Theoretical Computer Science 9 (1979) 287–309

SIMPLE CHAIN GRAMMARS AND LANGUAGES*

Anton NIJHOLT

Department of Mathematics, Vrije Universiteit, Amsterdam, The Netherlands

Communicated by M. A. Harrison
Received March 1978
Revised September 1978

Abstract. A subclass of the LR(0)-grammars, the class of simple chain grammars is introduced. Although there exist simple chain grammars which are not LL(k) for any $k>0$, this new class of grammars is very closely related to the LL(1) and simple LL(1) grammars. In fact it can be shown that every simple chain grammar has an equivalent simple LL(1) grammar.

Cover properties for simple chain grammars are investigated and a deterministic pushdown transducer which acts as a right parser for simple chain grammars is presented.

1. Introduction

In this paper we consider a subclass of the LR(0) *grammars* which has some interesting properties. This class of grammars, called the *simple chain grammars*, has a very simple and natural bottom-up parsing method. Our definition of a simple chain grammar was originally motivated by the parsing method for *production prefix grammars* as introduced by Geller, Graham and Harrison [4]. However, they start by constructing a parsing graph for a context-free grammar and give conditions which ensure that the parsing algorithm works correctly. In our approach we start with a grammatical definition and as can be shown a slightly adapted version of their parsing method can be used. There is also a very strong and clear correspondence with the LR(0) *parsing method* [3].

This paper is mainly concerned with the properties of simple chain grammars and languages. For the time being we consider only simple chain grammars for which no *look-ahead* is allowed. An extension with lookahead seems to be straightforward and is not considered here. The class of simple chain grammars is such that it properly contains the class of *simple* LL(1) *grammars*. However, each simple chain grammar can be transformed to an equivalent simple LL(1) grammar. Thus the simple chain grammars generate exactly the simple LL(1) (or *simple deterministic* [13]) languages. Besides the research reported in [4], work which is related to ours has been done by Lomet [16] and Conway [2].

* Part of the research reported in this paper was first presented at the Fourth Colloquium on Automata, Languages and Programming in Turku (Finland), 1977 (cf. [18]).

Material which is closely related to the parsing method which can be used for simple chain grammars appears in the work of Král [14] and Král and Demner [15]. They consider some top-down properties of DeRemers LR(0) parsing method. A comparison with this work will not be given here.

The organization of this paper is as follows. The remainder of this section is devoted to some preliminaries. In Section 2 we introduce the simple chain grammars. We develop some of their properties and give examples of grammars which are simple chain grammars but which are not, for any k, LL(k), LC(k) (i.e. grammars which can be parsed in a *left corner manner* with a deterministic pushdown transducer scanning the input from left to right [1]) or *left parsable* (i.e. grammars which can be parsed with a deterministic pushdown transducer scanning the input from left to right and resulting in a left parse [17]). Section 3 is devoted to relationships with some other classes of grammars and in Section 4 we give our results on simple chain languages. We give transformations to simple chain grammars in *Greibach normal form* and to simple LL (1) grammars. From these results some decidability questions can be answered. Section 4 is concluded with results concerning grammar covers for simple chain grammars.

Preliminaries

We assume the reader is familiar with the basic concepts of formal languages and automata theory [1]. Some of them are reviewed below for notational reasons.

A *context-free grammar* (cfg) is denoted by $G=(N, T, P, S)$, where N is the set of *nonterminals*, T is the set of *terminals*, $V=N\cup T$, $P\subseteq N\times V^+$ is the set of context-free *productions* and S is the *start symbol*. Elements of N will be denoted by $A, \ldots, S$; elements of T by $a, b, c, \ldots$; elements of V by $U, \ldots, Z$; elements of V^* by $\alpha, \beta, \gamma, \delta, \ldots$; and elements of T^* by u, v, w, x, y, z. Instead of writing (A, α) in P we will write $A\to\alpha$ in P. The *length* of $\alpha\in V^*$ is denoted by $|\alpha|$; the symbol ε is reserved for the empty string. If $\alpha\in V^*$, then ${}^{(n)}\alpha$ denotes α if $|\alpha|<n$ and otherwise the prefix of α of length n. The set of productions P is said to be *prefix-free* if $A\to\alpha$ and $A\to\alpha\beta$ in P implies $\beta=\varepsilon$.

The relation $\Rightarrow\subseteq V^*\times V^*$ is defined as follows: for any $\alpha, \beta\in V^*$ $\alpha\Rightarrow\beta$ iff $\alpha=\alpha_1A\alpha_2$, $\beta=\alpha_1\beta_1\alpha_2$ and $A\to\beta_1$ is in P for some $A\in N$ and $\alpha_1, \alpha_2, \beta_1\in V^*$. If $\alpha_1\in T^*$ or $\alpha_2\in T^*$ we write $\alpha\Rightarrow_l\beta$ and $\alpha\Rightarrow_r\beta$ respectively. *Transitive* and *reflexive-transitive* closures of these relations are defined in the usual way. If $\alpha_0\Rightarrow\alpha_1\Rightarrow\cdots\Rightarrow\alpha_n$, then this sequence is said to be a *derivation* of α_n from α_0. If in this sequence $\Rightarrow_l$ is used, then it is a *leftmost* derivation; if $\Rightarrow_r$ is used, then it is a *rightmost* derivation. If $\alpha\in V^*$, then $L(\alpha)=\{w\in T^*\mid \alpha\Rightarrow^* w\}$. The *language* of G, denoted by $L(G)$ is the set $L(S)$. FIRST $(\alpha)=\{a\in T\mid \alpha\Rightarrow^* a\phi$ for some $\phi\in V^*\}$. Notice that $P\subseteq N\times V^+$. Hence there are no productions of the form $A\to\varepsilon$, so that the cfg's in this paper are assumed to be ε-free. A cfg is *cycle-free* if there is no derivation $A\Rightarrow^+A$ for any $A\in N$. A nonterminal A is said to be *left-recursive* if

$A \Rightarrow^+ A\alpha$ for some $\alpha \in V^*$. A cfg is said to be left-recursive if it has at least one left-recursive nonterminal. We assume that all the context-free grammars in this paper are *reduced.*

Definition 1.1. A cfg $G = (N, T, P, S)$ is in *pseudo-Greibach normal form* (pseudo-GNF) if every production in P is of the form $A \to a\alpha$, where $a \in T$ and $\alpha \in V^*$. If $\alpha \in N^*$, then G is said to be in *Greibach normal form* (GNF).

