INFORMATION AND CONTROL **53**, 141–164 (1982)

On LL(k) Parsing*

SEPPO SIPPU AND ELJAS SOISALON-SOININEN

Department of Computer Science, University of Helsinki, Tukholmankatu 2, SF-00250 Helsinki 25, Finland

The theory of LL(k) parsing of context-free grammars is developed as a dual of the theory of LR(k) parsing. An LL(k) parser is regarded as a push-down transducer in which the push-down symbols are certain equivalence classes of "viable suffixes," a dual concept of the "viable prefixes" used in the LR(k) theory. The approach allows a rigorous mathematical treatment, including general correctness proofs, of LL(k) parsers obtained via different equivalence relations on the viable suffixes. In particular, the equivalence relation that yields the canonical LL(k) parser is considered, and a new method for constructing the canonical parser is given. This method is based on sets of "items" similar to those used in Knuth's method for constructing LR(k) parsers. An implication of this is that various techniques in the LR(k) theory, e.g., optimization and efficient testing methods, can easily be adopted in the LL(k) theory as well. This is also of practical importance because canonical LL(k) parsers for $k = 1$ have gained new attention in error handling, thanks to their capability for early error detection.

1. INTRODUCTION

The theory of top-down parsing was initiated by Lewis and Stearns (1968) and Knuth (1971). They were the first who defined the class of LL(k) grammars, i.e., grammars that can be parsed deterministically top-down in a "natural" manner using lookahead strings of length k at most. Knuth (1971) suggested the term "LL(k)," where "LL" means that parsing is carried out from left to right constructing the leftmost derivation, and "k" denotes the lookahead length. Soon after the introduction of these grammars Rosenkrantz and Stearns (1970) studied extensively their properties emphasizing both properties of languages and constructive properties of grammars.

The LL(k) parser of an LL(k) grammar is essentially a one state deterministic push-down transducer which produces the left parse of the given

* Preliminary versions of some of the results in this paper were presented at the 6th International Colloquium on Automata, Languages, and Programming, Graz, Austria, July 1979. The work was supported by the Academy of Finland.

0019-9958/82 $2.00

Copyright © 1982 by Academic Press, Inc.
All rights of reproduction in any form reserved.

input string. For a subclass of LL(k) grammars, called "strong" LL(k) grammars, this parser is particularly simple. The push-down symbols of a strong LL(k) parser are symbols of the grammar, i.e., nonterminals and terminals.

Different methods for constructing LL(k) parsers of general LL(k) grammars have been devised. Rosenkrantz and Stearns (1970) presented a method for transforming LL(k) grammars into strong LL(k) grammars. This transformation actually yields a parsing algorithm for all LL(k) grammars because the transformed grammar and the original grammar are structurally equivalent, i.e., the derivation trees without nonterminal labels in the nodes are the same. Thus, when parsing the transformed grammar, the parse according to the original grammar can be produced. Aho and Ullman (1972) construct the LL(k) parser directly without any grammatical transformation but their construction is closely related to the strong LL(k) transformation of Rosenkrantz and Stearns (1970).

In the case $k = 1$ all LL(k) grammars are strong LL(k) grammars as well. Thus for LL(1) grammars the simple strong LL(1) parsing method can be used. However, even though the strong LL(k) parser of an LL(k) grammar worked correctly, it may detect errors later than the true LL(k) parser. In the case $k = 1$ this means that the strong parser may perform several extra produce actions. New methods for good quality error recovery (e.g., Fischer *et al.*, 1980) require that errors are detected as early as possible, i.e., in the case of LL(1) parsing immediately after the last symbol of the longest correct prefix has been shifted. This motivates the consideration of the general construction method even in the case $k = 1$.

We present a method for constructing general LL(k) parsers, which is very similar to the usual method for constructing LR(k) parsers (Knuth, 1965; Aho and Ullman, 1972). This approach allows a mathematical treatment of the theory of LL(k) parsing. Also different methods in the well established LR(k) theory can almost directly be applied to the LL(k) theory as well because of the dual relationship between the theories. An example of this is the fast algorithm for testing a grammar for the LL(k) property (Sippu and Soisalon-Soininen, 1980). This algorithm relies on the construction method given in the present paper, and it nicely demonstrates the difference between the LL(k) and LR(k) properties.

The present paper is organized as follows: In Section 2 the background terminology is given in such a way that a rigorous treatment of the subject is possible. Section 3 is devoted to the analysis of "viable suffixes," the dual concept of viable prefixes essential in the LR(k) theory. Actually, viable suffixes of a grammar are viable prefixes of the grammar obtained by reversing the right-hand sides of the productions. Based on the division of viable suffixes into a finite number of equivalence classes, a general scheme for LL(k) parsing is presented in Section 4. In Section 5 we then present our method for constructing LL(k) parsers, which can be considered as a dual of the usual method for constructing LR(k) parsers. A short comparison with other construction methods is given in Section 6.

2. Notations and Definitions

In this paper we use a somewhat unconventional formalization for grammars, push-down automata, and parsers: we regard these as special cases of a general rewriting system (cf. Deussen, 1979). A *rewriting system* $\mathscr{S}$ is a pair (V, P), where V is a finite vocabulary and P is a finite relation on V^*. The elements (ω_1, ω_2) in P are called *productions* or *rules* of $\mathscr{S}$, and denoted by $\omega_1 \to \omega_2$.

If $\omega_1 \to \omega_2$ is a production of $\mathscr{S}$, we define $\Rightarrow_{\mathscr{S}}^{\omega_1 \to \omega_2}$ to be the relation $\{(\alpha\omega_1\beta, \alpha\omega_2\beta) \mid \alpha, \beta \in V^*\}$ on V^*. If π is a string of productions and r is a single production, we define $\Rightarrow_{\mathscr{S}}^{\pi r}$ to be the composite relation $\Rightarrow_{\mathscr{S}}^{r} \circ \Rightarrow_{\mathscr{S}}^{\pi}$. We stipulate that $\Rightarrow_{\mathscr{S}}^{\pi}$ is the identity relation on V^* if π is the empty string ε. The union of all $\Rightarrow_{\mathscr{S}}^{r}$, r in P, is denoted by $\Rightarrow_{\mathscr{S}}$ and called the *derives* relation of $\mathscr{S}$. As usual, we drop $\mathscr{S}$ from $\Rightarrow_{\mathscr{S}}^{\pi}$ and $\Rightarrow_{\mathscr{S}}$ if $\mathscr{S}$ is understood.

A rewriting system $G = (V, P)$ is a (*context-free*) *grammar* with *start symbol* S and *terminal vocabulary* T if $T \subset V$ and S is a distinguished element in $V \backslash T$ and if each production in P is of the form $A \to \omega$, where A is in $V \backslash T$.

Throughout the paper we use the notational convention that (1) A, B, C, S denote *nonterminals*, i.e., symbols in $V \backslash T$, (2) a, b, c denote *terminals*, i.e., symbols in T, (3) X, Y, Z denote either nonterminals or terminals, (4) $u, v, \ldots, z$ denote terminal strings, and that (5) $\alpha, \beta, \ldots, \omega$ denote general strings in V^*.

The *language generated* by G, i.e., the set $\{w \in T^* \mid S \overset{*}{\Rightarrow} w\}$, is denoted by $L(G)$. The elements of $L(G)$ are called *sentences* of G. A symbol X of G is *useful* if $S \overset{*}{\Rightarrow} \alpha X \beta \overset{*}{\Rightarrow} w$ holds for some strings α and β and terminal string w. A grammar that has useful symbols only is called *reduced*.

The subrelation $\{(\alpha A y, \alpha\omega y) \mid \alpha \in V^*, y \in T^*\}$ of $\Rightarrow^{A \to \omega}$ is denoted by $\Rightarrow_{rm}^{A \to \omega}$, and the subrelation $\{(xA\beta, x\omega\beta) \mid x \in T^*, \beta \in V^*\}$ by $\Rightarrow_{lm}^{A \to \omega}$. These relations generalize to $\Rightarrow_{rm}^{\pi}$, $\Rightarrow_{lm}^{\pi}$, $\Rightarrow_{rm}$, and $\Rightarrow_{lm}$, as above. A production string π is a *left parse* of a string ϕ in G if $S \Rightarrow_{lm}^{\pi} \phi$, and a right parse if $S \Rightarrow_{rm}^{\pi^R} \phi$. Here π^R means the *reversal*, i.e., mirror image, of the string π. String ϕ is called a *left sentential form* if it has a left parse, and a *right sentential form* if it has a right parse.

If ϕ is a string and k is a nonnegative integer, then $k{:}\phi$ denotes the k-length prefix of ϕ, or ϕ itself if k is greater than $|\phi|$, the length of ϕ. Similarly, $\phi{:}k$ denotes the k-length suffix of ϕ. $\mathrm{FIRST}_k(\phi)$ means the set $\{k{:}w \mid \phi \overset{*}{\Rightarrow} w \in T^*\}$, and $\mathrm{FOLLOW}_k(\phi)$ the set $\{w \mid S \overset{*}{\Rightarrow} \alpha\phi\beta,\ w \in \mathrm{FIRST}_k(\beta)\}$.

