gamma'$ such that for our chosen x*

$$(q_0', x, Z_b) \overset{*}{\vdash} (\bar{q}, \Lambda, Z_b\bar{\alpha}\bar{Z})$$

where for some $\bar{a} \in \Sigma$, $\delta'(\bar{q}, \bar{a}, \bar{Z})$ is defined.

Proof. This follows directly from the hypothesis of Case 2. □

We shall omit the formal construction of A'' from A but will describe it. One adds to A' a new final state and a new stack symbol Z_f to be used for acceptance. When an A'-accept configuration is reached, the stack is erased until the bottom marker is reached and then we go into an A''-accepting configuration. The machine pretends that x has just been accepted and so adjusts its stack. The computation proceeds under the assumption that is is in fact x that has just been accepted. Otherwise A'' proceeds as A'.

It is not hard to see that A'' is a DPDA and that

$$L = T(A) = T(A') = T(A'', \Gamma) = T(A'', Z_f).$$

This completes the argument that (b) implies (c).

To help prove (c) implies (d), we first show that (c) implies (b). Assume there exists a DPDA $A = \langle Q, \Sigma, \Gamma, \delta, q_0, Z, F\rangle$ where $F = \{q_f\}$ and there exists $Z_f \in \Gamma$ such that $L = T(A, Z_f) = T(A, \Gamma)$. Clearly $L \in \Delta_1$. By [11], $L \in \Delta_0$. Suppose that for $x \in \Sigma^+$; $w, y \in \Sigma^*$, we have $w \in L$, $wx \in L$, and $y \in L$. Then we have

$$(q_0, wx, Z_0) \overset{*}{\vdash} (q_f, x, Z_f) \overset{*}{\vdash} (q_f, \Lambda, Z_f)$$

and

$$(q_0, y, Z_0) \overset{*}{\vdash} (q_f, \Lambda, Z_f).$$

Therefore

$$(q_0, yx, Z_0) \overset{*}{\vdash} (q_f, x, Z_f) \overset{*}{\vdash} (q_f, \Lambda, Z_f).$$

Thus $yx \in L$. This completes the proof that (c) implies (b).

We now prove that (c) implies (d). We assume that there exists a DPDA $A = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$ where $F = \{q_f\}$ and $L = T(A, \Gamma) = T(A, Z_f)$ for some $Z_f \in \Gamma$. Let $L_0 = \min(L)$. By [9] L_0 is deterministic. By the definition of min, L is prefix-free. Thus by Theorem 4.2 of [11], L_0 is strict deterministic. We now consider two cases, when $L = L_0$ and when $L \neq L_0$.

Case 1. $L = L_0$. Since L_0 is strict deterministic, $L = L_0(\emptyset)^*$.

Case 2. $L \neq L_0$. Since $L \neq L_0$, there exist $x \in \Sigma^*$, $z \in \Sigma^+$ such that $x \in L$ and

$xz \in L$. Let $L' = \{y \in \Sigma^* \mid xy \in L\}$. We shall show that L' is deterministic, and that $L = L_0L'$.

We now construct a DPDA to accept L'. Let $A' = \langle Q, \Sigma, \Gamma, \delta, q_f, Z_f, \{q_f\}\rangle$, where $A = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$ as just defined. Clearly A' is deterministic.

Claim. $L = L_0L'$.

Proof. We first show that $L \subseteq L_0L'$. Suppose that for some $w \in \Sigma^*$, that $w \in L$. Then for some $w_0, w_1 \in \Sigma^*$, $w = w_0w_1$, where $w_0 \in L_0 = \min(L)$.

If $w_1 = \Lambda$, clearly $w_1 \in L'$, thus $w \in L_0L'$.

Suppose $w_1 \neq \Lambda$. Since $w_0 \in L$, $w_0w_1 \in L$, and $x \in L$, we know $xw_1 \in L$ by characterization (b) of LR(0) languages. Recall that we are assuming (c) and (c) implies (b). Thus $w_1 \in L'$.

Conversely, we show that $L_0L' \subseteq L$.