Definition 1.2. A cfg $G = (N, T, P, S)$ is an LL(1) *grammar* if for every pair $A \to \alpha$ and $A \to \beta$ in P, if $\alpha \neq \beta$, then FIRST $(\alpha) \cap$ FIRST $(\beta) = \emptyset$. G is said to be a *simple* LL(1) *grammar*

(i) if every production is of the form $A \to a\phi (a \in T, \phi \in V^*)$, and

(ii) if $A \to a\phi$ and $A \to b\psi$, then either $a \neq b$ or $a\phi = b\psi$.

Our definition of an LL(1) grammar is somewhat simpler than the usual one (see for example [1]), this being a result of the fact that our grammars are ε-free. It is well-known that LL(1) grammars are not left-recursive and that each simple LL(1) grammar is LL(1).

2. Simple chain grammars

In this section we introduce the class of simple chain grammars and discuss some of their properties.

Definition 2.1. A cfg $G = (N, T, P, S)$ is said to be a *simple chain grammar* if P is prefix-free and for any $A \in N, \alpha, \phi, \psi \in V^*$ and $X, Y \in V$ with $X \neq Y$, if $A \to \alpha X\phi$ and $A \to \alpha Y\psi$, then FIRST$(X) \cap$ FIRST$(Y) = \emptyset$.

Our first task is to prove that each (ε-free) LL(1) grammar is a simple chain grammar. After that we will be concerned with a definition of simple chain grammars which is equivalent with Def. 2.1 but in which some of the useful properties of simple chain grammars are explicitly mentioned.

Lemma 2.1. *Every* LL(1) *grammar is a simple chain grammar.*

Proof. Let $G = (N, T, P, S)$ be an (ε-free) LL(1) grammar and assume that G is not a simple chain grammar. If P is not prefix-free, then there is $A \in N$ and $\alpha, \beta \in V^*$ such that $A \to \alpha$, $A \to \alpha\beta$ and $\beta \neq \varepsilon$. This obviously contradicts the LL(1)-definition. Now suppose there exist $A \in N, \alpha, \phi, \psi \in V^*, X, Y \in V$ and rules $A \to \alpha X\phi$, $A \to \alpha Y\psi$ with $X \neq Y$ and FIRST$(X) \cap$ FIRST$(Y) \neq \emptyset$. Since $\alpha X\phi \neq \alpha Y\psi$ and FIRST$(\alpha X\phi) \cap$ FIRST$(\alpha Y\psi) \neq \emptyset$ there is again a contradiction with the LL(1)-definition.

Definition 2.2. Let $G = (N, T, P, S)$ be a cfg and let $X_0 \in V$. Then $\mathrm{CH}(X_0)$, the set of chains of X_0, is defined by

$$\mathrm{CH}(X_0) = \{X_0X_1 \cdots X_n \in N^*T \mid X_0 \underset{l}{\Rightarrow} X_1\psi_1 \underset{l}{\Rightarrow} \cdots \underset{l}{\Rightarrow} X_n\psi_n, \psi_i \in V^*, 1 \leq i \leq n\}.$$

From Def. 2.2. it follows that each chain of $X_0 \in V$ ends with a terminal. Moreover, if $X_0 \in T$, then $\mathrm{CH}(X_0) = \{X_0\}$. For any $\pi \in \mathrm{CH}(X_0)$, $l(\pi)$ denotes the last element of π. Thus if $\pi = X_0X_1 \cdots X_n$, then $l(\pi) = X_n$ and $l(\pi) \in T$. In the following lemmas and definitions $G = (N, T, P, S)$ and $V = N \cup T$.

Definition 2.3. Let $X \in V$. X is said to be *chain-independent* iff for each pair π_1, π_2 in $\mathrm{CH}(X)$, if $\pi_1 \neq \pi_2$, then $l(\pi_1) \neq l(\pi_2)$. If each element of V is chain-independent, then V is said to be chain-independent.

Clearly each terminal is chain-independent. Some other properties are listed in the following lemma.

Lemma 2.2. *Let* $A \in N$:

(i) *If G is in* GNF, then each chain in $\mathrm{CH}(A)$ *is of length* 2.
(ii) *A is a non-left recursive nonterminal if* $\mathrm{CH}(A)$ is finite.
(iii) $\mathrm{CH}(A)$ *is a regular set.*
(iv) *If V is chain-independent, then for each $X \in V$,* $\mathrm{CH}(X)$ *is finite.*
(v) *If V is chain-independent, then G is a non-left recursive* cfg.

Proof. Results (i) and (ii) are obvious. Result (iii) can easily be obtained by constructing a regular grammar for $\mathrm{CH}(A)$. Result (iv) is also obvious and result (v) follows from (ii) and (iv).

Definition 2.4. Let $X, Y \in V$, $X \neq Y$. X and Y are said to be *mutually chain-independent*, and we write $X \not\equiv Y$, if for each pair $\pi_1 \in \mathrm{CH}(X)$ and $\pi_2 \in \mathrm{CH}(Y)$ $l(\pi_1) \neq l(\pi_2)$.