A grammar G is LL(k) if $\mathrm{FIRST}_k(\omega_1\delta)$ and $\mathrm{FIRST}_k(\omega_2\delta)$ are disjoint whenever $xA\delta$ is a left sentential form of G and $A \to \omega_1$ and $A \to \omega_2$ are distinct productions of G. Grammar G is *strong* LL(k) if $\mathrm{FIRST}_k(\omega_1\,\mathrm{FOLLOW}_k(A))$ and $\mathrm{FIRST}_k(\omega_2\,\mathrm{FOLLOW}_k(A))$ are disjoint whenever $A \to \omega_1$ and $A \to \omega_2$ are distinct productions of G.

A rewriting system $M = (V, P)$ is an *(extended) push-down automaton* with *input vocabulary* T, *bottom-of-stack marker* §, *input head* |, *end-of-input marker* \$, *initial stack contents* ϕ_S and *final stack contents* ϕ_F if $T \subset V$, |, § and \$ are distinct symbols in $V \backslash T$, ϕ_S and ϕ_F are in V^* and if each production in P is of the form

$$\alpha \mid xy \to \beta \mid y, \tag{2.1}$$

where $\alpha, \beta \in V^*$ and $xy \in T^* \cup T^*\$$. By way of distinction from grammars, we often call productions of a push-down automaton M *actions* of M.

An action string π is a *parse* of an input string w in M if

$$\S\phi_S \mid w\$ \overset{\pi}{\underset{M}{\Longrightarrow}} \S\phi_F \mid \$.$$

The *language accepted* by M, denoted $L(M)$, is defined to be the set of all input strings that have a parse in M.

M is *deterministic* if for any configuration Φ, i.e., a string in $\S(V\backslash\{|, \S, \$\})^* \mid T^*\$$, M has at most one production r that applies to Φ, i.e., $\Phi \Rightarrow^r_M \Phi'$ for some Φ'.

Let G be a grammar with productions P_G and terminals T, and let M be a push-down automaton with productions P_M and input vocabulary T. Furthermore, let τ be a homomorphism: $P_M^* \to P_G^*$. We then say that the pair (M, τ) is a *left* (resp. *right*) *parser* of G if (1) $\tau(\pi')$ is a left (resp. right) parse of w in G whenever π' is a parse of w in M, and (2) for each left (resp. right) parse π of w in G, $\tau(\pi') = \pi$ for some parse π' of w in M.

Note that $L(G) = L(M)$ if (M, τ) is a left or a right parser of G.

If ϕ_1 and ϕ_2 are strings of a rewriting system $\mathcal{S}$ such that $\phi_1 \overset{*}{\Rightarrow}_{\mathcal{S}} \phi_2$, then the *time complexity* of deriving ϕ_2 from ϕ_1 in $\mathcal{S}$ is defined as

$$\mathrm{TIME}_{\mathcal{S}}(\phi_1, \phi_2) = \min\{n \geqslant 0 \mid \phi_1 \overset{n}{\underset{\mathcal{S}}{\Longrightarrow}} \phi_2\}.$$

For a grammar G with start symbol S, and for any w in $L(G)$ we define

$$\mathrm{TIME}_G(w) = \mathrm{TIME}_G(S, w).$$

For a push-down automaton M, and for any w in $L(M)$ we define

$$\mathrm{TIME}_M(w) = \mathrm{TIME}_M(\S\phi_S \mid w\$, \S\phi_F \mid \$).$$

3. VIABLE PREFIXES AND VIABLE SUFFIXES

We recall that a string γ is a *viable prefix* of a grammar G if

$$S \underset{rm}{\overset{*}{\Longrightarrow}} \delta A y \underset{rm}{\Longrightarrow} \delta\alpha\beta y = \gamma\beta y \tag{3.1}$$

holds in G for some string δ, production $A \to \alpha\beta$ and terminal string y.

Actually both Aho and Ullman (1972) and Harrison (1978) give a slightly different definition for a viable prefix: they call any prefix of $\delta\alpha\beta$ in a derivation of the form (3.1) a viable prefix. Indeed, our viable prefixes are the *valid* prefixes of Harrison (1978). However, the two concepts turn out to be equivalent, as is shown in Theorem 13.5.3 in Harrison (1978).

Viable prefixes play an important role in the theory of LR parsing: in an LR parser the entry symbols of the states appearing in the parsing stack always form a viable prefix. For a rigorous and symmetric treatment of LL parsing it is necessary to give an analogous grammatical characterization for the strings appearing in the stack of a (strong) LL(k) parser (see Aho and Ullman, 1972).

Formally, we say that a string γ is a *viable suffix* of G if in G

$$S \underset{lm}{\overset{*}{\Longrightarrow}} xA\delta \underset{lm}{\Longrightarrow} x\alpha\beta\delta = x\alpha\gamma^{\mathrm{R}} \tag{3.2}$$

for some terminal string x, production $A \to \alpha\beta$ and string δ. Thus, viable suffixes are reversals of certain suffixes of left sentential forms, whereas viable prefixes are certain prefixes of right sentential forms.

It turns out that the viable suffixes of a grammar G coincide with viable prefixes of the *reversed grammar* G^{R}. Here G^{R} is obtained from G by replacing each production $p = A \to \omega$ in G by its *reversal* p^{R}, i.e., by the production $A \to \omega^{\mathrm{R}}$ in which ω^{R} is the reversal of the string ω.

FACT 3.1. (a) *$\phi_1 \Rightarrow_{rm}^{p_1 \cdots p_n} \phi_2$ in G if and only if $\phi_1^{\mathrm{R}} \Rightarrow_{lm}^{p_1^{\mathrm{R}} \cdots p_n^{\mathrm{R}}} \phi_2^{\mathrm{R}}$ in G^{R}.*
(b) *$\phi_1 \Rightarrow_{lm}^{p_1 \cdots p_n} \phi_2$ in G if and only if $\phi_1^{\mathrm{R}} \Rightarrow_{rm}^{p_1^{\mathrm{R}} \cdots p_n^{\mathrm{R}}} \phi_2^{\mathrm{R}}$ in G^{R}.*

Proof. A simple induction on n (cf. the proof of Theorem 3.3.1 in Harrison, 1978). ∎

A string γ is called a *complete* viable prefix if (3.1) holds for some $\beta = \varepsilon$. Similarly, γ is called a *complete* viable suffix if (3.2) holds for some $\alpha = \varepsilon$. Complete viable prefixes, we recall, correspond to LR parsing configurations in which a reduce action can be applied. As we shall see, complete viable suffixes similarly correspond to LL parsing configurations that result from an application of a produce action.

FACT 3.2. (a) *A string* γ *is a (complete) viable prefix of* G *if and only if* γ *is a (complete) viable suffix of* G^{R}.

(b) *A string* γ *is a (complete) viable suffix of* G *if and only if* γ *is a (complete) viable prefix of* G^{R}.

Proof. The "only if" part of (a) follows from the fact that if

$$S \underset{rm}{\overset{*}{\Longrightarrow}} \delta A y \underset{rm}{\Longrightarrow} \delta\alpha\beta y = \gamma\beta y \tag{1}$$

holds in G then, by Fact 3.1,

$$S \underset{lm}{\overset{*}{\Longrightarrow}} (\delta A y)^{\mathrm{R}} = y^{\mathrm{R}} A \delta^{\mathrm{R}} \underset{lm}{\Longrightarrow} y^{\mathrm{R}}(\alpha\beta)^{\mathrm{R}}\,\delta^{\mathrm{R}} = y^{\mathrm{R}}\beta^{\mathrm{R}}(\delta\alpha)^{\mathrm{R}} = y^{\mathrm{R}}\beta^{\mathrm{R}}\gamma^{\mathrm{R}} \tag{2}$$

holds in G^{R}. Furthermore, note that if $\beta = \varepsilon$ in (1), then $\beta^{\mathrm{R}} = \varepsilon$ in (2). The "only if" part of (b) can be handled analogously. Finally, the "if" parts of both (a) and (b) follow from the fact that $(G^{\mathrm{R}})^{\mathrm{R}} = G$. ∎

The following technical lemma is useful in proving properties of viable prefixes. It is similar to Lemma 13.5.2 in Harrison (1978), which is used to prove the equivalence of viable prefixes and valid prefixes.