Suppose that for some $w \in \Sigma^*$, $w \in L_0L'$. Then for some $w_0, w_1 \in \Sigma^*$, we have $w = w_0w_1$ where $w_0 \in L_0$ and $w_1 \in L'$. Since $w_1 \in L'$, we know $xw_1 \in L$. Since $x \in L$, $xw_1 \in L$, and $w_0 \in L$, we know $w = w_0w_1 \in L$ by characterization (b) of LR(0) languages. Thus $L_0L_1 \subseteq L$ and therefore we see that $L = L_0L'$.

Now, let $L_1 = \min(L' - \{\Lambda\})$. Clearly L_1 is strict deterministic, since L' is deterministic

Claim. $L' = L_1^*$.

Proof. We first show $L' \subseteq L_1^*$. For some $w \in \Sigma^*$, assume $w \in L'$. Suppose $w = \Lambda$. Then clearly $w \in L_1^*$. Assume $w \in \Sigma^+$. Then there exist $n \geq 1$, $w_i \in \Sigma^+$, for $1 \leq i \leq n$, where $w = w_1 \cdots w_n$ such that in machine A',

$$(q_f, w_1 \cdots w_n, Z_f) \overset{*}{\vdash} (q_f, w_2 \cdots w_n, Z_f) \overset{+}{\vdash} (q_f, w_3 \cdots w_n, Z_f) \overset{+}{\vdash} \cdots \overset{+}{\vdash} (q_f, \Lambda, Z_f)$$

where these are the only instances in which the machine A' goes through state q_f.

Thus for $1 \leq i \leq n$

$$(q_f, w_i, Z_f) \overset{+}{\vdash} (q_f, \Lambda, Z_f).$$

Thus $w_i \in L_1$. Thus $w \in L_1^*$

We now show that $L_1^* \subseteq L'$. For some $w \in \Sigma^*$, assume $w \in L_1^*$. If $w = \Lambda$, clearly $w \in L'$, by definition of L'. Assume $w \neq \Lambda$. Since $w \in L_1^*$, there exist $n \geq 1$, $w_i \in \Sigma^*$ such that $w_i \in L$ for $1 \leq i \leq n$, where $w = w_1 \cdots w_n$. We now have in A'

$$(q_f, w_1 \cdots w_n, Z_f) \overset{*}{\vdash} (q_f, w_2 \cdots w_n, Z_f) \overset{*}{\vdash} \cdots \overset{*}{\vdash} (q_f, \Lambda, Z_f).$$

This gives us that

$$w = w_1 \cdots w_n \in L'.$$

Therefore

$$L' = L_1^*.$$

Thus, we have $L = L_0L_1^*$ with $L_0, L_1 \in \Delta_2$. This completes our proof that (c) implies (d).

Finally, we show that (d) implies (a). We assume that

$$L = L_0L_1^* \quad \text{where } L_0, L_1 \in \Delta_2.$$

We first consider the degenerate cases. Suppose $L_0 = \emptyset$. Then $L = \emptyset$ and clearly is an LR(0) language. Suppose $L_1 = \emptyset$ or $\{\Lambda\}$. Then $L = L_0$. Since L_0 is a strict deterministic language, L must be an LR(0) language, cf. [11].

Now, we handle the non-degenerate cases. We assume that

$$L_0, L_1 \neq \emptyset, \qquad L_1 \neq \{\Lambda\}.$$

Since L_i is a strict deterministic language, there exist strict deterministic, thus LR(0) grammars $G_i = (V_i, \Sigma_i, P_i, S_i)$ such that $L_i = L(G_i)$, for $i = 0, 1$, with $N_0 \cap N_1 = \emptyset$.

Let $G = (V, \Sigma, P, S)$ where $V = V_0 \cup V_1 \cup \{S\}$, $S \cap \{V_0 \cup V_1\} = \emptyset$, $P = P_0 \cup P_1 \cup \{S \to SS_1, S \to S_0\}$, $\Sigma = \Sigma_0 \cup \Sigma_1$. Clearly $L(G) = L$, since G lays down one word of L_0 followed by a series of strings of any length of words of L_1. We need only show that G is LR(0). Since $L_1 \neq \{\Lambda\}$, and $L_1 \in \Delta_2$, we know $\Lambda \notin L_1$ by [11].