Lemma 2.3. *Let* $X, Y \in V$, $X \neq Y$. *Then* $X \not\equiv Y$ *iff* $\mathrm{FIRST}(X) \cap \mathrm{FIRST}(Y) = \emptyset$.

Proof. Trivial.

Notice that $a \not\equiv b$ for each pair a, b in T with $a \neq b$. The following corollary is obvious.

Corollary 2.1. Cfg $G = (N, T, P, S)$ *is a simple chain grammar iff P is prefix-free and if there are α, ϕ, $\psi \in V^*$, $A \in N$ and $X, Y \in V$ with $X \neq Y$, such that $A \to \alpha X\phi$ and $A \to \alpha Y\psi$, then $X \not\equiv Y$.*

Lemma 2.4. *If* $\mathrm{FIRST}(X) \cap \mathrm{FIRST}(Y) = \emptyset$ *for each pair* $A \to \alpha X \phi$, $A \to \alpha Y \psi$, *where* $\alpha, \phi, \psi \in V^*$, $X, Y \in V$ *and* $X \neq Y$, *then* V *is chain-independent.*

Proof. Assume that V is not chain-independent. Hence there exist $A \in N$ and π_1, $\pi_2 \in \mathrm{CH}(A)$ such that $\pi_1 \neq \pi_2$ and $l(\pi_1) = l(\pi_2)$. Let $\pi_1 = X_0 X_1 \cdots X_n$ and $\pi_2 = Y_0 Y_1 \cdots Y_m$, where $X_0 = Y_0 = A$ and $X_n = Y_m$. Then there exists a maximal $i \geq 0$ such that $X_0 X_1 \cdots X_i = Y_0 Y_1 \cdots Y_i$, there exists a derivation $A \Rightarrow_l^* X_i \psi_i$ for some $\psi_i \in V^*$ and there exist productions $X_i \to X_{i+1} \psi_{i+1}$, $X_i \to Y_{i+1} \psi'_{i+1}$, for some ψ_{i+1}, $\psi'_{i+1} \in V^*$ such that $X_{i+1} \neq Y_{i+1}$.

By hypothesis, then $\mathrm{FIRST}(X_{i+1}) \cap \mathrm{FIRST}(Y_{i+1}) = \emptyset$. But this contradicts the assumption that $l(\pi_1) = l(\pi_2)$.

Notice that in this proof we explicitly need the possibility that $\alpha = \varepsilon$. From Corollary 2.1 and Lemma 2.4, the following corollary is obtained in an obvious way.

Corollary 2.2. *A* cfg $G = (N, T, P, S)$ *is a simple chain grammar iff the following three conditions are satisfied:*

(i) V *is chain-independent.*

(ii) *If there exist* $\alpha \in V^+$, $\phi, \psi \in V^*$, $A \in N$ *and* $X, Y \in V$ *with* $X \neq Y$ *such that* $A \to \alpha X \phi$ *and* $A \to \alpha Y \psi$, *then* $X \not\equiv Y$.

(iii) P *is prefix-free.*

Hence the three conditions in this corollary can be used as a definition of simple chain grammars and will be useful in proofs of properties of simple chain grammars.

To illustrate the definition of a simple chain grammar we consider a few examples.

Example 2.1. The cfg G with productions $S \to AF$, $A \to Ba$, $B \to Cd$, $C \to dF$, $F \to Ga$, $G \to Cb$, $C \to dB'$, $F \to a$, $B' \to b$. For this cfg we have for instance $\mathrm{CH}(C) = \{Cd\}$, $\mathrm{CH}\{a\} = \{a\}$, and $\mathrm{CH}(F) = \{Fa, FGCd\}$. One can easily verify that G satisfies the three conditions of Corollary 2.2 and therefore G is a simple chain grammar.

In the following two examples we list simple chain grammars which are not $\mathrm{LL}(k)$, $\mathrm{LC}(k)$ (for any $k > 0$) and left-parsable respectively. For definitions of these classes of grammars the reader should consult [1, 17]. The proofs are straightforward from these definitions.

Example 2.2. The cfg G with productions $S \to aEc$, $S \to aEd$, $E \to aE$ and $E \to ab$ is a simple chain grammar since V is chain-independent, P is prefix-free and $c \not\equiv d$ and $E \not\equiv b$. However, there is no k such that G is $\mathrm{LL}(k)$ or G is $\mathrm{LC}(k)$.

Example 2.3. The cfg G with productions $S \to aEc$, $S \to aEd$, $E \to aEb$, $E \to ab$ is a simple chain grammar. However G is not left-parasable, that is, there does not exist a deterministic pushdown transducer which can act as a left parser for G.

Remark. In Section 4 we will give a transformation from simple chain grammars to simple LL(1) grammars. From the last example we can obtain a nice application of the theory of *parsable grammars* [17] and of *covers* [8]. The cfg G of Example 2.3 is not left-parsable, from which one can prove that there is no left parsable grammar which left covers G. Therefore, since simple LL(1) grammars are left-parsable, there is no simple LL(1) grammar which left covers G. Thus we can conclude that there is no algorithm which transforms a simple chain grammar G to a simple LL(1) grammar G' such that, in general, G' left covers G. For a more detailed discussion see [19] and Section 4. Also, the cfg of Example 2.2 is a simple chain grammar and it is not LL(1). Therefore the LL(1) grammars are properly included in the class of simple chain grammars.

Definition 2.5. Let $G=(N, T, P, S)$ be a cfg and let $\alpha \in V^*$. α is said to be *prefix-free* if $\alpha \Rightarrow^* w_1$ and $\alpha \Rightarrow^* w_1w_2$ implies $w_2=\varepsilon$. A cfg is said to be prefix-free if all nonterminals are prefix-free. A language L is prefix-free if $w_1 \in L$ and $w_1w_2 \in L$ implies $w_2=\varepsilon$.