LEMMA 3.3. *Let* G *be a grammar,* π *a production string,* γ, η, *and* δ *strings,* A *a nonterminal and* y *a terminal string of* G *such that*

$$S \underset{rm}{\overset{\pi}{\Longrightarrow}} \gamma\eta y = \delta A y \qquad \textit{and} \qquad \pi \neq \varepsilon. \tag{a}$$

In other words, γ *is a prefix of a nontrivially derived right sentential form not extending over the last nonterminal. Then* G *has a string* δ', *production strings* π' *and* π'', *a production* $r = A' \to \alpha'\beta'$ *and a terminal string* y' *such that*

$$\begin{gathered} S \underset{rm}{\overset{\pi'}{\Longrightarrow}} \delta' A' y' \underset{rm}{\overset{r}{\Longrightarrow}} \delta'\alpha'\beta' y' = \gamma\beta' y', \qquad \beta' y' \underset{rm}{\overset{\pi''}{\Longrightarrow}} \eta y, \\ \pi' r \pi'' = \pi \qquad \textit{and} \qquad \alpha' : 1 = \gamma : 1. \end{gathered} \tag{b}$$

In other words, derivation (a) *has a segment that proves* γ *to be a viable prefix, even so that the right-hand side of the production* r *"cuts"* γ *properly.*

Proof. A straightforward induction on the length $|\pi|$ of the production string π. ∎

Lemma 3.3 has its natural counterpart for leftmost derivations:

LEMMA 3.4. *Let G be a grammar, π a production string, x a terminal string, η, γ, and δ strings and A a nonterminal of G such that*

$$S \underset{lm}{\overset{\pi}{\Longrightarrow}} x\eta\gamma = xA\delta \qquad \textit{and} \qquad \pi \neq \varepsilon. \tag{a}$$

Then G has a terminal string x', production strings π' and π'', a production $r = A' \to \alpha'\beta'$ and a string δ' such that

$$\begin{gathered} S \underset{lm}{\overset{\pi'}{\Longrightarrow}} x'A'\delta' \underset{lm}{\overset{r}{\Longrightarrow}} x'\alpha'\beta'\delta' = x'\alpha'\gamma, \qquad x'\alpha' \underset{lm}{\overset{\pi''}{\Longrightarrow}} x\eta, \\ \pi' r \pi'' = \pi \qquad \textit{and} \qquad 1:\beta' = 1:\gamma. \end{gathered} \tag{b}$$

Proof. Analogous to that of Lemma 3.3. The result can also be obtained from Lemma 3.3 by means of Fact 3.1. ∎

As an immediate consequence of Lemmas 3.3 and 3.4 we have

THEOREM 3.5. *Let G be a grammar, A a nonterminal and y a terminal string of G.*

(a) *If $S \Rightarrow_{rm}^{+} \delta A y$ in G, then δA is a viable prefix of G.*

(b) *If $S \Rightarrow_{lm}^{+} yA\delta$ in G, then $\delta^{R}A$ is a viable suffix of G.* ∎

The following theorem states that every prefix of a viable prefix (or suffix) is viable. In effect, the theorem establishes the equivalence of our definition for a viable prefix and that by Aho and Ullman (1972) and Harrison (1978).

THEOREM 3.6. *If $\gamma_1\gamma_2$ is a viable prefix (resp. viable suffix) of G, then γ_1 is a viable prefix (resp. viable suffix) of G.*

Proof. We first handle the case in which $\gamma_1\gamma_2$ is a viable prefix. By definition,

$$S \underset{rm}{\overset{n}{\Longrightarrow}} \delta A y \underset{rm}{\Longrightarrow} \delta\alpha\beta y = \gamma_1\gamma_2\beta y \tag{1}$$

holds for some integer $n \geqslant 0$, string δ, production $A \to \alpha\beta$ and terminal string y. Then either δ is a prefix of γ_1, or $\delta \neq \varepsilon$ and γ_1 is a prefix of δ. In the former case (1) proves γ_1 to be a viable prefix because we may then write $\alpha\beta$ as $\alpha'\beta'$ such that $\delta\alpha' = \gamma_1$. In the latter case we may write $\delta A y$ in (1) as $\gamma_1\eta y$ for some η. Because $\delta \neq \varepsilon$ implies that $n > 0$ in (1), we can then conclude by Lemma 3.3 that γ_1 is a viable prefix.

The case in which $\gamma_1\gamma_2$ is a viable suffix can now be handled easily: By Fact 3.2, $\gamma_1\gamma_2$ is a viable prefix of G^{R}. By the above reasoning γ_1 is then a viable prefix of G^{R}, and therefore, by Fact 3.2, a viable suffix of G. ∎

The following fact follows immediately from the definitions of a viable prefix and a viable suffix.

FACT 3.7. *Let G be a reduced grammar, δ a string and $A \rightarrow \alpha\beta$ a production of G.*

(a) *If δA is a viable prefix of G, then so is $\delta\alpha$.*

(b) *If δA is a viable suffix of G, then so is $\delta\beta^R$.* ∎

4. General LL Parsing

The canonical LR(k) parsing machine of a grammar G induces in a natural way an equivalence relation on the set of viable prefixes of G: viable prefixes γ_1 and γ_2 might be called LR(k)-*equivalent* if they lead to the same state in the parsing machine. In effect, γ_1 and γ_2 are LR(k)-equivalent if and only if exactly the same canonical LR(k) parsing actions apply both to γ_1 and γ_2.

The LR(k)-equivalence has the following properties: (1) it is of a *finite index*, i.e., the number of equivalence classes is finite (the parsing machine has only a finite number of states); (2) it is *right invariant*, i.e., if X is a single symbol and $\gamma_1 X$ and $\gamma_2 X$ are viable prefixes such that γ_1 is equivalent to γ_2, then $\gamma_1 X$ is equivalent to $\gamma_2 X$ (if γ_1 and γ_2 lead to the same state then, of course, do $\gamma_1 X$ and $\gamma_2 X$); and (3) $\gamma_1 : 1 = \gamma_2 : 1$ holds for equivalent viable prefixes γ_1 and γ_2 (each state has a unique entry symbol).

These three conditions capture the "essence" of LR parser construction in that all "LR-like" parsers are obtained using in place of the LR(k)-equivalence *any* equivalence relation ρ on the viable prefixes that satisfies the three conditions. This kind of a general approach to the LR theory has been adopted by Geller and Harrison (1977) in their "characteristic parsing scheme" (in fact, this scheme is even more general in that it also yields non-LR parsers such as strict deterministic parsers as special cases).

In what follows we use an analogous strategy to capture the essence of LL parser construction: the relation ρ will be an arbitrary equivalence relation on the *viable suffixes* that satisfies the above three conditions. Particular relations ρ will give rise to particular LL parsers, such as canonical LL(k) or strong LL(k) parsers.

Formally, we say that a mapping ρ is an LL-*equivalence* if it maps every grammar G to an equivalence relation $\rho(G)$ on the viable suffixes of G such that the following conditions are satisfied: (1) $\rho(G)$ is of a finite index, (2) $\rho(G)$ is right invariant, and (3) $\gamma_1 : 1 = \gamma_2 : 1$ holds for $\rho(G)$-equivalent viable suffixes γ_1 and γ_2.

When G is understood we usually drop G and write ρ instead of $\rho(G)$, for short. Also, we feel free to speak of a relation ρ (rather than a relation-valued mapping ρ).

We now define the general notion of an "LL(ρ, k) parser," i.e., an LL(k) parser based on an arbitrary LL-equivalence ρ. In the definition, we need (for reasons to be explained later) \$-*augmented* versions of grammars, i.e., grammars augmented by adding a new production $S' \to \$S\$$, where S is the start symbol of the original grammar, \$ is a new terminal, and S' is a new nonterminal, the start symbol of the augmented grammar. If γ is a viable suffix, we denote by $[\gamma]_\rho$ its ρ-equivalence class.

Let ρ be an LL-equivalence, k a positive integer[1] and G a reduced grammar. We say that a rule of the form

$$[\delta a]_\rho \mid ay \to \mid y \tag{4.1}$$

is an LL(ρ, k) *shift action* of G by terminal a if δa is a viable suffix of the \$-augmented grammar G' for G, a is a terminal of G and y is a terminal string of G' such that ay is in $\mathrm{FIRST}_k(\gamma^R)$ for some viable suffix γ ρ-equivalent to δa.

Furthermore, we say that a rule of the form

$$[\delta]_\rho[\delta A]_\rho \mid y \to [\delta]_\rho[\delta X_n]_\rho \cdots [\delta X_n \cdots X_1]_\rho \mid y \tag{4.2}$$

is an LL(ρ, k) *produce action* of G by production $A \to X_1 \cdots X_n$ if δA is a viable suffix of G', $A \to X_1 \cdots X_n$ is a production of G ($X_1,\dots, X_n$ are single symbols) and y is in $\mathrm{FIRST}_k(\gamma^R)$ for some viable suffix γ ρ-equivalent to $\delta X_n \cdots X_1$. If $n=0$, i.e., $X_1 \cdots X_n = \varepsilon$, then we stipulate that $[\delta X_n]_\rho \cdots [\delta X_n \cdots X_1]_\rho$ means ε, too. Observe that Fact 3.7 guarantees that δ and each $\delta X_n \cdots X_i$, $i = 1,\dots, n$, is a viable suffix.

Let M be a push-down automaton in which the set of productions consists of all the LL(ρ, k) shift and produce actions of G and where the input vocabulary is the terminal vocabulary of G, the bottom-of-stack marker is $[\$]_\rho$, the end-of-input marker is \$, the initial stack contents are $[\$S]_\rho$ and the final stack contents are ε. We say that the pair (M, τ) is the LL(ρ, k) *parser* of G if τ is the homomorphism that maps every shift action to ε and every produce action by production p to p. Observe that τ is well defined because $\gamma_1 : 1 = \gamma_2 : 1$ holds, by definition, for all ρ-equivalent viable suffixes γ_1 and γ_2.

In what follows we prove that the LL(ρ, k) parser indeed is a left parser of G, as defined in Section 2. For this purpose we make the following notational convention: if ϕ is a string of ρ-equivalence classes we denote by $\bar{\phi}$ the string formed by the last symbols of representatives in the classes of ϕ, i.e., $\bar{\varepsilon} = \varepsilon$ and $\overline{\phi[\gamma]_\rho} = \bar{\phi}(\gamma : 1)$. Observe that, like the mapping τ above, the operator $\phi \curvearrowright \bar{\phi}$, too, is well defined.

[1] For simplicity we omit the trivial case $k = 0$.