We assume now for the sake of contradiction that G is not an LR(0) grammar. Then by using Lemma 2.5 and a tedious case analysis we arrive at a contradiction. More details can be found in [4]. □

Condition (b) of the LR(0) characterization will prove most useful in checking whether or not a language is LR(0). The following corollary to characterization (b) will be particularly useful.

Corollary. *Suppose $L \subset \Sigma^*$ is an LR(0) language. For $w \in \Sigma^*$, $x \in \Sigma^+$, if $w \in L$ and $wx \in L$ then $wxx \in L$.*

The following theorem shows us that the factorization of an LR(0) language of the form given in the LR(0) language characterization theorem is unique.

Recall that we defined a language $L \subseteq \Sigma^*$ to be *degenerate* if $L = \emptyset$ or $L = \{\Lambda\}$.

Theorem 3.2 (Unique Factorization of LR(0) Languages). *Let $L = L_0L_1^*$ be a nonempty LR(0) language where L_0, L_1 are strict deterministic languages. If there are two strict deterministic languages L_0', L_1' such that $L = L_0'(L_1')^*$ then $L_0 = L_0'$ and either*

(i) $L_1 = L_1'$,

or

(ii) *L_1, L_1' are degenerate.*

Proof. For the sake of contradiction, we assume there exist strict deterministic languages L_0, L_0', L_1, L_1' such that $L_0L_1^* = L_0'L_1'^*$, where $L_0 \neq L_0'$ or $L_1 \neq L_1'$ and L_1 and L_1' are not degenerate.

Case 1 ($L_0 \neq L_0'$). We assume without loss of generality that there exists some $x \in \Sigma^*$ such that $x \in L_0'$ but $x \notin L_0$. This is possible since $L \neq \emptyset$ implies $L_0 \neq \emptyset$. Since $x \in L_0'$, we have $x \in L_0'(L_1')^*$. Thus, $x \in L$ and hence $x \in L_0L_1^*$. Then for some $x_0 \in \Sigma^*$, $x_1 \in \Sigma^+$, we have $x = x_0x_1$ where $x_0 \in L_0$ and $x_1 \in L_1^*$. Since $x_0 \in L_0$, we know $x_0 \in L_0L_1^*$, therefore $x_0 \in L$. Now, we know that $x_0 \in L_0'(L_1')^*$. Clearly $x_0 \notin L_0'$, since x_0 is a proper prefix of x, $x \in L_0'$, and L_0' is prefix free. Thus there exist some $x_2 \in \Sigma^*$, $x_3 \in \Sigma^+$, such that $x_2 \in L_0'$, $x_3 \in L_1'^*$, where $x_0 = x_2x_3$. We also know $x = x_0x_1 = x_2(x_3x_1) \in L_0'$. Since $x_3 \neq \Lambda$, L_0 is not prefix free. But this is a contradiction.

Case 2 ($L_0 = L_0'$). We have $L = L_0L_1^* = L_0L_1'^*$. We assume for the sake of contradiction that $L_1 \neq L_1'$ and we do not have L_1 and L_1' degenerate. Without loss of generality, there exists a $y \in \Sigma^+$ such that $y \in L_1'$ but $y \notin L_1$. Since $L_0 = \emptyset$ there exists some $x \in \Sigma^*$ such that $x \in L_0$. Clearly $xy \in L_0L_1'^*$, giving us $xy \in L_0L_1^*$. Therefore $xy = x'y'$ where $x' \in L_0$ and $y' \in L_1^*$. Clearly x must be a prefix of x' or x' a prefix of x. Since L_0 is prefix free, we must have $x = x'$, and thus $y = y'$. Therefore, we see that $y \in L_1^*$. Now, since $y \notin L_1$, we know there exist $y_0 \in \Sigma^*$, $y_1 \in \Sigma^+$ such that $y = y_0y_1$, where $y_0 \in L_1$. It follows that $xy_0 \in L$ and hence $xy_0 \in L_0L_1'^*$. Thus $xy_0 = x'y_0'$ where $x' \in L_0$ and $y_0' \in L_1'^*$. Again, since L_0 is prefix free, we have $x = x'$ and $y_0 = y_0'$, giving us $y_0 \in L_1'^*$. But, we know $y_0 \notin L_1'$, since if it were we would have $y_0 \in L_1'$ and $y_0y_1 \in L_1'$, where $y_1 \neq \Lambda$, contradicting the fact that L_1' is prefix free. Now, since $y_0 \in L_1'^*$, there exist $y_2 \in \Sigma^*$, $y_3 \in \Sigma^+$ such that $y_0 = y_2y_3$, where $y_2 \in L_1'$. Now we have $y = y_0y_1 = y_2(y_3y_1) \in L_1'$. But $y_3y_1 \neq \Lambda_0$, and $y_2 \in L_1'$. But this contradicts the fact that L_1' is prefix free. $\square$