Theorem 2.1. *Every simple chain grammar is prefix-free.*

Proof. We have to prove that every nonterminal of a simple chain grammar is prefix-free. Let $G=(N, T, P, S)$ be a simple chain grammar. By induction on the length of the derivations we prove that any $\mu \in V^+$ is prefix-free.

Basis. Consider two derivations of length 1 to obtain w_1 and w_1w_2 in T^*; the case in which one derivation is of length 1 and the other is of length 0 cannot occur. If $\mu \Rightarrow_r w_1$ and $\mu \Rightarrow_r w_1w_2$, then there exists a variable $C \in N$ and strings w', w'', z_1, $z_2 \in T^*$ such that

$$\mu = w'Cw'' \underset{r}{\Rightarrow} w'z_1w'' = w_1 \quad \text{and} \quad \mu = w'Cw'' \underset{r}{\Rightarrow} w'z_2w'' = w_1w_2.$$

If $w_2 \neq \varepsilon$, then z_1 is a prefix of z_2 and P is not prefix-free, whence $w_2=\varepsilon$.

Induction. Assume for all $\mu \in V^+$ and derivations $\mu \Rightarrow_r^* w_1$ and $\mu \Rightarrow_r^* w_1w_2$ with length less than n, we have $w_2=\varepsilon$. Now consider derivations $\mu \Rightarrow_r^* w_1$ and $\mu \Rightarrow_r^* w_1w_2$ with lengths less than or equal to n. Then there exist $C \in N$, ρ, ρ_1, ϕ_1, $\phi_2 \in V^*$, v_1, v_2, $w' \in T^*$ and X, $Y \in V$ such that $C \to \rho_1 X \phi_1$ and $C \to \rho_1 Y \phi_2$ are in P, with $X \neq Y$ and

$$\mu \overset{*}{\underset{r}{\Rightarrow}} \rho Cw' \underset{r}{\Rightarrow} \rho\rho_1 X\phi_1 w' \overset{*}{\underset{r}{\Rightarrow}} \rho\rho_1 Xv_1w' \overset{*}{\underset{r}{\Rightarrow}} w_1$$

and

$$\mu \overset{*}{\underset{r}{\Rightarrow}} \rho Cw' \underset{r}{\Rightarrow} \rho\rho_1 Y\phi_2 w' \overset{*}{\underset{r}{\Rightarrow}} \rho\rho_1 Yv_2w' \overset{*}{\underset{r}{\Rightarrow}} w_1w_2,$$

where $\rho Cw'$ is the last right sentential form which these two derivations have in common. Since $\mathrm{FIRST}(X) \cap \mathrm{FIRST}(Y) = \emptyset$ we must have $\rho\rho_1 \neq \varepsilon$. Moreover, to

obtain both w_1 and w_1w_2 there exists $w \neq \varepsilon$ and $\bar{w} \neq \varepsilon$ such that $\rho\rho_1 \Rightarrow_r^* w\bar{w}$ and $\rho\rho_1 \Rightarrow_r^* w$, where both w and $w\bar{w}$ are prefixes of w_1, and both derivations are of length less than n. Since this contradicts the induction hypothesis we must conclude $w_2 = \varepsilon$. This concludes the proof that every $\mu \in V^+$ and hence every $A \in N$ is prefix-free.

Theorem 2.2. *Every simple chain grammar is unambiguous.*

Proof. We have to prove that each $w \in L(G)$, where $G = (N, T, P, S)$ is a simple chain grammar, has exactly one (rightmost) derivation from S. Suppose $S \Rightarrow_r^* w$ by at least two rightmost derivations. Then there exist $A \in N$, $\rho, \phi_1, \phi_2 \in V^*$ and $X, Y \in V$, where $X \neq Y$ such that there are derivations

$$A \Rightarrow \rho X\phi_1 \overset{*}{\underset{r}{\Rightarrow}} w' \quad \text{and} \quad A \Rightarrow \rho Y\phi_2 \overset{*}{\underset{r}{\Rightarrow}} w',$$

where $w' \neq \varepsilon$ is a substring of w.

Since $X \not\equiv Y$ we must conclude that ρ is not prefix-free which is in contradiction with Theorem 2.1. Therefore there are no two such derivations.

A characteristic feature of simple chain grammars is mentioned in the following theorem. The notation $\Rightarrow_l^n$ is used to indicate that the derivation is of length n.

Theorem 2.3. *Let $G = (N, T, P, S)$ be a simple chain grammar. Suppose there exist $X, Y \in V$ and $\alpha, \phi, \psi \in V^*$ such that $S \Rightarrow_l^n \alpha X\phi$ and $S \Rightarrow_l^n \alpha Y\psi$. If $X \neq Y$, then $X \not\equiv Y$.*

Proof. The proof is by induction on the length of the derivations. To facilitate the induction proof we take an arbitrary string $\mu \in V^+$ instead of the start symbol S.

Basis. Let $\mu \Rightarrow_l^1 \alpha X\phi$ and $\mu \Rightarrow_l^1 \alpha Y\phi$. Suppose $\mu = \gamma C\rho$, where $C \in N$ and $\rho, \gamma \in V^*$. Then there are productions $C \to \gamma_1 X\rho_1$ and $C \to \gamma_1 Y\rho_2$ in P such that $\gamma\gamma_1 = \alpha$, $\rho_1\rho = \phi$ and $\rho_2\rho = \psi$. Since $X \neq Y$ we obtain $X \not\equiv Y$.