LEMMA 4.1. *If in the* LL(ρ, k) *parser* (M, τ) *of a reduced grammar* G

$$\phi_1 \mid y_1 \underset{M}{\overset{\pi'}{\Longrightarrow}} \Phi, \tag{a}$$

then for some strings x, y_2, *and* ϕ_2

$$y_1 = xy_2, \qquad \Phi = \phi_2 \mid y_2,$$
$$|\pi'| = |\tau(\pi')| + |x| \qquad \textit{and} \qquad \bar{\phi}_1^{\mathrm{R}} \underset{lm}{\overset{\tau(\pi')}{\Longrightarrow}} x\bar{\phi}_2^{\mathrm{R}}. \tag{b}$$

Proof. The proof is by induction on the length $|\pi'|$ of the action string π'. We first assume, as the basis of the induction, that $\pi' = \varepsilon$. But then $\Phi = \phi_1 \mid y_1$, and conditions (b) hold if we choose $x = \varepsilon$, $y_2 = y_1$, and $\phi_2 = \phi_1$.

We then assume that $\pi' \neq \varepsilon$ and, as an induction hypothesis, that the lemma holds for action strings shorter than π'. Then π' is of the form $r'\pi''$, where r' is a single action. If r' is a produce action by some production $r = A \to X_1 \cdots X_n$, then (a) implies that for some δ, α, and ψ_1

$$\phi_1 \mid y_1 = \alpha[\delta]_\rho[\delta A]_\rho \mid y_1 \underset{M}{\overset{r'}{\Longrightarrow}} \alpha[\delta]_\rho[\delta X_n]_\rho \cdots [\delta X_n \cdots X_1]_\rho \mid y_1$$

$$= \psi_1 \mid y_1 \underset{M}{\overset{\pi''}{\Longrightarrow}} \Phi. \tag{1}$$

We then have

$$\bar{\phi}_1^{\mathrm{R}} = A(\overline{\alpha[\delta]_\rho})^{\mathrm{R}} \underset{lm}{\overset{r}{\Longrightarrow}} X_1 \cdots X_n(\overline{\alpha[\delta]_\rho})^{\mathrm{R}} = \bar{\psi}_1^{\mathrm{R}}. \tag{2}$$

On the other hand, by applying the induction hypothesis to the latter derivation segment in (1) we can conclude that for some x, y_2, and ϕ_2

$$y_1 = xy_2, \qquad \Phi = \phi_2 \mid y_2, \qquad |\pi''| = |\tau(\pi'')| + |x|,$$

and

$$\bar{\psi}_1^{\mathrm{R}} \underset{lm}{\overset{\tau(\pi'')}{\Longrightarrow}} x\bar{\phi}_2^{\mathrm{R}}. \tag{3}$$

By combining (2) and (3) it is then easy to see that conditions (b) hold. Note that, by definition, $r = \tau(r')$ and $\tau(r')\,\tau(\pi'') = \tau(r'\pi'')$.

We have yet to consider the case in which r' is a shift action by terminal a. Then condition (a) implies that for some δ, α, and z

$$\phi_1 \mid y_1 = \alpha[\delta a]_\rho \mid az \underset{M}{\overset{r'}{\Longrightarrow}} \alpha \mid z \underset{M}{\overset{\pi''}{\Longrightarrow}} \Phi. \tag{4}$$

Thus $\bar{\phi}_1^{\mathrm{R}} = a\bar{\alpha}^{\mathrm{R}}$, and we can conclude, by applying the induction hypothesis to the latter derivation segment in (4), that for some x', y_2, and ϕ_2

$$\begin{gathered} z = x'y_2, \qquad \Phi = \phi_2 \mid y_2, \\ |\pi''| = |\tau(\pi'')| + |x'|, \qquad \text{and} \qquad \bar{\alpha}^{\mathrm{R}} \xrightarrow[lm]{\tau(\pi'')} x'\bar{\phi}_2^{\mathrm{R}} \end{gathered} \tag{5}$$

Conditions (b) then hold if we choose $x = ax'$. Note that $\tau(r') = \varepsilon$ and that $y_1 = az$. ∎

LEMMA 4.2. *If (M, τ) is the LL(ρ, k) parser of a reduced grammar G, then $L(M) \subset L(G)$, and $\tau(\pi')$ is a left parse of w in G whenever π' is a parse of w in M. Moreover,* $\mathrm{TIME}_G(w) \leqslant \mathrm{TIME}_M(w) - |w|$.

Proof. Choose $\phi_1 = [\$]_\rho[\$S]_\rho$, $y_1 = w\$$ and $\Phi = [\$]_\rho \mid \$$ in Lemma 4.1. ∎

LEMMA 4.3. *Let X be a symbol and x a terminal string of a reduced grammar G, y a terminal string, δX and γX viable suffixes of the \$-augmented grammar G' for G, and ϕ a string of ρ-equivalence classes of G'. If*

$$X \xrightarrow[lm]{\pi} x, \qquad k : y \in \mathrm{FIRST}_k(\gamma^{\mathrm{R}}), \qquad \textit{and} \qquad \gamma \;\rho\; \delta, \tag{a}$$

then in the LL(ρ, k) *parser (M, τ) of G*

$$\phi[\delta]_\rho[\delta X]_\rho \mid xy \xrightarrow[lm]{\pi'} \phi[\delta]_\rho \mid y, \quad \tau(\pi') = \pi, \quad \textit{and} \quad |\pi'| = |\pi| + |x| \tag{b}$$

for some action string π'.

Proof. The proof is by induction on the length $|\pi|$ of the production string π. We first assume, as the basis of the induction, that $\pi = \varepsilon$. Then $X = x$ and $k : xy$ is in $\mathrm{FIRST}_k(X\gamma^{\mathrm{R}})$. Since, by the right invariance of ρ, γX is ρ-equivalent to δX, we can conclude that M has a shift action $r' = [\delta X]_\rho \mid xy' \to \mid y'$, where $xy' = k : xy$. But then conditions (b) hold if we choose $\pi' = r'$.

We then assume that $\pi \neq \varepsilon$ and, as an induction hypothesis, that the lemma holds for all production strings shorter that π. Then G has a production $r = X \to X_1 \cdots X_n$ $(n \geqslant 0)$, production strings $\pi_1, \ldots, \pi_n$ and terminal strings $x_1, \ldots, x_n$ such that

$$\pi = r\pi_1 \cdots \pi_n, \quad X_i \xrightarrow[lm]{\pi_i} x_i \quad \text{for all } i \quad \text{and} \quad x_1 \cdots x_n = x. \tag{1}$$

Thus

$$k:x_i \cdots x_n y \in \mathrm{FIRST}_k(X_i \cdots X_n \gamma^{\mathrm{R}}) \qquad \text{for all } i. \tag{2}$$

By Fact 3.7, $\delta X_n \cdots X_i$ and $\gamma X_n \cdots X_i$ are viable suffixes for all i. By the right invariance of ρ, $\delta X_n \cdots X_i$ is ρ-equivalent to $\gamma X_n \cdots X_i$ for all i. In the case $i=1$ condition (2) implies that M has a produce action $r' = [\delta]_\rho[\delta X]_\rho \mid y' \to [\delta]_\rho[\delta X_n]_\rho \cdots [\delta X_n \cdots X_1]_\rho \mid y'$, where $y' = k:xy$. Thus,

$$\phi[\delta]_\rho[\delta X]_\rho \mid xy \xRightarrow{r'} \phi[\delta]_\rho[\delta X_n]_\rho \cdots [\delta X_n \cdots X_1]_\rho \mid xy, \quad \text{and} \quad \tau(r') = r \tag{3}$$

hold in (M, τ). On the other hand, by applying the induction hypothesis, in the case of each i, to symbol X_i, terminal strings x_i and $y_i = x_{i+1} \cdots x_n y$, viable suffix $\delta_i = \delta X_n \cdots X_{i+1}$, and string $\phi_i[\delta_i]_\rho = \phi[\delta]_\rho[\delta X_n]_\rho \cdots [\delta X_n \cdots X_{i+1}]_\rho$, we conclude that for all i there is a production string π_i' satisfying

$$\phi[\delta]_\rho[\delta X_n]_\rho \cdots [\delta X_n \cdots X_i]_\rho \mid x_i \cdots x_n y$$

$$\xRightarrow{\pi_i'} \phi[\delta]_\rho[\delta X_n]_\rho \cdots [\delta X_n \cdots X_{i+1}]_\rho \mid x_{i+1} \cdots x_n y,$$

$$\tau(\pi_i') = \pi_i, \qquad \text{and} \qquad |\pi_i'| = |\pi_i| + |x_i|.$$

Conditions (b) hold if we choose $\pi' = r'\pi_1' \cdots \pi_n'$. ∎

LEMMA 4.4. *If (M, τ) is the* LL(ρ, k) *parser of a reduced grammar G, then $L(G) \subset L(M)$, and for any left parse π of w in G, $\tau(\pi') = \pi$ for some parse π' of w in M. Moreover,* $\mathrm{TIME}_M(w) \leqslant \mathrm{TIME}_G(w) + |w|$.

Proof. Choose $X = S$, $x = w$, $y = \$$, $\delta = \gamma = \$$, and $\phi = \varepsilon$ in Lemma 4.3. ∎

By Lemmas 4.2 and 4.4 we have

THEOREM 4.5. *For any* LL*-equivalence ρ and integer $k \geqslant 1$, the* LL(ρ, k) *parser (M, τ) of a reduced grammar G is a left parser of G. Moreover, for each sentence w in $L(G)$,* $\mathrm{TIME}_M(w) = \mathrm{TIME}_G(w) + |w|$. ∎

We now investigate on which conditions the LL(ρ, k) parser of a given grammar G is deterministic.

