ON THE RELATIONSHIP BETWEEN THE LL(k) AND LR(k) GRAMMARS *

Anton NIJHOLT

Faculty of Science, Department of Informatics, Nijmegen University, 6525 ED Nijmegen, The Netherlands

Received 14 March 1981; revised version received 23 July 1982

In the literature various proofs of the inclusion of the class of LL(k) grammars into the class of LR(k) grammars can be found. Some of these proofs are not correct, others are informal, semi-formal or contain flaws. Some of them are correct but the proof is less straightforward than demonstrated here.

Keywords: Formal languages, LR(k) grammars, LL(k) grammars, parsing

1. Introduction

In [19] the LR(k) grammars were introduced. In [22] and [20] definitions of LL(k) grammars appear. The three main theoretical problems which are associated with these grammar classes are the following.

Problem A. The decidability of the equivalence problems for LL(k) and LR(k) grammars.

Problem B. The proof that each deterministic language can be generated by an LR(1) grammar.

Problem C. The proof that each LL(k) grammar is an LR(k) grammar.

This paper is concerned with Problem C. We give an alternative proof that each LL(k) grammar is an LR(k) grammar. There are good reasons to do so. Firstly, we note that some of the published proofs are poor from a mathematical point of view. The arguments which are used are sometimes intuitive and sometimes contain flaws. In [20] it is remarked that a careful and clear proof of this inclusion result would make a suitable master's thesis.

We give a short survey of the relevant literature on this inclusion problem.

From the transduction results for LL(k) and LR(k) grammars in [22] (obtained by observations on deterministic pushdown transducers and syntax-directed translation schemes) it becomes clear that every LL(k) grammar is also an LR(k) grammar. A similar approach can be found in [12] and, in a much more formal setting, in [5].

Yet another approach results if, instead of the general device of a deterministic pushdown transducer made suitable for LR(k) analysis, we use the practical parsing algorithm which is based on the construction of LR(k) state sets (cf. [8]). Then it is possible to show that the state sets which can be constructed for an LL(k) grammar satisfy the properties of the state sets of LR(k) grammars. Examples of this kind of argument can be found in [13,14,17,21,4]. Moreover such an argument has also been used to show the inclusion of other classes of grammars in the class of LR(k) grammars (cf., e.g., [18,25]).

In the third approach which we want to mention the inclusion can be shown by observations on transition diagrams which are used as parsers for a grammar (cf. [6,23,9]). It should be men-

* The preparation of this paper was partially supported by a Natural Sciences and Engineering Research Council of Canada Grant No. A-7700 during the author's stay at McMaster University, Hamilton, Ontario.

0020-0190/82/0000-0000/$02.75

tioned that in the latter papers the inclusion problem is not explicitly considered.

Instead of establishing the inclusion result by looking at properties of the parsing methods it is possible to study properties of the grammatical trees of LL(k) and LR(k) grammars. Examples of this argument can be found in [16] (e.g., the inclusion of the strict deterministic grammars in the class of LR(0) grammars) and in [3].

Finally we come to the approach which we prefer. LL(k) grammars are usually defined by introducing conditions on the leftmost derivations of a grammar. Similarly, LR(k) grammars are defined with the help of rightmost derivations. It is then natural to prove the inclusion result by showing that if the leftmost derivations of a grammar satisfy the LL(k) conditions, then the rightmost derivations satisfy the LR(k) conditions. Such proofs can be found in [29] (with a different LR-definition), [1] and [2]. Part of Aho and Ullman's proof [1] has been further formalized by Beatty [3] by studying properties of grammatical trees. Other authors who introduce classes of grammars between the LL(k) and LR(k) grammars obtain the inclusion result by showing the inclusion of the LL(k) grammars in this newly introduced grammar class and then by showing the inclusion of the new class of grammars in the class of LR(k) grammars (cf. Soisalon-Soininen and Ukkonen [31] and Pittl [28]). In [24] related inclusion results can be found. In [30] an alternative for the proof in [1] is given. That is, a straightforward and formal proof is given which only uses the definitions of LL(k) and LR(k) grammars and some properties of derivations.