We conclude this section with an example of the use of the LR(0) characterization theorem to show us immediately that a given language which is not strict deterministic is LR(0). We shall show that the Dyck language is contained in the class of LR(0) languages. We begin by defining a Dyck language.

Definition 3.3. Let $n \geq 1$. D_n is a *Dyck language* if there exists a context free grammar $G_n = (V, \Sigma, P, S)$, where

$$\Sigma = \{a_1, a_2, \ldots, a_n, a_1', \ldots, a_n'\},$$

$$V = \{S\} \cup \Sigma,$$

and

$$P = \{S \to Sa_iSa_i'S \mid 1 \leq i \leq n\} \cup \{S \to \Lambda\},$$

such that

$$D_n = L(G_n).$$

Theorem 3.4. *Let $D_n \subseteq \Sigma^*$ be a Dyck language for some $n \geq 1$. Then D_n is an LR(0) language, but not a strict deterministic language.*

Proof. It is well known that $D_n = D$ is deterministic. For instance, see [10]. Suppose that for $x \in \Sigma^+$, $w, y \in \Sigma^*$ we have $w \in D$, $wx \in D$, and $y \in D$. By [8] we have $x \in D$. Since $y \in D$ and $x \in D$, by [8] we have $yx \in D$. By (b) of the LR(0) language characterization theorem, D is an LR(0) language. Since $a_1a_1' \in D$ and $a_1a_1'a_1a_1' \in D$, D is not strict deterministic since it is not prefix free. $\square$

4. Closure and decidability results for subfamilies of the deterministic languages

In Section 3 we showed the relationship between the classes of LR(0) and strict deterministic languages. In this section, we study Δ_2 (the strict deterministic class of languages), Δ_0 (the deterministic languages), LR(0) (the LR(0) languages), and Δ_1 of [11].[4]

We begin by showing that $\Delta_2 \subsetneq \mathrm{LR}(0) \subsetneq \Delta_1 \subsetneq \Delta_0$. We then study the closure properties of these classes of languages. Our results shall be of the form "class X is (not) preserved under operation Y." This signifies that given a language or languages in class X, after performing operation Y to these languages, the resulting language is or is not a member of class X. These results will then be used in solving certain decidability problems relating to these classes of languages. Finally, we give a chart summarizing the closure properties of the given classes of languages.

We begin by proving, with the aid of two lemmas, that $\Delta_2 \subsetneq \mathrm{LR}(0) \subsetneq \Delta_1 \subsetneq \Delta_0$.

Lemma 4.1. $\Delta_2 \subsetneq LR(0)$.

Proof. We first show that $\Delta_2 \subseteq \mathrm{LR}(0)$. Suppose $L \in \Delta_2$. Then $L = L\{\Lambda\}^* \in \mathrm{LR}(0)$ by (d) of the LR(0) characterization theorem. We now show proper inclusion. We know $a^* \notin \Delta_2$, but $a^* \in \mathrm{LR}(0)$ by (d) of the LR(0) characterization theorem, since $a^* = \Lambda a^*$. $\square$

Lemma 4.2. $LR(0) \subsetneq \Delta_1$.