We say that a grammar G is an LL(ρ, k) *grammar* if the conditions

$$S \overset{*}{\underset{lm}{\Longrightarrow}} xA\delta \underset{lm}{\Longrightarrow} x\omega_1\delta,$$

$$S \overset{*}{\underset{lm}{\Longrightarrow}} xA\delta \underset{lm}{\Longrightarrow} x\omega_2\delta,$$

$$\gamma_1 \,\rho\, \delta^{\mathrm{R}}\omega_1^{\mathrm{R}}, \qquad \gamma_2 \,\rho\, \delta^{\mathrm{R}}\omega_2^{\mathrm{R}},$$

and

$$\mathrm{FIRST}_k(\gamma_1^{\mathrm{R}}) \cap \mathrm{FIRST}_k(\gamma_2^{\mathrm{R}}) \neq \varnothing$$

always imply that $\omega_1 = \omega_2$.

We note in passing that the reflexivity of an LL-equivalence implies immediately

FACT 4.6. *A grammar is* LL(ρ, k) *only if it is* LL(k). ∎

The following fact follows immediately from the definition and provides a sufficient condition for the converse of Fact 4.6.

FACT 4.7. *If* $\mathrm{FIRST}_k(\gamma) = \mathrm{FIRST}_k(\delta)$ *holds in* G *whenever* γ *and* δ *are* ρ*-equivalent viable suffixes, then* G *is* LL(ρ, k) *whenever it is* LL(k). ∎

The following theorem delineates the correspondence between LL(ρ, k) grammars and deterministic LL(ρ, k) parsers.

THEOREM 4.8. *The* \$*-augmented grammar* G' *of a reduced grammar* G *is* LL(ρ, k) *if and only if the* LL(ρ, k) *parser of* G *is deterministic.*

Proof. First, observe that nondeterminism can occur in the LL(ρ, k) parser only between produce actions by productions of the same nonterminal and that these produce actions must have the same left-hand side. That is, the conflicting parsing actions must be of the forms

$$\begin{aligned}
&[\delta]_\rho[\delta A]_\rho \mid y \to [\delta]_\rho[\delta X_m]_\rho \cdots [\delta X_m \cdots X_1]_\rho \mid y, \\
&[\delta]_\rho[\delta A]_\rho \mid y \to [\delta]_\rho[\delta Y_n]_\rho \cdots [\delta Y_n \cdots Y_1]_\rho \mid y,
\end{aligned} \tag{1}$$

where $A \to X_1 \cdots X_m$ and $A \to Y_1 \cdots Y_n$ are productions of G. That the actions must be produce actions by productions of the same nonterminal follows from the property of ρ that ρ-equivalent viable suffixes always end with the same symbol. The \$-augmentation of G, in turn, guarantees that the lookahead strings in both actions must be equal: if γ is a viable suffix of the \$-augmented grammar G' and y is in $\mathrm{FIRST}_k(\gamma^{\mathrm{R}})$, then either $|y| = k$ or y ends with \$ (note that without the \$-augmentation the lookahead string in one of the actions might be a proper prefix of that in the other).

The existence of the pair of produce actions (1) implies, by definition, the existence of viable suffixes γ_1 and γ_2 such that

$$\gamma_1 \,\rho\, \delta X_m \cdots X_1, \quad \gamma_2 \,\rho\, \delta Y_n \cdots Y_1, \quad \text{and} \quad y \in \mathrm{FIRST}_k(\gamma_1^{\mathrm{R}}) \cap \mathrm{FIRST}_k(\gamma_2^{\mathrm{R}}). \tag{2}$$

Since δA is a viable suffix of G' and G is reduced, we have, for some terminal string x

$$S' \underset{lm}{\overset{*}{\Longrightarrow}} xA\delta^{R} \underset{lm}{\Longrightarrow} xX_1 \cdots X_m \delta^{R},$$
$$S' \underset{lm}{\overset{*}{\Longrightarrow}} xA\delta^{R} \underset{lm}{\Longrightarrow} xY_1 \cdots Y_n \delta^{R}. \tag{3}$$

This means that the actions (1) are equal if G' is LL(ρ, k). Conversely, if conditions (2) and (3) hold, then G must have the pair of LL(ρ, k) produce actions (1). These actions can coincide only if $X_1 \cdots X_m = Y_1 \cdots Y_n$, due to the definition of ρ. ∎

5. Construction of LL Parsers

In this section we consider specific LL(ρ, k) parsers and their construction. In particular, we give a new construction method for the canonical LL(k) parser. This method is based on the "item" approach and can be regarded as a dual of Knuth's method for constructing canonical LR(k) parsers.

We say that a pair $[A \to \alpha . \beta, y]$ is a *k-item* $(k \geqslant 0)$ of a grammar G if $A \to \alpha\beta$ is a production of G, y is a terminal string of length k or less, and if the dot (which marks a position in the right-hand side) is a special symbol that does not appear in the vocabulary of G. The dotted production $A \to \alpha . \beta$ is the *core* of the item, and y is its *lookahead string.*

We call a k-item $[A \to \alpha . \beta, y]$ an LR(k)-item if y is in $\mathrm{FOLLOW}_k(A)$, and an LL(k)-item if y is in $\mathrm{FIRST}_k(\beta\, \mathrm{FOLLOW}_k(A))$. Recall that an LR$(k)$-item of the form $[A \to \omega ., y]$ indicates that the canonical LR(k) parser has a reduce action by production $A \to \omega$ in the case of lookahead y. Analogously, an LL(k)-item of the form $[A \to . \omega, y]$ will indicate that the canonical LL(k) parser has a produce action by production $A \to \omega$ in the case of lookahead y.

We say that an item $[A \to \alpha . \beta, y]$ of a grammar G is LL(k)-*valid* for a string γ of G if

$$S \underset{lm}{\overset{*}{\Longrightarrow}} xA\delta \underset{lm}{\Longrightarrow} x\alpha\beta\delta = x\alpha\gamma^{R} \qquad \text{and} \qquad y \in \mathrm{FIRST}_k(\gamma^{R})$$

hold in G for some terminal string x and string δ.

We denote by $V_k(\gamma)$ the set of all LL(k)-valid items for γ. We have

FACT 5.1. *$V_k(\gamma)$ is nonempty if and only if γ is a viable suffix and $\mathrm{FIRST}_k(\gamma^{R})$ is nonempty.* ∎

We say that viable suffixes γ_1 and γ_2 are LL(k)-*equivalent*, written $\gamma_1 \rho_k \gamma_2$, if $V_k(\gamma_1) = V_k(\gamma_2)$.

Clearly, the LL(k)-equivalence ρ_k is an equivalence relation. Moreover, it is of a finite index; the set $V_k(\gamma)$ can be regarded as a finite representation of the LL(k)-equivalence class $[\gamma]_k$ (we write $[\gamma]_k$ instead of $[\gamma]_{\rho_k}$, for short).

To establishing that ρ_k is an LL-equivalence it remains to be shown that it is right invariant and that $\gamma_1 : 1 = \gamma_2 : 1$ holds whenever $\gamma_1 \rho_k \gamma_2$. We do this by giving an explicit construction algorithm for the sets $V_k(\gamma)$. We call the collection of all sets $V_k(\gamma)$, for fixed k, the *canonical* LL(k)-*collection* for the grammar.

First, we call each k-item $[B \to \omega., y]$ an *immediate* LL(k)-*descendant* of a k-item $[A \to \alpha B.\beta, y]$ (or the latter an *immediate* LL(k)-*ancestor* of the former) and write

$$[A \to \alpha B.\beta, y]\, D_k [B \to \omega., y].$$

Observe that, unlike in the LR(k) construction, the dot is located *to the right* of B and ω, and that the lookahead string is *the same* in both items.

The class $V_k(\varepsilon)$ will be obtained as the closure under D_k of the set

$$\{[S \to \omega., \varepsilon] \mid S \to \omega \text{ is a production of the start symbol } S\}.$$

Observe again that the dot is placed first at the rightmost position in the right-hand side of start productions.

If q is a set of k-items and X is a single symbol, we define the *basis of the X-successor* of q to be the set

$$B_k(q, X) = \{[A \to \alpha.X\beta, z] \mid [A \to \alpha X.\beta, y] \in q, z \in \mathrm{FIRST}_k(Xy)\}.$$

Observe that the dot is moved *from right to left* and that the lookahead string y is *changed* to z. As might be expected, $V_k(\gamma X)$ will be obtained as the closure under D_k of the set $B_k(V_k(\gamma), X)$.

As an example, consider the grammar G_1 with productions (Aho and Ullman, 1972)

$$S \to aAaa \mid bAba,$$
$$A \to b \mid \varepsilon.$$

This grammar is a prototype of an LL(2) grammar that is not strong LL(2). The canonical LL(2) collection for the \$-augmented grammar G_1' for G_1 is depicted in Fig. 1 as a directed graph.