This paper gives another short and formal proof of the inclusion result. It should be mentioned that the proof method is already available in the literature. It is an adaptation of a method which has been used in [11,28,26]. However, we think it is useful to give a straightforward textbook-like proof of the inclusion result using this proof method.

1.1. Preliminaries

We assume that the reader is familiar with [15]. Our notation follows this book. Let $G = (V, \Sigma, P, S)$ be a context-free grammar, let $\alpha \in V^*$ and let k be a non-negative integer. Define

$$\text{FIRST}_k(\alpha) = \left\{ {}^{(k)}w \mid \alpha \overset{*}{\Rightarrow} w \right\}.$$

For convenience we repeat the definitions of LL(k) and LR(k) grammars.

Definition 1.1. Let $k \geq 0$ and $G = (V, \Sigma, P, S)$ be a reduced context-free grammar such that $S \overset{+}{\underset{R}{\Rightarrow}} S$ is impossible in G. G is LR(k) if, for each w, w', x in Σ^*; α, α', β, β', γ in V^*; A, A' in N, if

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

(ii) $S \overset{*}{\underset{R}{\Rightarrow}} \alpha' A' x \underset{R}{\Rightarrow} \alpha'\beta' x = \gamma w'$,

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

then $(A \to \beta, \ell g(\alpha\beta)) = (A' \to \beta', \ell g(\alpha'\beta'))$.

Definition 1.2. Let $k \geq 0$ and let $G = (V, \Sigma, P, S)$ be a context-free grammar. G is LL(k) if, for each w, x, y in Σ^*; α, β, ω in V^*; A in N, if

(i) $S \overset{*}{\underset{L}{\Rightarrow}} wA\omega \underset{L}{\Rightarrow} w\alpha\omega \overset{*}{\Rightarrow} wx$,

(ii) $S \overset{*}{\underset{L}{\Rightarrow}} wA\omega \underset{L}{\Rightarrow} w\beta\omega \overset{*}{\Rightarrow} wy$,

(iii) ${}^{(k)}x = {}^{(k)}y$,

then $\alpha = \beta$.

Reduced LL(k) grammars are non-left-recursive. That is, for any A in N and α in V^* a derivation $A \overset{+}{\Rightarrow} A\alpha$ is impossible. If an LL(k) grammar is not reduced, then it is not necessarily non-left-recursive. For example, for any $k \geq 0$, the left-recursive grammar G with the productions $S \to A$, $S \to a$ and $A \to A$ is an LL(k) grammar. Moreover, for any $k \geq 0$ this grammar is not LR(k).

Proposition 1.3. *Let* $G = (N, \Sigma, P, S)$ *be a reduced* LL(k) *grammar. For any* $n \geq 0$, A *in* N, w *in* Σ^*, X, Y *in* V *and* α, β *in* V^*, *if*

(i) $A \overset{n}{\underset{L}{\Rightarrow}} wX\alpha$ *and* $A \overset{n}{\underset{L}{\Rightarrow}} wY\beta$,

(ii) $\text{FIRST}_k(X\alpha) \cap \text{FIRST}_k(Y\beta) \neq \emptyset$,

then $X\alpha = Y\beta$.

Proof. Straightforward induction on the length n of the derivations.

2. LL(k) grammars are LR(k) grammars

We now show the inclusion of the class of LL(k) grammars in the class of LR(k) grammars.

Theorem 2.1. *Let* $k \geq 0$. *Any reduced* LL(k) *grammar is an* LR(k) *grammar.*