Proof. By (c) of the LR(0) characterization theorem, we know $\mathrm{LR}(0) \subseteq \Delta_1$. We next show proper inclusion. Let $L = \{ab^*\} \cup \{cd^*\}$. Since L is regular, $L \in \Delta_1$ by [11]. Suppose L were LR(0). Since $a, ab, c \in L$, we have $cb \in L$ by the LR(0) characterization theorem. But this is a contradiction and L is not LR(0). $\square$

We now prove our inclusion theorem.

[4] Recall that Δ_1 is the family of languages accepted by DPDA's by final state and with one symbol on the stack.

Theorem 4.3. $\Delta_2 \subsetneq LR(0) \subsetneq \Delta_1 \subsetneq \Delta_0$.

Proof. $\Delta_2 \subsetneq \mathrm{LR}(0)$ by Lemma 4.1. $\mathrm{LR}(0) \subsetneq \Delta_1$ by Lemma 4.2. $\Delta_1 \subsetneq \Delta_0$ by [11]. □

We now consider the closure properties of the classes of languages Δ_2, LR(0), Δ_1 and Δ_0 under operations with regular sets, boolean operations, Kleene operations, marked operations, etc. We begin with a theorem which will help us to check if a language is in Δ_1, but a new definition is required first.

Definition 4.4. Let $L \subseteq \Sigma^*$ be a deterministic context free language. We define the *relative right congruence relation induced by* L, R_L as follows:

For $x, y \in L$,

$$(x, y) \in R_L \quad \text{if and only if for all } z \in \Sigma^*,\ xz \in L \quad \text{if and only if } yz \in L.$$

This is clearly an equivalence relation. It is quite similar to the induced right congruence relations defined on regular sets, cf. [14]. However, this relation is defined only among elements in L, whereas the right congruence relation is defined on all elements of Σ^*.

Our first theorem shows that R_L is of finite rank when $L \in \Delta_1$. This compares with the result that the right congruence relation induced on regular sets is finite when L is regular.

Theorem 4.5. *Let $L \subseteq \Sigma^*$ be a deterministic language. Then R_L is of finite rank if and only if L is in Δ_1.*

Proof. Assume that L is a Δ_1 language. Assume for the sake of contradiction that R_L is not of finite rank on L. Since L is a Δ_1 language, there exists a DPDA $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$ such that $T(M, \Gamma) = L$ where for $m, n \geq 1$, $\Gamma = \{Z_0, \ldots, Z_{n-1}\}$ and $F = \{q_1, \ldots, q_m\}$.

Thus, there exist mn final configurations of the form (q_i, Λ, Z_j) $1 \leq i \leq m$, $0 \leq j \leq n-1$. Since R_L is not of finite rank on L, for some $x, y \in L$ such that $(x, y) \notin R_L$ we must have for some i, j such that $0 \leq i \leq n-1$, $1 \leq j \leq m$,

$$(q_0, x, Z_0) \overset{*}{\vdash} (q_i, \Lambda, Z_j)$$
$$(q_0, y, Z_0) \overset{*}{\vdash} (q_i, \Lambda, Z_j).$$

Thus for all $z \in \Sigma^*$, $xz \in L$ if and only if $yz \in L$. Thus $(x, y) \in R_L$ and this is a contradiction.

In Theorem 7.1 of [20], it is shown that a sufficiently large reachable configuration is equivalent to a smaller reachable configuration or there exist infinitely many pairwise inequivalent configurations. This can be shown to hold with accepting configurations. If R_L has finite rank, it is easy to see that the second possibility cannot occur. Moreover, one can carry out the transformation to the smaller accepting configuration in a finite state control. Hence it is possible to convert from a DPDA A for L to another DPDA A' which accepts L as $T_1(A')$. The details are omitted. This sketch of the proof was suggested by the referee. □

We now wish to study the various closure properties of these classes of languages. Since we wish to study the four classes Δ_i and LR(0) and some fifteen operations, we would have to deal with some sixty cases. To avoid this tedious detail, we summarize the results in Table 1 and shall present the proof of two typical negative results. The rest of the proofs are omitted but the reader can find full details in [4].

Table 1. Closure properties of deterministic subfamilies.