Properties of LL(k)-valid items are most conveniently proved by induction on the length of the initial derivation segment $S \overset{*}{\Rightarrow}_{lm} xA\delta$ in the definition of

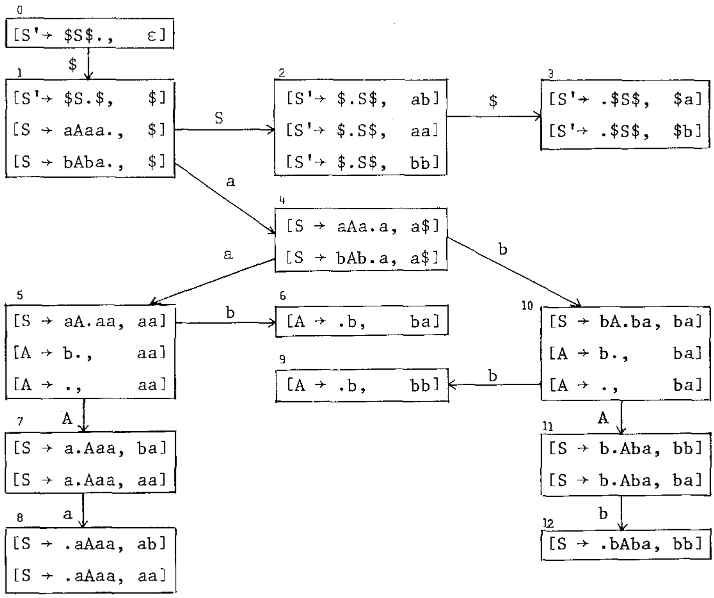

FIG. 1. The canonical LL(2) collection for the grammar G_1' with productions $S' \to \$S\$$, $S \to aAaa \mid bAba$, $A \to b \mid \varepsilon$. The graph has an edge labelled X from $V_2(\gamma)$ to $V_2(\gamma X)$ (γ is a viable suffix). The vertex labelled 0 is $V_2(\varepsilon)$.

LL(k)-validity. To make the proofs rigorous we define explicitly, for each integer $n \geqslant 0$, $V_{k,n}(\gamma)$ to be the set of k-items $[A \to \alpha . \beta, y]$ for which

$$S \xRightarrow[lm]{n} xA\delta \xRightarrow[lm]{} x\alpha\beta\delta = x\alpha\gamma^{\mathrm{R}} \qquad \text{and} \qquad y \in \mathrm{FIRST}_k(\gamma^{\mathrm{R}})$$

hold for some x and δ.

We have

FACT 5.2.

$$V_k(\gamma) = \bigcup_{n=0}^{\infty} V_{k,n}(\gamma).$$

FACT 5.3. *If $[A \to \alpha B . \beta, y]$ is an item in $V_{k,n}(\gamma)$ and if α derives in m steps some terminal string v, then, for all productions $B \to \omega$, $V_{k,n+m+1}(\gamma)$ contains the item $[B \to \omega ., y]$.*

Proof. By definition, we have, for some x and δ,

$$S \xRightarrow[lm]{n} xA\delta \xRightarrow[lm]{} x\alpha B\beta\delta = x\alpha B\gamma^{\mathrm{R}} \qquad \text{and} \qquad y \in \mathrm{FIRST}_k(\gamma^{\mathrm{R}}).$$

But then the condition $\alpha \Rightarrow_{lm}^{m} v$ implies that, for all $B \to \omega$,

$$S \xRightarrow[lm]{n+1+m} xvB\beta\delta \xRightarrow[lm]{} xv\omega\beta\delta = xv\omega\gamma^{\mathrm{R}},$$

i.e., $[B \to \omega., y]$ is in $V_{k,n+m+1}(\gamma)$. ∎

Facts 5.2 and 5.3 imply immediately

FACT 5.4. *In a reduced grammar, each* $V_k(\gamma)$ *is closed under* D_k, *i.e.,* $D_k^*(V_k(\gamma)) = V_k(\gamma)$.

FACT 5.5. *If* $n > 0$ *and* $[B \to \omega., y]$ *is an item in* $V_{k,n}(\gamma)$, *then, for some* $m < n$, $V_{k,m}(\gamma)$ *contains an item* $[A \to \alpha B.\beta, y]$, *where* α *derives in* $n-m-1$ *steps some terminal string* v.

Proof. By definition, we have, for some x

$$S \xRightarrow[lm]{n} xB\gamma^{\mathrm{R}} \xRightarrow[lm]{} x\omega\gamma^{\mathrm{R}} \qquad \text{and} \qquad y \in \mathrm{FIRST}_k(\gamma^{\mathrm{R}}).$$

By choosing in Lemma 3.4 $\eta = \varepsilon$ and $\delta = \gamma^{\mathrm{R}}$ we can then conclude that

$$S \xRightarrow[lm]{m} x'A\delta' \xRightarrow[lm]{} x'\alpha\beta'\delta' = x'\alpha B\gamma^{\mathrm{R}} \qquad \text{and} \qquad x'\alpha \xRightarrow[lm]{n-m-1} x$$

for some $m < n$, x', δ' and $A \to \alpha\beta'$, where $1:\beta' = 1:B\gamma^{\mathrm{R}}$. Thus, β' is of the form $B\beta$, and $[A \to \alpha B.\beta, y]$ is in $V_{k,m}(\gamma)$. Moreover, α derives in $n-m-1$ steps some suffix v of x. ∎

The following fact follows immediately from the definition of $V_{k,0}(\gamma)$.

FACT 5.6. $V_{k,0}(\gamma) = \{[S \to \alpha.\gamma^{\mathrm{R}}, y] \mid S \to \alpha\gamma^{\mathrm{R}}$ *is a production of the start symbol* S *of* G *and* $y \in \mathrm{FIRST}_k(\gamma^{\mathrm{R}})\}$.

We say that a k-item $[A \to \alpha.\beta, y]$ is LL-*essential* if $\beta \neq \varepsilon$. (Recall that $[A \to \alpha.\beta, y]$ is LR-*essential*, i.e., essential in the LR sense, if $\alpha \neq \varepsilon$.) If q is set of k-items, then we denote by $E(q)$ the set of LL-essential items in q.

LEMMA 5.7. *If* $\gamma \neq \varepsilon$, *then, in a reduced grammar,* $E(V_k(\gamma))$ *spans* $V_k(\gamma)$ *under* D_k, *i.e.,* $D_k^*(E(V_k(\gamma))) = V_k(\gamma)$.

Proof. Fact 5.4 implies immediately that $D_k^*(E(V_k(\gamma)))$ is included in $V_k(\gamma)$. To prove the converse, it suffices, by Fact 5.2, to show that each $V_{k,n}(\gamma)$, $n \geqslant 0$, is included in $D_k^*(E(V_k(\gamma)))$. The proof is by induction on n. If $n = 0$ then, by Fact 5.6, $V_{k,n}(\gamma)$ even is included in $E(V_k(\gamma))$. Thus, we may assume that $n > 0$ and, as an induction hypothesis, that each $V_{k,m}(\gamma)$, $m < n$, is included in $D_k^*(E(V_k(\gamma)))$. Let $[B \to \omega.\delta, y]$ be an item in $V_{k,n}(\gamma)$. If $\delta \neq \varepsilon$, $[B \to \omega.\delta, y]$ even is in $E(V_k(\gamma))$. Otherwise, we may conclude, by Fact 5.5, that some $V_{k,m}(\gamma)$, $m < n$, contains an item $[A \to \alpha B.\beta, y]$. By the induction hypothesis, this item is in $D_k^*(E(V_k(\gamma)))$; so is its immediate descendant $[B \to \omega.\delta, y]$. ∎

LEMMA 5.8. *In a reduced grammar, the set*

$$I = \{[S \to \omega., \varepsilon] \mid S \to \omega \text{ is a production of the start symbol } S \text{ of the grammar}\}$$

spans $V_k(\varepsilon)$ *under* D_k, *i.e.*, $D_k^*(I) = V_k(\varepsilon)$.

Proof. Analogous to that of Lemma 5.7. ∎

FACT 5.9. *If* $[A \to \alpha\omega.\beta, y]$ *is an item in* $V_{k,n}(\gamma)$, *then* β *is a prefix of* γ^{R}, $\gamma\omega^{\mathrm{R}}$ *is a viable suffix, and* $[A \to \alpha.\omega\beta, z]$ *is in* $V_{k,n}(\gamma\omega^{\mathrm{R}})$ *for all* z *in* $\mathrm{FIRST}_k(\omega y)$.

Proof. By definition, we have, for some x and δ,

$$S \underset{lm}{\overset{n}{\Longrightarrow}} xA\delta \underset{lm}{\Longrightarrow} x\alpha\omega\beta\delta = x\alpha\omega\gamma^{\mathrm{R}} \qquad \text{and} \qquad y \in \mathrm{FIRST}_k(\gamma^{\mathrm{R}}).$$

Thus, β is a prefix of γ^{R}, and γ^{R} derives some terminal string y' such that $k{:}y' = y$. If z is a string in $\mathrm{FIRST}_k(\omega y)$, then ω derives a terminal string v such that $k{:}vy = z$. Because $k{:}vy = k{:}v(k{:}y') = k{:}vy' \in \mathrm{FIRST}_k(\omega\gamma^{\mathrm{R}})$, we thus have

$$S \underset{lm}{\overset{n}{\Longrightarrow}} xA\delta \underset{lm}{\Longrightarrow} x\alpha\omega\beta\delta = x\alpha(\gamma\omega^{\mathrm{R}})^{\mathrm{R}} \qquad \text{and} \qquad z \in \mathrm{FIRST}_k((\gamma\omega^{\mathrm{R}})^{\mathrm{R}}),$$

which means that $\gamma\omega^{\mathrm{R}}$ is a viable suffix and that $[A \to \alpha.\omega\beta, z]$ is in $V_{k,n}(\gamma\omega^{\mathrm{R}})$. ∎

FACT 5.10. *If* $[A \to \alpha.\omega\beta, z]$ *is an item in* $V_{k,n}(\gamma)$, *then there is a viable suffix* γ_1 *and a terminal string* y *such that* $\gamma = \gamma_1\omega^{\mathrm{R}}$, $[A \to \alpha\omega.\beta, y]$ *is in* $V_{k,n}(\gamma_1)$ *and* z *is in* $\mathrm{FIRST}_k(\omega y)$.