Induction. Let $\mu \Rightarrow_l^n \alpha X\phi$ and $\mu \Rightarrow_l^n \alpha Y\psi$ where $X \neq Y$ and assume the property holds for all $\mu \in V^*$ and leftmost derivations with length less than n. Then there exist $\alpha_1, \delta, \phi_1, \psi_1, \rho \in V^*$, $X_1, Y_1 \in V$ and $C \in N$ such that $C \to \delta X_1\phi_1$ and $C \to \delta Y_1\psi_1$, where $X_1 \neq Y_1$, and

$$\mu \overset{k}{\underset{l}{\Rightarrow}} \alpha_1 C\rho \underset{l}{\Rightarrow} \alpha_1\delta X_1\phi_1\rho \overset{m}{\underset{l}{\Rightarrow}} \alpha X\phi$$

and

$$\mu \overset{k}{\underset{l}{\Rightarrow}} \alpha_1 C\rho \underset{l}{\Rightarrow} \alpha_1\delta Y_1\psi_1\rho \overset{m}{\underset{l}{\Rightarrow}} \alpha Y\psi,$$

with $n = k+m+1$. If $m = 0$, then $X_1 = X$, $Y_1 = Y$ and since $X_1 \not\equiv Y_1$ we have also $X \not\equiv Y$. Otherwise, since $\alpha_1\delta$ is prefix-free, there are two possibilities:

(i)

$$\alpha_1\delta \underset{l}{\overset{m}{\Rightarrow}} \alpha X\phi_1', \quad \text{where } \phi_1' X_1 \phi_1 \rho = \phi$$

and

$$\alpha_1\delta \underset{l}{\overset{m}{\Rightarrow}} \alpha Y\psi_1', \quad \text{where } \psi_1' Y_1 \psi_1 \rho = \psi.$$

Since $m < n$ we have $X \not\equiv Y$.

(ii) $\alpha_1\delta \Rightarrow^* \alpha'$ is a prefix of α, that is $\alpha = \alpha'\alpha''$ and

$$X_1 \underset{l}{\overset{*}{\Rightarrow}} \alpha'' X\phi_1', \quad \text{where } \phi_1' \text{ is a prefix of } \phi$$

and

$$Y_1 \underset{l}{\overset{*}{\Rightarrow}} \alpha'' Y\psi_1', \quad \text{where } \psi_1' \text{ is a prefix of } \psi.$$

Since $X_1 \not\equiv Y_1$ we have $\alpha'' = \varepsilon$ and $X \not\equiv Y$.

It follows that $S \Rightarrow_l^n wX\phi$ and $S \Rightarrow_l^n wY\psi$ with $X \neq Y$ implies $X \not\equiv Y$.

In the remainder of this section we present some results on the rightmost derivations of simple chain grammars.

First we have the following result. In this lemma π denotes the concatenation of the productions in the rightmost derivation.

Lemma 2.5. *Let $G = (N, T, P, S)$ be a simple chain grammar. Let $A \in N$, $X \in V$, $\phi \in V^*$ and $v_1, v_2 \in T^*$ such that $S \Rightarrow_r^* \phi A v_1 \Rightarrow_r^\pi \phi X v_2$, where $A \neq X$. Then there is $V' \in T^*$ such that $v'v_1 = v_2$ and $A \Rightarrow_r^\pi Xv'$.*

Proof. Notice that we can not have $\phi \Rightarrow_r^* \phi Xu$ for some $u \in T^*$ since ϕ is prefix-free. Neither can we have $\phi \Rightarrow_r^* \phi'$, where ϕ' is a proper prefix of ϕ since there are no ε-productions. Therefore we must conclude that $A \Rightarrow_r^\pi Xv'$ for $v' \in T^*$ and $v'v_1 = v_2$.

Theorem 2.4. *Let $G = (N, T, P, S)$ be a simple chain grammar. Let $\alpha \in V^*$, $X, Y \in V$ and $w_1, w_2 \in T^*$ such that $S \Rightarrow_r^* \alpha X w_1$ and $S \Rightarrow_r^* \alpha Y w_2$, where $X \neq Y$. Then either $X \not\equiv Y$ or there is a string $u \in T^*$ such that $S \Rightarrow_r^* \alpha Xu \Rightarrow_r^* \alpha Y w_2$ (or the symmetric case $S \Rightarrow_r^* \alpha Yu \Rightarrow_r^* \alpha X w_1$).*

Proof. The proof is by induction on the length of the derivations. Let $\mu \in V^+$. As basis we consider derivations of length one or less.

Basis. First consider derivations

$$\mu \underset{r}{\Rightarrow} \alpha X w_1 \quad \text{and} \quad \mu \underset{r}{\Rightarrow} \alpha Y w_2,$$

where $X \neq Y$. Then there exists $\rho \in V^*$, $C \in N$, $w \in T$ and $C \to \rho_1 X v_1$, $C \to \rho_1 Y v_2$ in P, such that $\rho\rho_1 = \alpha$, $v_1 w = w_1$ and $v_2 w = w_2$. Since G is a simple chain grammar we have $X \not\equiv Y$.

Now suppose that $\mu = \alpha X w_1$ and $\mu \Rightarrow_r \alpha Y w_2$. Then $\alpha X w_1 \Rightarrow_r \alpha Y w_2$, hence we have a derivation $\mu \Rightarrow_r^* \alpha X w_1 \Rightarrow_r^* \alpha Y w_2$, which is of the desired form. The basis of the induction is now satisfied.