Proof. Let $G = (V, \Sigma, P, S)$ be a reduced LL(k) grammar. Since reduced LL(k) grammars are non-left-recursive a derivation $S \overset{+}{\Rightarrow}_R S$ is impossible. Now consider the derivations (i) and (ii) of Definition 1.1. We may assume that there exist n, $m \geq 0$ and

$$A_i \in N, \quad \alpha_i, \gamma_i \in V^*, \qquad 1 \leq i \leq n,$$
$$A'_j \in N, \quad \alpha'_j, \gamma'_j \in V^*, \qquad 1 \leq j \leq m,$$

with $A_n = A$, $A'_m = A'$, $\alpha = \alpha_1 \alpha_2 \cdots \alpha_n$, $\alpha' = \alpha'_1 \cdots \alpha'_m$, $A_0 = A'_0 = S$, and productions

$$A_{i-1} \to \alpha_i A_i \gamma_i, \quad 1 \leq i \leq n,$$
$$A'_{j-1} \to \alpha'_j A'_j \gamma'_j, \quad 1 \leq j \leq m,$$

such that $\gamma_n \cdots \gamma_1 \overset{*}{\Rightarrow} w$ and $\gamma'_m \cdots \gamma'_1 \overset{*}{\Rightarrow} x$.

Claim 2.2. $n = m$, $\beta = \beta'$, $\alpha_i = \alpha'_i$, $A_i = A'_i$ *and* $\gamma_i = \gamma'_i$, $1 \leq i \leq n$.

Proof of Claim 2.2. First we prove that $\alpha_i = \alpha'_i$ for all i such that $1 \leq i \leq \min\{n, m\}$. Assume for the sake of contradiction that this is not the case. Let j be the smallest integer such that $\alpha_j \neq \alpha'_j$. Thus one of the words α_j, α'_j must be a prefix of the other. First assume $\alpha'_j = \alpha_j X \alpha$ for some $X \in V$ and $\alpha \in V^*$. We can write

$$\text{(a)} \quad S \overset{j}{\Rightarrow} \alpha_1 \cdots \alpha_j A_j \gamma_j \cdots \gamma_1 \overset{+}{\Rightarrow} \alpha_1 \cdots \alpha_j \alpha_{j+1} \cdots \alpha_n \beta \gamma_n \cdots \gamma_1,$$

$$\text{(b)} \quad S \overset{j}{\Rightarrow} \alpha_1 \cdots \alpha_j X \alpha A'_j \gamma'_j \cdots \gamma'_1 \overset{+}{\Rightarrow} \alpha_1 \cdots \alpha_j X \alpha \alpha'_{j+1} \cdots \alpha'_m \beta' \gamma'_m \cdots \gamma'_1.$$

By rewriting $\alpha_1 \cdots \alpha_j$ these derivations can be brought into a form which makes it possible to use Proposition 1.3. Notice that $\gamma w' = \alpha\beta w' = \alpha'\beta' x$. Hence,

$$\alpha_1 \cdots \alpha_j \alpha_{j+1} \cdots \alpha_n \beta w' = \alpha_1 \cdots \alpha_j X \alpha \alpha'_{j+1} \cdots \alpha'_m \beta' x$$

and

$$\alpha_{j+1} \cdots \alpha_n \beta w' = X \alpha \alpha'_{j+1} \cdots \alpha'_m \beta' x. \tag{1}$$

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

$$\text{FIRST}_k(\alpha_{j+1} \cdots \alpha_n \beta w) = \text{FIRST}_k(X \alpha \alpha'_{j+1} \cdots \alpha'_m \beta' x),$$

and since $w \in L(\gamma_n \cdots \gamma_1)$, $x \in L(\gamma'_m \cdots \gamma'_1)$

$$\text{FIRST}_k(\alpha_{j+1} \cdots \alpha_n \beta \gamma_n \cdots \gamma_1) \cap \cap \text{FIRST}_k(X \alpha \alpha'_{j+1} \cdots \alpha'_m \beta' \gamma'_m \cdots \gamma'_1) \neq \emptyset.$$

From Proposition 1.3 and derivations (a) and (b) we now may conclude that $A_j = X$. Hence, X is a nonterminal symbol and from (1) we conclude that ${}^{(1)}\alpha_{j+1} \cdots \alpha_n \beta = X = A_j$. It follows from derivation (a) that A_j is left-recursive. However, this is not possible for a reduced LL(k) grammar. Similarly the case $\alpha_j = \alpha'_j X \alpha$ leads to a contradiction. Therefore $\alpha_i = \alpha'_i$ for $1 \leq i \leq \min\{n, m\}$.