Proof. By definition, we have, for some x and δ,

$$S \underset{lm}{\overset{n}{\Longrightarrow}} xA\delta \underset{lm}{\Longrightarrow} x\alpha\omega\beta\delta = x\alpha\gamma^{\mathrm{R}} \qquad \text{and} \qquad z \in \mathrm{FIRST}_k(\gamma^{\mathrm{R}}).$$

If we denote $\gamma_1 = (\beta\delta)^R$, then $\gamma^R = \omega\gamma_1^R$, and ω derives a terminal string v and γ_1^R a terminal string y' such that $k:vy' = z$. If we denote $y = k:y'$, we then have

$$S \overset{n}{\underset{lm}{\Longrightarrow}} xA\delta \underset{lm}{\Longrightarrow} x\alpha\omega\beta\delta = x\alpha\omega\gamma_1^R \qquad \text{and} \qquad y \in \mathrm{FIRST}_k(\gamma_1^R),$$

which means that γ_1 is a viable suffix and that $[A \to \alpha\omega.\beta, y]$ is in $V_{k,n}(\gamma_1)$. Moreover, $z = k:vy' = k:v(k:y') = k:vy \in \mathrm{FIRST}_k(\omega y)$. ∎

LEMMA 5.11. $E(V_k(\gamma X)) = B_k(V_k(\gamma), X)$.

Proof. Any item in $B_k(V_k(\gamma), X)$ is of the form $[A \to \alpha.X\beta, z]$ such that, for some y, $[A \to \alpha X.\beta, y]$ is in $V_k(\gamma)$ and z is in $\mathrm{FIRST}_k(Xy)$. By Facts 5.2 and 5.9, $[A \to \alpha.X\beta, z]$ is in $V_k(\gamma X)$. Thus $B_k(V_k(\gamma), X)$ is included in $E(V_k(\gamma X))$. Conversely, if $[A \to \alpha.Y\beta, z]$ is an item in $V_k(\gamma X)$, then by Facts 5.2 and 5.10, $Y = X$ and, for some y, $[A \to \alpha X.\beta, y]$ is in $V_k(\gamma)$ and z is in $\mathrm{FIRST}_k(Xy)$. Thus, $[A \to \alpha.Y\beta, z]$ is in $B_k(V_k(\gamma), X)$, which means that $E(V_k(\gamma X))$ is included in $B_k(V_k(\gamma), X)$. ∎

Lemmas 5.11 and 5.7 immediately imply

LEMMA 5.12. *In a reduced grammar,* $V_k(\gamma X) = D_k^*(B_k(V_k(\gamma), X))$. ∎

We can now prove

THEOREM 5.13. *In the case of a reduced grammar, the* LL(k)-*equivalence is always an* LL-*equivalence.*

Proof. First note that Lemma 5.12 implies immediately the right invariance of the LL(k)-equivalence. To prove that $\gamma_1:1 = \gamma_2:1$ for LL(k)-equivalent viable suffixes γ_1 and γ_2, we note that, by the definition of LL(k)-validity, $V_k(\varepsilon)$ cannot contain LL-essential items and, by Fact 5.1, in a reduced grammar $V_k(\gamma)$ is nonempty for all viable suffixes γ. Because Lemma 5.12 then implies that any $V_k(\gamma X)$, X a single symbol, contains at least one LL-essential item and that each LL-essential item in $V_k(\gamma X)$ is of the form $[A \to \alpha.X\beta, y]$, we can conclude that $[\varepsilon]_k = \{\varepsilon\}$ and that whenever $V_k(\gamma_1 X) = V_k(\gamma_2 Y)$ for viable suffixes $\gamma_1 X$ and $\gamma_2 Y$, then $X = Y$. ∎

By the definition of LL(k)-validity we have

FACT 5.14. *If* $V_k(\gamma)$ *contains at least one item with the core* $A \to \alpha.\beta$, *then* $\{y \mid [A \to \alpha.\beta, y] \in V_k(\gamma)\} = \mathrm{FIRST}_k(\gamma^R)$.

By the definition of LL(k)-equivalence and by Facts 5.1 and 5.14 we have

LEMMA 5.15. *If* γ *and* δ *are* LL(k)*-equivalent viable suffixes, then* $\mathrm{FIRST}_k(\gamma^{\mathrm{R}}) = \mathrm{FIRST}_k(\delta^{\mathrm{R}})$.

If q is a set of items and $m \geqslant 0$, we denote by $m:q$ the set of all items $[A \to \alpha.\beta, m:y]$ for which $[A \to \alpha.\beta, y]$ is in q. That is, $m:q$ is obtained from the items in q by truncating the lookahead strings.

The definition of LL(k)-validity implies immediately

FACT 5.16. $m:V_k(\gamma) = V_m(\gamma)$ *whenever* $k \geqslant m$.

Let ρ_m be the LL(m)-equivalence and $k \geqslant m$. We call the LL(ρ_m, k) parser of a grammar G the LA(k) LL(m) *parser* of G. In particular, we call the LA(k) LL(k) parser the *canonical* LL(k) *parser*, and the LA(k) LL(0) parser the LALL(k) *parser*. Correspondingly, we call LL(ρ_m, k) grammars LA(k) LL(m) *grammars*, and, in particular, LA(k) LL(0) grammars LALL(k) *grammars*.

The following theorem implies a construction algorithm for the LA(k) LL(m) parser.

THEOREM 5.17. *Let* G *be a reduced grammar. Then the* LA(k) LL(m) *parser of* G *has a shift action of the form*

$$[\delta a]_m \mid ay \to \mid y \tag{a}$$

if and only if a *is a terminal of* G *and, for some string* γ *in the augmented grammar* G', $m:V_k(\gamma) = m:V_k(\delta a)$ *and* $V_k(\gamma)$ *contains an item of the form* $[A \to \alpha.a\beta, ay]$.

Correspondingly, LA(k) LL(m) *parser has a produce action of the form*

$$[\delta]_m[\delta A]_m \mid y \to [\delta]_m[\delta X_n]_m \cdots [\delta X_n \cdots X_1]_m \mid y \tag{b}$$

if and only if $A \to X_1 \cdots X_n$ *is a production of* G *and, for some string* γ *in* G', $m:V_k(\gamma) = m:V_k(\delta X_n \cdots X_1)$ *and* $V_k(\gamma)$ *contains the item* $[A \to .X_1 \cdots X_n, y]$.

Proof. If the parser has a shift action of the form (a), then a is a terminal of G and ay is in $\mathrm{FIRST}_k(\gamma^{\mathrm{R}})$ for some viable suffix γ LL(m)-equivalent to δa. By Fact 5.16, $m:V_k(\gamma) = m:V_k(\delta a)$. By Fact 5.1, $V_k(\gamma)$ is nonempty. By Lemma 5.12, $V_k(\gamma)$ contains an item with core $A \to \alpha.a\beta$. By Fact 5.14, $V_k(\gamma)$ contains $[A \to \alpha.a\beta, ay]$. Conversely, if a is a terminal of G and $m:V_k(\gamma) = m:V_k(\delta a)$, and if $V_k(\gamma)$ contains $[A \to \alpha.a\beta, ay]$, then γ is a viable suffix, ay is in $\mathrm{FIRST}_k(\gamma^{\mathrm{R}})$ and, by Fact 5.16, γ is LL(m)-equivalent to δa. Thus, the parser has a shift action of the form (a). This proves the first part of the theorem.