Induction. Now suppose we have derivations

$$\mu \underset{r}{\overset{*}{\Rightarrow}} \alpha X w_1 \quad \text{and} \quad \mu \underset{r}{\overset{*}{\Rightarrow}} \alpha Y w_2,$$

where $X \neq Y$ and the lengths of the derivations are less than or equal to n. Assume the property holds for all derivations with lengths less than n. There exist ρ, ρ_1, ϕ_1, $\phi_2 \in V^*$, $C \in N$, X_1, $Y_1 \in V$ such that $C \to \rho_1 X_1 \phi_1$ and $C \to \rho_1 Y_1 \phi_2$ are in P, with $X_1 \neq Y_1$ and there exist derivations

$$\mu \underset{r}{\overset{*}{\Rightarrow}} \rho C w \underset{r}{\Rightarrow} \rho\rho_1 X_1 \phi_1 w \underset{r}{\overset{*}{\Rightarrow}} \alpha X w_1$$

and

$$\mu \underset{r}{\overset{*}{\Rightarrow}} \rho C w \underset{r}{\Rightarrow} \rho\rho_1 Y_1 \phi_2 w \underset{r}{\overset{*}{\Rightarrow}} \alpha Y w_2.$$

Since $X_1 \not\equiv Y_1$ and $\rho\rho_1$ is prefix-free there are two possibilities:

(i) $$\rho\rho_1 \underset{r}{\overset{*}{\Rightarrow}} \alpha X v_1 \quad \text{and} \quad \rho\rho_1 \underset{r}{\overset{*}{\Rightarrow}} \alpha Y v_2,$$

where v_1 is a prefix of w_1 and v_2 is a prefix of w_2. But then, since the lengths of these derivations are less than n, we have by the induction hypothesis either $X \not\equiv Y$ or there is a string $v \in T^*$ such that $\rho\rho_1 \Rightarrow_r^* \alpha X v \Rightarrow_r^* \alpha Y v_2$, where v_2 is a prefix of w_2. If $\rho\rho_1 \Rightarrow_r^* \alpha Y v_2$, then there is a derivation

$$\mu \underset{r}{\overset{*}{\Rightarrow}} \rho\rho_1 Y_1 \phi_2 w \underset{r}{\overset{*}{\Rightarrow}} \rho\rho_1 w' \underset{r}{\overset{*}{\Rightarrow}} \alpha Y v_2 w' = \alpha Y w_2.$$

Moreover, since $\rho\rho_1 \Rightarrow_r^* \alpha X v \Rightarrow_r^* \alpha Y v_2$ we can write

$$\mu \underset{r}{\overset{*}{\Rightarrow}} \rho\rho_1 Y_1 \phi_2 w \underset{r}{\overset{*}{\Rightarrow}} \rho\rho_1 w' \underset{r}{\overset{*}{\Rightarrow}} \alpha X v w' \underset{r}{\overset{*}{\Rightarrow}} \alpha Y v_2 w' = \alpha Y w_2.$$

Therefore there is $u = vw' \in T^*$ such that $\mu \Rightarrow_r^* \alpha X u \Rightarrow_r^* \alpha Y w_2$.

(ii) $X_1 \overset{*}{\underset{r}{\Rightarrow}} \rho' X w_1'$, and $Y_1 \overset{*}{\underset{r}{\Rightarrow}} \rho' Y w_2'$,

where ρ' is a suffix of α and w_1' and w_2' are prefixes of w_1 and w_2 respectively. But then, since $X_1 \not\equiv Y_1$ we have $\rho' = \varepsilon$ and $X \not\equiv Y$.

The proof of this theorem is now complete if we take $\mu = S$.

Note. It follows from Theorem 2.4 that, if $S \Rightarrow_r^* \alpha X w_1$ and $S \Rightarrow_r^* \alpha Y w_2$, where $X \neq Y$ and we do not have $X \not\equiv Y$, then there exists $v \in T^*$ such that $S \Rightarrow_r^* \alpha X v \Rightarrow_r^{\pi} \alpha Y w_2$, where $X \Rightarrow_r^{\pi} Y w'$ and w' is such that $w'v = w_2$ (or the symmetric case).

The following corollary is immediate from Theorem 2.4. We will need the following notation. For $X, Y \in V$ we write $X \perp Y$ if there does not exist $\psi \in V^*$ such that either $X \Rightarrow^* Y\psi$ or $Y \Rightarrow^* X\psi$. Notice that if $X \neq Y$, then $X \not\equiv Y$ implies $X \perp Y$. If $X \perp Y$, then $X \neq Y$.

Corollary 2.3. *Let $G = (N, T, P, S)$ be a simple chain grammar. Suppose there exist $\alpha \in V^*$, $X, Y \in V$ and $w_1, w_2 \in T^*$ such that $S \Rightarrow_r^* \alpha X w_1$ and $S \Rightarrow_r^* \alpha Y w_2$. If $X \perp Y$, then $X \not\equiv Y$.*

Note. Notice that we do not have $S \Rightarrow_r^* \alpha X w_1$ and $S \Rightarrow_r^* \alpha Y w_2$ implies $X \not\equiv Y$. A counter-example is the cfg G with productions $S \to aXb$, $X \to Yc$ and $Y \to a$. G is a simple chain grammar and we have $S \Rightarrow_r^* aXb$ and $S \Rightarrow_r^* aYcb$ as possible derivations but we do not have $X \not\equiv Y$. Another example is the cfg with only productions $S \to aXD$, $S \to aXe$, $X \to Y$, $Y \to b$ and $D \to d$ which is also a simple chain grammar. Here we have derivations $S \Rightarrow_r^2 aXd$ and $S \Rightarrow_r^2 aYe$, but we do not have $X \not\equiv Y$.

3. Relationship of simple chain grammars with other classes of grammars

In the examples of the preceeding section we have already seen some results of this kind. In particular we saw that each ε-free LL(1) grammar is a simple chain grammar. Here we will compare the class of simple chain grammars with the classes of grammars that are *simple precedence, strict deterministic* and LR(0). For the definition of simple precedence grammar the reader is referred to [1]. The definition of strict deterministic grammar can be found in [10].