Next we show that $n = m$. First assume $n < m$. Thus $\alpha_i = \alpha'_i$ for $1 \leq i \leq n$. Instead of (a) and (b) we now have derivations

$$\text{(a')} \quad S \overset{n}{\Rightarrow} \alpha_1 \cdots \alpha_n A \gamma_n \cdots \gamma_1 \overset{+}{\Rightarrow} \alpha_1 \cdots \alpha_n \beta \gamma_n \cdots \gamma_1,$$

$$\text{(b')} \quad S \overset{n}{\Rightarrow} \alpha_1 \cdots \alpha_n A'_n \gamma'_n \cdots \gamma'_1 \overset{+}{\Rightarrow} \alpha_1 \cdots \alpha_n \alpha'_{n+1} \cdots \alpha'_m \beta' \gamma'_m \cdots \gamma'_1.$$

By rewriting $\alpha_1 \cdots \alpha_n$ these derivations can be brought into a form which makes it possible to use Proposition 1.3. We know that $\gamma w' = \alpha\beta w' = \alpha'\beta' x$. Therefore,

$$\alpha_1 \cdots \alpha_n \beta w' = \alpha_1 \cdots \alpha_n \alpha'_{n+1} \cdots \alpha'_m \beta' x$$

and

$$\beta w' = \alpha'_{n+1} \cdots \alpha'_m \beta' x. \quad (2)$$

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

$$\text{FIRST}_k(\beta w) = \text{FIRST}_k(\alpha'_{n+1} \cdots \alpha'_m \beta' x)$$

and

$$\text{FIRST}_k(\beta\gamma_n \cdots \gamma_1) \cap \cap \text{FIRST}_k(\alpha'_{n+1} \cdots \alpha'_m \beta' \gamma'_m \cdots \gamma'_1) \neq \emptyset.$$

From Proposition 1.3 and derivations (a') and (b') we may conclude that $A = A'_n$ and since G is LL(k) it follows that $\beta = \alpha'_{n+1} A'_{n+1} \gamma'_{n+1}$. Thus, instead of (2) we may write

$$\alpha'_{n+1} A'_{n+1} \gamma'_{n+1} w' = \alpha'_{n+1} \cdots \alpha'_m \beta' x$$

and we may conclude that $A'_{n+1} = {}^{(1)}\alpha'_{n+2} \cdots \alpha'_m \beta' x$. That is, $A'_{n+1} = {}^{(1)}\alpha'_{n+2} \cdots \alpha'_m \beta'$. Therefore A'_{n+1} is left-recursive. However, this is not possible for a reduced LL(k) grammar. Similarly the case $n > m$ leads to a contradiction and we conclude that $n = m$ and $\alpha_i = \alpha'_i$, $1 \leq i \leq n$. From the LL-definition and from Proposition 1.3 we now may conclude that $A_i = A'_i$ and $\gamma_i = \gamma'_i$ for $1 \leq i \leq n$.

It remains to show that $\beta = \beta'$. Notice that the derivations (a') and (b') reduce to the derivations

$$S \overset{n}{\Rightarrow} \alpha A \gamma \Rightarrow \alpha\beta\gamma$$

and

$$S \overset{n}{\Rightarrow} \alpha A \gamma \Rightarrow \alpha\beta'\gamma$$

with $\text{FIRST}_k(\beta\gamma) \cap \text{FIRST}_k(\beta'\gamma) \neq \emptyset$. Since G is LL(k) it follows that $\beta = \beta'$.

Proof of Theorem 2.1 (*continued*). It follows from Claim 2.2 that

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

That is, G is an LR(k) grammar.

Acknowledgement

I am grateful to the referee for some helpful suggestions and comments on an earlier version of this paper.

References

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

[2] R.C. Backhouse, Syntax of Programming Languages: Theory and Practice (Prentice Hall International, 1979).

[3] J.C. Beatty, Two iteration theorems for the LL(k) languages, Theoret. Comput. Sci. 12 (1980) 193–228.

[4] J.C. Beatty, On the relationship between the LL(1) and LR(1) grammars, Rept. CS-79-36, University of Waterloo, 1979.