		Δ_2	LR(0)	Δ_1	Δ_0
Operations with regular sets					
Product	LR	No	No	No	Yes
Intersection	$L_1 \cap L_2$	Yes	No	Yes	Yes
Quotient	LR^{-1}	No	No	No	Yes
Boolean operations					
Union	$L_1 \cup L_2$	No	No	No	No
Intersection	$L_1 \cap L_2$	No	No	No	No
Complement	$\bar{L}$	No	No	No	Yes
Kleene operations					
*	L^*	No	No	No	No
Product	L_1L_2	Yes	Yes	No	No
Marked operations					
Union	$c_1L_1 \cup c_2L_2$	Yes	No	Yes	Yes
Product	$L_1\$L_2$	Yes	No	Yes	Yes
Other operations					
Min		Yes	Yes	Yes	Yes
Max		Yes	Yes	Yes	Yes
Reversal		No	No	No	No
Homomorphism		No	No	No	No
Inverse G.S.M.		No	No	Yes	Yes

Δ_1 is not closed under complement

Consider the language $L = \{a^nb^n \mid n \geq 1\}$. Since $L \in \Delta_2$, we know $L \in \Delta_1$. Assume for the sake of contradiction that $\bar{L} \in \Delta_1$. Now, for all $i \geq 1$, $a^i \in \bar{L}$. Therefore, by Theorem 4.5, for some k_1, k_2 such that $0 \leq k_1 < k_2$ we have $(a^{k_1}, a^{k_2}) \in R_L$. Thus for all $z \in \Sigma^*$,

$$a^{k_1}z \in \bar{L} \quad \text{if and only if } a^{k_2}z \in \bar{L}.$$

Let $z = b^{k_2}$. We know $a^{k_1}b^{k_2} \in \bar{L}$. Thus $a^{k_2}b^{k_2} \in \bar{L}$. But this is a contradiction.

LR(0) is not closed under complement

Let $L=(abb)(b^*)\subseteq\{a,b\}^*$. We know $a, ab\in\bar{L}$ but $abb\notin\bar{L}$. Then by the corollary to the LR(0) characterization theorem, $\bar{L}\notin$ LR(0).

There are some natural decision questions which are closely associated with the present study. We know that one can decide if a deterministic language is regular or not by [19]. If the given language is not known to be deterministic then the corresponding question is undecidable from [2]. Is it recursively decidable whether or not a deterministic language is strict deterministic? In view of [11] it is equivalent to ask if a deterministic language is prefix free. First, we consider the general case and quote the relevant result from [3].

Theorem 4.6. *It is recursively undecidable whether or not a context free language is prefix free.*

The problem becomes decidable in the deterministic case. Also cf. [20].

Theorem 4.7. *There is an algorithm to decide whether or not a given deterministic context free language is prefix free.*

Proof. Our original proof of this result was based on the properties of strict deterministic grammars. We sketch a much simpler proof, suggested by Ullman, based on DPDA's. Let $L\subseteq\Sigma^*$ be a deterministic context free language. By construction of a DPDA for $L-\min(L)$, it is not hard to see that the set $L-\min(L)$ is a deterministic language. Moreover, L is prefix free if and only if $L-\min(L)=\emptyset$. Since it is decidable if a context free language is empty, the result follows. □

From the previous result, we get an important consequence.

Corollary. *It is decidable whether or not a deterministic language is strict deterministic.*

Proof. From [11] a deterministic language is strict deterministic if and only if it is prefix free. □

There is a natural extension of the previous question. Can one decide if a deterministic language is LR(0)? We will show that this seemingly mild question is equivalent to the equality problem for deterministic context free languages.

Next, we state the equivalence problem for DPDA's.

$\mathcal{P}_0$: *Equivalence problem for Δ_0.* Is it recursively solvable to determine of two DPDA's, A_1 and A_2, whether or not $T(A_1)=T(A_2)$?

The present problem can be stated as follows.

$\mathcal{P}_1$: *Decidability of LR*(0). Is it recursively decidable whether or not a given deterministic language is LR(0)?