To prove the second part, let the parser have a produce action of the form (b). Then δA is a viable suffix, $A \to X_1 \cdots X_n$ is a production of G and y is in $\mathrm{FIRST}_k(\gamma^{\mathrm{R}})$ for some viable suffix γ LL(m)-equivalent to $\delta X_n \cdots X_1$. By Fact 5.16, $m : V_k(\gamma) = m : V_k(\delta X_n \cdots X_1)$. By Fact 5.1, $V_k(\delta A)$ is nonempty. By Lemma 5.12, $V_k(\delta)$ contains an item with core $B \to \alpha A . \beta$. By Facts 5.3 and 5.9, $V_k(\delta X_n \cdots X_1)$ contains an item with the core $A \to . X_1 \cdots X_n$. By Fact 5.16, $V_m(\gamma)$ and $V_k(\gamma)$ contain an item with the core $A \to . X_1 \cdots X_n$. By Fact 5.14, $V_k(\gamma)$ contains $[A \to . X_1 \cdots X_n, y]$. Conversely, if $A \to X_1 \cdots X_n$ is a production of G and $m : V_k(\gamma) = m : V_k(\delta X_n \cdots X_1)$, and if $V_k(\gamma)$ contains $[A \to . X_1 \cdots X_n, y]$, then γ is a viable suffix, y is in $\mathrm{FIRST}_k(\gamma^{\mathrm{R}})$ and, by Fact 5.16, γ is LL(m)-equivalent to $\delta X_n \cdots X_1$. Moreover, by Fact 5.16, $V_m(\delta X_n \cdots X_1)$ contains an item with the core $A \to . X_1 \cdots X_n$. By Theorem 3.5, δA is a viable suffix. Thus, the parser has a produce action of the form (b). ∎

As an example, consider the canonical LL(2) parser (M, τ) of our grammar G_1 (the canonical LL(2) collection for the \$-augmented grammar G_1' for G_1 is depicted in Fig. 1). The parser has the following rules (we denote by q_i the class $[\gamma]_k$ if $V_k(\gamma)$ is labelled by i in Fig. 1):

rule p	$\tau(p)$
$q_1 q_2 \mid ab \to q_1 q_4 q_5 q_7 q_8 \mid ab$	$S \to aAaa$
$q_1 q_2 \mid aa \to q_1 q_4 q_5 q_7 q_8 \mid aa$	$S \to aAaa$
$q_1 q_2 \mid bb \to q_1 q_4 q_{10} q_{11} q_{12} \mid bb$	$S \to bAba$
$q_5 q_7 \mid ba \to q_5 q_6 \mid ba$	$A \to b$
$q_{10} q_{11} \mid bb \to q_{10} q_9 \mid bb$	$A \to b$
$q_5 q_7 \mid aa \to q_5 \mid aa$	$A \to \varepsilon$
$q_{10} q_{11} \mid ba \to q_{10} \mid ba$	$A \to \varepsilon$
$q_4 \mid a\$ \to \mid \$$	ε
$q_5 \mid aa \to \mid a$	ε
$q_6 \mid ba \to \mid a$	ε
$q_8 \mid ab \to \mid b$	ε
$q_8 \mid aa \to \mid a$	ε
$q_9 \mid bb \to \mid b$	ε
$q_{10} \mid ba \to \mid a$	ε
$q_{12} \mid bb \to \mid b$	ε

This parser is deterministic. The LALL(2) parser, on the contrary, is nondeterministic: the fact that $[\$aab]_0 = [\$abb]_0$ implies the existence of an additional, conflicting rule

$$q_{10} q_{11} \mid ba \to q_{10} q_9 \mid ba.$$

(Here the q_i's mean LL(0)-equivalence classes.)

By Facts 4.6 and 4.7 and by Lemma 5.15 we have

THEOREM 5.18. *A grammar is* LA(k) LL(k) *if and only if it is* LL(k). ∎

Since a grammar G clearly is LL(k) if and only if its \$-augmented version G' is LL(k), we have, by Theorems 4.8 and 5.18.

THEOREM 5.19. *A reduced grammar is* LL(k) *if and only if its canonical* LL(k) *parser is deterministic.* ∎

We leave the proof of the following theorem to the reader:

THEOREM 5.20. *The classes of strong* LL(1), LALL(1) *and* LL(1) *grammars are equal. For all* $0 \leqslant m \leqslant k \geqslant 1$, *the class of* LA($k$) LL($m$) *grammars is properly included in the class of* LA($k+1$) LL(m) *grammars, which in turn is properly included in the class of* LA($k+1$) LL($m+1$) *grammars.* ∎

6. COMPARISONS WITH OTHER APPROACHES

The usual way to construct a canonical LL(k) parser (see e.g., Rosenkrantz and Stearns, 1970; Aho and Ullman, 1972) involves pairs of the form $[X, R]$, where X is a grammar symbol and R is a subset of $\mathrm{FOLLOW}_k(X)$. The initial stack contents of the parser are $[S, \{\$\}]$ (S is the start symbol and \$ the end marker). For each pair $[a, R]$ and the lookahead string ay in $\mathrm{FIRST}_k(aR)$, the parser has the shift action

$$[a, R] \mid ay \to \mid y. \tag{6.1}$$

For each pair $[A, R]$, production $A \to X_1 \cdots X_n$ and lookahead string y in $\mathrm{FIRST}_k(X_1 \cdots X_n R)$, the parser has the produce action

$$[A, R] \mid y \to [X_n, R_n] \cdots [X_1, R_1] \mid y, \tag{6.2}$$

where $R_n = R$ and $R_i = \mathrm{FIRST}_k(X_{i+1} \cdots X_n R)$, $i < n$. (Actually, in the parser presented in Aho and Ullman (1972) each pair $[a, R]$, where a is a terminal, has been further replaced by the terminal itself. This has the effect of delaying error detection in the case $k > 1$.)

There is a close correspondence between this and our approach. In fact, the above parser can easily be obtained from the canonical LL(k) collection presented in Section 5. The construction relies on the fact that for each $V_k(\gamma)$ the set

$$\mathrm{LA}(V_k(\gamma)) = \{y \mid V_k(\gamma) \text{ contains an item with lookahead string } y\}$$

equals the set $\mathrm{FIRST}_k(\gamma^{\mathrm{R}})$. (This is an immediate consequence of Fact 5.14.)

We state without proof

THEOREM 8.1. *The conventional canonical* LL(k) *parser has a shift action of the form* (6.1) *if and only if a is a terminal and, for some string δ, $R = \mathrm{LA}(V_k(\delta))$ and ay is in $\mathrm{LA}(V_k(\delta a))$. The parser has a produce action of the form* (6.2) *if and only if $A \to X_1 \cdots X_n$ is a production and, for some string δ, $R = R_n = \mathrm{LA}(V_k(\delta))$, $R_i = \mathrm{LA}(V_k(\delta X_n \cdots X_{i+1}))$, $i < n$, and y is in $\mathrm{LA}(V_k(\delta X_n \cdots X_1))$.* ▮

Finally, we note that we could well have defined the notion of a general LL(ρ, k) parser using a push-down transducer in which the stack alphabet consists of pairs $[X, R]$ rather than ρ-equivalence classes $[\gamma]_\rho$. This parser has a shift action of the form (6.1) whenever a is a terminal and there is a viable suffix δa such that $R = \mathrm{FIRST}_k([\delta]_\rho^{\mathrm{R}})$ and ay is in $\mathrm{FIRST}_k([\delta a]_\rho^{\mathrm{R}})$ (here $\mathrm{FIRST}_k([\delta]_\rho^{\mathrm{R}})$ means the union of the sets $\mathrm{FIRST}_k(\gamma^{\mathrm{R}})$, $\gamma \rho \delta$). The parser has a produce action of the form (6.2) whenever $A \to X_1 \cdots X_n$ is a production and there is a viable suffix δA such that $R = R_n = \mathrm{FIRST}_k([\delta]_\rho^{\mathrm{R}})$ and $R_i = \mathrm{FIRST}_k([\delta X_n \cdots X_{i+1}]_\rho^{\mathrm{R}})$.

This approach is practical in that it leads to parsers that are slightly smaller than those in our original approach presented in Section 4. By the *size* of a rewriting system (V, P) we mean the sum of the lengths of its productions, i.e., the sum of all $|\omega_1 \omega_2|$, where $\omega_1 \to \omega_2$ is a production in P. In our original approach, i.e., when actions of the forms (4.1) and (4.2) are used, the canonical LL(k) parser is of size of the order $|G|^{k+1} \cdot 2^{|G|^{k+1}}$, where $|G|$ is the size of the grammar G in question (observe that the number of different item cores in G is $|G|$ and that the number of different lookahead strings of length k or less is at most $|G|^k$). If actions of the forms (6.1) and (6.2) are used, then the size of the canonical LL(k) parser is of the order $|G|^{k+1} \cdot 2^{|G|^k}$.

REFERENCES

AHO, A. V. AND ULLMAN, J. D. (1972), "The Theory of Parsing, Translation, and Compiling," Vol. I, Prentice–Hall, Englewood Cliffs, N. J.

DEUSSEN, P. (1979), One abstract accepting algorithm for all kinds of parsers, *in* "Automata, Languages and Programming, Sixth Colloquium, Graz, July 1979," Lecture Notes in Computer Science, No. 71, pp. 203–217, Springer-Verlag, Berlin/Heidelberg/New York.

FISCHER, C. N., MILTON, D. R., AND QUIRING, S. B. (1980), Efficient LL(1) error correction and recovery using only insertions, *Acta Inform.* **13**, 141–154.

GELLER, M. M. AND HARRISON, M. A. (1977), Characteristic parsing: A framework for producing compact deterministic parsers. I and II, *J. Comput. System Sci.* **14**, 265–317; 318–343.

HARRISON, M. A. (1978), "Introduction to Formal Language Theory," Addison–Wesley, Reading, Mass.

KNUTH, D. E. (1965), On the translation of languages from left to right, *Inform. and Control.* **8**, 607–639.

KNUTH, D. E. (1971), Top-down syntax analysis, *Acta Inform.* **1**, 79–110.

LEWIS, P. M., III AND STEARNS, R. E. (1968), Syntax-directed transductions, *J. Assoc. Comput. Mach.* **15**, 465–488.

ROSENKRANTZ, D. J. AND STEARNS, R. E. (1970), Properties of deterministic topdown grammars, *Inform. and Control.* **17**, 226–255.

SIPPU, S. AND SOISALON-SOININEN, E. (1980), Characterizations of the LL(k) property, *in* "Automata, Languages and Programming, Seventh Colloquium, Noordwijkerhout, July 1980," Lecture Notes in Computer Science, No. 85, pp. 596–608, Springer-Verlag, Berlin/Heidelberg/New York.