The cfg with productions $S \to Ab$, $S \to Bc$, $A \to a$ and $B \to ad$ is not a simple chain grammar. By constructing the *Wirth–Weber precedence matrix* one can easily verify that there are no precedence conflicts. Since the grammar is also *uniquely invertible* it is a simple precedence grammar. On the other hand, the cfg with only productions $S \to aA$, $S \to bB$, $A \to dc$, $B \to dC$ and $C \to c$ is a simple chain grammar and not a simple precedence grammar.

Corollary 3.1. *The classes of simple chain grammars and of simple precedence grammars are incomparable.*

The cfg with only productions $S \to cb$, $S \to Ab$ and $A \to a$ is a simple chain grammar but not a strict deterministic grammar. The cfg with only productions $S \to Ab$, $S \to Bc$, $A \to ad$ and $B \to ae$ is a strict deterministic grammar and not a simple chain grammar.

Corollary 3.2. *The classes of simple chain grammars and of strict deterministic grammars are incomparable.*

Definition 3.1 (LR(0) grammar [6]). The cfg $G = (N, T, P, S)$ is said to be LR(0) iff $S \Rightarrow_r^+ S$ is not possible in G and for each $w, w', x \in T^*$; $\gamma, \alpha, \alpha', \beta, \beta' \in V^*$ and $A, A' \in N$, if

$$S \overset{*}{\underset{r}{\Rightarrow}} \alpha A w \underset{r}{\Rightarrow} \alpha\beta w = \gamma w$$

and

$$S \overset{*}{\underset{r}{\Rightarrow}} \alpha' A' x \underset{r}{\Rightarrow} \alpha'\beta' x = \gamma w',$$

then $A \to \beta = A' \to \beta'$ and $|\alpha\beta| = |\alpha'\beta'|$.

The following proof, which shows that every simple chain grammar is an LR(0) grammar was suggested by a referee. We show that if a cfg G is a simple chain grammar then the state sets of the usual LR(0) parsing algorithm for G do not contain inconsistent items.

We recall a few definitions. However, to avoid too much repetition of terminology we assume that the reader is familiar with the construction of the LR(0) parsing method [1].

Definition 3.2. Suppose that $S \Rightarrow_r^* \alpha A w \Rightarrow_r \alpha\beta w$ in a cfg G. String γ is a *viable prefix* of G if γ is a prefix of $\alpha\beta$. We say that $[A \to \beta_1 \cdot \beta_2]$ is an LR(0) *item* for G if $A \to \beta_1\beta_2$ is a production in P. This LR(0) item is *valid* for $\alpha\beta_1$ (a viable prefix of G) if there is a derivation $S \Rightarrow_r^* \alpha A w \Rightarrow_r \alpha\beta_1\beta_2 w$.

For any viable prefix γ of G define $\mathcal{V}(\gamma)$ to be the set of LR(0) items valid for γ. Define

$$\mathcal{S} = \{s \mid s = \mathcal{V}(\gamma) \text{ for some viable prefix } \gamma \text{ of } G\},$$

the collection of LR(0) state sets for G.

In the construction of $\mathcal{S}$ each $s \in \mathcal{S}$ is obtained as the union of a *basis set* and a set which is achieved by taking the *closure* of this basis set. We denote the basis set of a set $s \in \mathcal{S}$ by basis(s).

Theorem 3.1. *Every simple chain grammar is on* LR(0) *grammar.*

Proof. Let $\mathcal{S}$ be the collection of state sets as defined above. First we have the following result:

Claim. Let $s \in \mathcal{S}$. If $[A \to \alpha_1 \cdot \alpha_2]$ and $[B \to \beta_1 \cdot \beta_2]$ are any two distinct items in basis(s), then $A = B$ and $\alpha_1 = \beta_1$.

Proof of the claim. By definition of $\mathcal{S}$ there exists $\gamma \in V^*$ such that $s = \mathcal{V}(\gamma)$. It is convenient to prove the claim by induction on $|\gamma|$.

Basis: $|\gamma| = 0$. By convention (see Algorithm 5.8 [1]) basis($\mathcal{V}(\varepsilon)$) = $\{[S \to \cdot\, \alpha] \mid S \to \alpha$ is in $P\}$ for which the claim is easily verified. Notice that for every state set s other than $\mathcal{V}(\varepsilon)$ an item $[A \to \alpha_1 \cdot \alpha_2]$ can only be in basis(s) if $\alpha_1 \neq \varepsilon$.

Induction. Consider a string γX where $\gamma \in V^*$ and $X \in V$. Assume that the claim is true for $s = \mathcal{V}(\gamma)$; we will show that it is likewise true for $s' = \mathcal{V}(\gamma X)$.

Let $[A \to \alpha_1 X \cdot \alpha_2]$ and $[B \to \beta_1 X \cdot \beta_2]$ be any two items in basis(s'). Then both $[A \to \alpha_1 \cdot X\alpha_2]$ and $[B \to \beta_1 \cdot X\beta_2]$ are in s. There are now several cases:

(i) $\alpha_1 \neq \varepsilon \neq \beta_1$. By the induction hypothesis $A = B$ and $\alpha_1 = \beta_1$, as desired since both items belong to basis(s).

(ii) $\alpha_1 \neq \varepsilon$ and $\beta_1 = \varepsilon$. In this case $[B \to \cdot\, X\beta_2]$ is obtained from some item $[C \to \gamma_1 \cdot Y\gamma_2] \in$ basis(s), so that $Y \Rightarrow_1^+ X\phi$ for some $\phi \in V^*$ and because of the induction hypothesis, $C = A$ and $\gamma_1 = \alpha_1$. Hence we have productions $A \to \alpha_1 Y\gamma_2$ and $A \to \alpha_1 X\alpha_2$. If $X = Y$, then X is left-recursive, which is not possible in a simple chain grammar. If $X \neq Y$, then, since FIRST(X) $\cap$ FIRST(Y) $\neq \emptyset$, we obtain a contradiction with the definition of a simple chain grammar.