Theorem 4.8. *$\mathcal{P}_0$ is equivalent to $\mathcal{P}_1$, i.e., there is an algorithm to decide if a deterministic language is LR*(0) *if and only if there is an algorithm to decide if two deterministic context free languages are equal.*

Proof. We first assume that there is an algorithm to decide if a deterministic language is LR(0). Let $L_1, L_2 \subseteq \Sigma^*$ be two deterministic context free languages and let $c_1, c_2, c_3, \$$ be four new symbols not in Σ. Consider the following set:

$$L = c_1(L\$)^* c_3(L_2\$)^* \cup c_2(L_2\$)^* c_3(L_2\$)^*.$$

L is a variant of a set proposed by Ullman.

Claim 1. *L is deterministic.*

Proof. Since L_1 and L_2 are deterministic, $L_1\$$ and $L_2\$$ are strict deterministic. Therefore $(L_1\$)^*$ and $(L_2\$)^*$ are LR(0) languages by (b) of the LR(0) language characterization theorem, thus deterministic. It follows that $(L_1\$)^* c_3$ and $(L_2\$)^* c_3$ are strict deterministic. Therefore $(L_1\$)^* c_3(L_2\$)^*$ and $(L_2\$)^* c_3(L_1\$)^*$ are LR(0) languages again by (b) of the LR(0) characterization theorem and thus deterministic. Thus the marked union of these two languages, namely

$$L = c_1(L_1\$)^* c_3(L_2\$)^* \cup c_2(L_2\$)^* c_3(L_1\$)^*$$

is deterministic from the closure results of the previous section.

Claim 2. *$L_1 = L_2$ implies L is an LR*(0) *language.*

Proof. If $L_1 = L_2$, we have

$$\begin{aligned} L &= c_1(L_1\$)^* c_3(L_1\$)^* \cup c_2(L_1\$)^* c_3(L_1\$)^* \\ &= ((c_1(L_1\$)^* \cup c_2(L_1\$)^*)c_3)(L_1\$)^* \end{aligned}$$

which is an LR(0) language from the proof of Claim 1 and (d) of the LR(0) characterization theorem.

Claim 3. *If L is an LR*(0) *language then $L_1 = L_2$.*

Proof. Assume for the sake of contradiction that $L_1 \neq L_2$. We assume without loss of generality that there exists some $x \in L_1\$$ such that $x \notin L_2\$$, where $x \neq \Lambda$. Choose any $y \in L_2\$$. We know $c_2yc_3 \in L$, $c_2yc_3x \in L$, $c_1xc_3 \in L$. It follows from (b) of the LR(0) characterization theorem that $c_1xc_3x \in L$. Therefore $x \in L_2\$$, but this is a contradiction and therefore $L_1 = L_2$.

We have therefore shown that $L_1 = L_2$ if and only if L is an LR(0) language. In order to decide if $L_1 = L_2$ we simply construct the DPDA for L, and then use our algorithm to decide if L is LR(0). Thus, we have an algorithm to decide if two deterministic context free languages are equal.

Conversely, we assume that there is an algorithm to decide if two deterministic context free languages are equal. Let $L \subseteq \Sigma^*$ be a deterministic context free language. We now provide an algorithm for deciding if L is an LR(0) language. Now let $L_0 = \min(L)$. If $L \neq L_0$, we choose some $x \in \Sigma^*$, $z \in \Sigma^+$ such that $x \in L_0$ and $xz \in L$. We let $L_1 = \min\{y \in \Sigma^* \mid xy \in L\}$, as in Case 2 of the proof that (c) implies (d) in the LR(0) characterization theorem. L_0 and L_1 will be strict deterministic. We then let $L' = L_0 L_1^*$.

Claim 4. *L is an LR(0) language if and only if $L = L'$.*

Proof. If L is an LR(0) language, by our LR(0) characterization theorem, $L = L'$. If $L = L'$, then $L = L_0 L_1^*$, where L_0 and L_1 are strict deterministic. Thus, by our characterization theorem, L is LR(0).

We have assumed that there is an algorithm to decide if two deterministic context free languages are equal; we need only to test if $L = L'$ in order to determine if L is an LR(0) language. □

The preceding theorem can in fact be strengthened.