[5] B.M. Brosgol, Deterministic translation grammars, Ph.D. Thesis, Rept. TR-3-74, Harvard University, Cambridge, 1974.

[6] M.E. Conway, Design of a separable transition diagram compiler, Comm. ACM 6 (1963) 396–408.

[7] A.J. Demers, Generalized left corner parsing, Conf. Record of the 4th ACM Symp. on Principles of Programming Languages (1977) pp. 170–182.

[8] F.L. DeRemer, Simple LR(k) grammars, Comm. ACM 14 (1971) 453–460.

[9] D. Friede, Transition diagrams and strict deterministic grammars, in: K. Weihrauch, ed., 4th GI Conf. on Theoretical Computer Science, Lect. Notes in Comput. Sci. 67 (Springer, Berlin, 1979) pp. 113–123.

[10] M.M. Geller and M.A. Harrison, On LR(k) grammars and languages, Theoret. Comput. Sci. 4 (1977) 245–276.

[11] M.M. Geller and M.A. Harrison, Characteristic parsing: A framework for producing compact deterministic parsers, J. Comput. System Sci. 14 (1977) 265–343.

[12] M. Griffiths and C. Pair, Toute grammaire LL(k) est LR(k), RAIRO Theoret. Comput. Sci. 18 (1974) 55–62.

[13] M. Hammer, A new grammatical transformation into deterministic top-down form, Rept. MAC TR-119, Massachusetts Institute of Technology, 1974.

[14] M. Hammer, A new grammatical transformation into LL(k) form, Conf. Record of the 6th Ann. Symp. on Theory of Computing (1974) pp. 266–275.

[15] M.A. Harrison, Introduction to Formal Language Theory (Addison-Wesley, Reading, MA, 1978).

[16] M.A. Harrison and I.M. Havel, On the parsing of deterministic languages, J. Assoc. Comput. Mach. 21 (1974) 525–548.

[17] S. Heilbrunner, A parsing automata approach to LR theory, Theoret. Comput. Sci. 15 (1981) 117–157.

[18] H.B. Hunt III and T.G. Szymanski, Lower bounds and reductions between grammar problems, J. Assoc. Comput. Mach. 25 (1978) 32–51.

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

[20] D.E. Knuth, Top-down syntax analysis, Acta Inform. 1 (1971) 79–110.

[21] J. Kral and J. Demner, A note on the number of states of DeRemer's recognizer, Inform. Process Lett. 2 (1973) 22–23.

[22] P.M. Lewis II and R.E. Stearns, Syntax-directed transduc-

tion, J. Assoc. Comput. Mach. 15 (1968) 465–488.

[23] D.B. Lomet, A formalization of transition diagram systems, J. Assoc. Comput. Mach. 20 (1973) 235–257.

[24] A.V. Moura, Syntactic equivalence of grammar classes, Ph.D. Thesis, University of California at Berkeley, 1980.

[25] A. Nijholt, Simple chain grammars and languages, Theoret. Comput. Sci. 9 (1979) 287–309.

[26] A. Nijholt, A framework for classes of grammars between the LL(k) and LR(k) grammars, Rept. CSTR 80-CS-25, McMaster University, Hamilton, 1980.

[27] A. Nijholt, Parsing strategies: A concise survey, in: M. Chytil and J. Gruska, eds., Mathematical Foundations of Computer Science, Lect. Notes in Comput. Sci. 118 (Springer, Berlin, 1981) pp. 103–120.

[28] J. Pittl, On LLP(k) grammars and languages, Theoret. Comput. Sci. 16 (1981) 149–175.

[29] D.J. Rosenkrantz and R.E. Stearns, Properties of deterministic top-down grammars, Inform. and Control 17 (1970) 226–256.

[30] E. Soisalon-Soininen, On comparing LL(k) and LR(k) grammars, Math. Systems Theory 13 (1980) 323–348.

[31] E. Soisalon-Soininen and E. Ukkonen, A method for transforming grammars into LL(k) form, Acta Inform. 12 (1979) 339–369.