Corollary 4.8.1. *There is an algorithm to decide if a Δ_1 language with two final configurations is LR(0) if and only if there is an algorithm to decide if two deterministic context free languages are equal.*

Proof. $L = c_1(L_1\$)^* c_3(L_2\$) \cup c_2(L_2\$)^* c_3(L_1\$)^*$ is a Δ_1 language with two final configurations. □

Corollary 4.8.2. *There is an algorithm to decide if a Δ_1 language is LR(0) if and only if there is an algorithm to decide if two deterministic context free languages are equal.*

Proof. Follows directly from Corollary 4.8.1. □

5. Conclusions

A new definition of LR(k) grammars has been given which is closely related to the original definition. It has been shown how this definition relates to other definitions in the literature. In particular, our definition gives unambiguous grammars, as well as a large class of both grammars and languages.

It remains for us to show that grammars that we have defined to be LR(k) can in fact be parsed left-to-right with k lookahead, with our parser outputting the rightmost derivation of a string in the language, and outputting "error" for a string not in the language defined by the grammar. We shall produce such parsers in a sequel [7] by paying careful attention to the halting condition on the parser. We do not use the LLR(k) definition on the grounds that it does not naturally correspond with a parser with k lookahead, since it does not deal with canonical derivations.

References

[1] A.V. Aho and J.D. Ullman, *The Theory of Parsing, Translating, and Compiling*, Vols. I and II (Prentice Hall, Englewood Cliffs, NJ, 1972 and 1973).

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

[3] J. Engelfriet, Translation of simple program schemes, in: M. Nivat, ed., *Automata, Languages, and Programming* (North-Holland Publishing Co., Amsterdam, 1973) 215–224.

[4] M.M. Geller, Compact parsers for deterministic languages, Ph.D. Thesis, Department of Computer Science, University of California, Berkeley, California (1975).

[5] M.M. Geller and M.A. Harrison, Strict deterministic versus LR(0) parsing, Conference Record of the ACM Symposium on Principles of Programming Languages, Boston, MA (1973) 22–32.

[6] M.M. Geller and M.A. Harrison, Characterizations of LR(0) languages, Conference Record of the 14th Annual Symposium on Switching and Automata Theory, 73 CHO 786–4–C (1973) 103–108.

[7] M.M. Geller and M.A. Harrison, Characteristic parsing: A framework for producing compact deterministic parsers, *J. Comput. Systems Sci.* (to appear).

[8] S. Ginsburg, *The Mathematical Theory of Context-Free Languages* (McGraw-Hill, NY, 1966).

[9] S. Ginsburg and S.A. Greibach, Deterministic context-free languages, *Information and Control* **9** (1966) 602–648.

[10] M.A. Harrison, Fall Quarter 1971 Class Notes for Computer Science 234, University of California, Berkeley (1971).

[11] M.A. Harrison and I.M. Havel, Strict deterministic languages, *J. Comput. Systems Sci.* **7** (1973) 237–277.

[12] M.A. Harrison and I.M. Havel, On the parsing of deterministic languages, *J. ACM* **21** (1974) 525–548.

[13] M.A. Harrison and I.M. Havel, Real-time strict deterministic languages, *SIAM J. Comput.* **1** (1972) 333–349.

[14] J.E. Hopcroft and J.D. Ullman, *Formal Languages and Their Relation to Automata* (Addison-Wesley, Reading, MA, 1969).

[15] D.E. Knuth, On the translation of languages from left to right, *Information and Control* **8** (1965) 607–639.

[16] D. Lehmann, LR(k) grammars and deterministic languages, *Israel J. Math.* **10** (1971) 526–530.

[17] P.M. Lewis and R.E. Stearns, Syntax-directed transductions, *J. ACM* **15** (1968) 465–485.

[18] A. Salomaa, *Formal Languages* (Academic Press, NY, 1973).

[19] R.E. Stearns, A regularity test for pushdown machines, *Information and Control* **11** (1967) 323–340.

[20] L.G. Valiant, Decision procedure for DPDAs, Ph.D. Thesis, University of Warwick, Warwick, England (1973).