(iii) $\alpha_1 = \varepsilon$ and $\beta_1 \neq \varepsilon$. This case is symmetric to (ii).

(iv) $\alpha_1 = \varepsilon = \beta_1$. Then either $A = B = S$, hence the claim is satisfied, or $[A \to X\alpha_2]$ and $[B \to \cdot\, X\beta_2]$ are obtained from items $[C \to \gamma_1 \cdot U\gamma_2]$ and $[C \to \gamma_1 \cdot U'\gamma_2']$, respectively, in basis(s). If $U = U'$, then either U is not chain independent, which is impossible, or $A = B$, as desired. If $U \neq U'$, then, since FIRST(U) $\cap$ FIRST(U') $\neq \emptyset$, G is not a simple chain grammar.

This concludes the proof of the claim.

Now suppose that G is not LR(0). Then there is some LR(0) state set s of G which contains two or more inconsistent items. There are two cases (see Def. 2.4 in [5]):

(i) A *shift/reduce conflict.* There are two items $[A \to \alpha_1 \cdot a\alpha_2]$ and $[B \to \beta \cdot\,]$ in s, where $\alpha_1, \alpha_2, \beta \in V^*$ and $a \in T$. Since $\beta \neq \varepsilon$ we have that $[B \to \beta \cdot\,]$ is in basis(s). There are two cases:

(a) $\alpha_1 \neq \varepsilon$. It follows that $\alpha_1 = \beta$, $A = B$ and P is not prefix-free which is impossible.

(b) $\alpha_1 = \varepsilon$. In this case there exists a production $B \to \beta X\phi$ in P, where $X\phi \in NV^*$ and $X \Rightarrow_l^+ A\psi$ for some $\psi \in V^*$, and also in this case we have that P is not prefix-free, which is impossible.

(ii) A *reduce/reduce conflict.* There are two items $[A \to \alpha \cdot]$ and $[B \to \beta \cdot]$ in s. Since G is ϵ-free $\alpha \neq \varepsilon \neq \beta$ and both items belong to basis(s). It follows from the claim that $A = B$ and $\alpha = \beta$, so that, in fact, no conflict exists in s.

It follows that every simple chain grammar is an LR(0) grammar.

Observe that, since we are only concerned with ε-free grammars, the combination of Lemma 2.1 and Theorem 3.1 does not lead to the incorrect result that any LL(1) grammar (not necessarily ϵ-free) is an LR(0) grammar. Clearly every simple LL(1) grammar is a simple chain grammar. The class of simple chain grammars is properly included in the LR(0) grammars since the cfg with only productions $S \to aB$, $S \to eB$, $B \to cD$, $B \to cF$, $D \to b$ and $F \to b$ is LR(0) but it is not a simple chain grammar.

4. Simple chain languages

In this section we show that the class of simple chain languages coincides with the class of simple LL(1) (or simple deterministic) languages. First we show that every simple chain grammar can be transformed to an equivalent simple chain grammar in Greibach normal form (GNF). A transformation which is similar to ours can be found in [7] where it is shown that each strict deterministic grammar can be transformed to a strict deterministic grammar in GNF.

Observation. Let $G = (N, T, P, S)$ be a simple chain grammar. Let $A \in N$ and $a \in \text{FIRST}(A)$. Then the chain from A to a in CH(A) is uniquely determined and therefore also its length. Let this length be n_A^a. If $A \Rightarrow_l^n a\alpha$ for some $\alpha \in V^*$, then $n \geq n_A^a$.

Theorem 4.1. *Each simple chain grammar can be transformed to an equivalent simple chain grammar in* GNF.

Proof. Let $G = (N, T, P, S)$ be a simple chain grammar. Let

$$P' = \{A \to a\alpha \mid A \in N, \alpha \in V^* \quad \text{and } a \in T \text{ such that } A \overset{n_A^a}{\underset{l}{\Rightarrow}} a\alpha\}$$

and let $G' = (N, T, P', S)$. In this way G' is well defined, G' has no ε-productions and moreover G' is in pseudo-GNF. Cfg G' can be reduced in the usual way [1].

Claim 1. G' is a simple chain grammar.

Proof of Claim 1. Consider Def. 2.1. Assume P' is not prefix-free, so that there exist productions $A \to \alpha$ and $A \to \alpha\beta$ in P' with $\beta \neq \varepsilon$. Then by definition of P' there exist derivations $A \Rightarrow_l^* \alpha$ and $A \Rightarrow_l^* \alpha\beta$ in G with $\beta \neq \varepsilon$. Since G is a simple chain grammar it follows from Theorem 2.1 that this is not possible. Thus P' is prefix-free. Now assume there exist $A \in N$, $\alpha, \phi, \psi \in V^*$ and $X, Y \in V$ with $X \neq Y$ such that $A \to \alpha X\phi$ and $A \to \alpha Y\psi$ are in P'. Let $\alpha \neq \varepsilon$ and let ${}^{(1)}\alpha = a$. Then both derivations can be written as $A \Rightarrow_l^{n_A^a} \alpha X\phi$ and $A \Rightarrow_l^{n_A^a} \alpha Y\psi$ and by Theorem 2.3 $X \not\equiv Y$. If $\alpha = \varepsilon$, then X and Y are in T, $X \neq Y$ and hence $X \not\equiv Y$. This completes the proof of Claim 1.

It is not difficult to see that transforming G' in pseudo-GNF to a cfg in GNF by replacing terminals inside productions does not disturb the simple chain properties of G'. Therefore we may assume G' is in GNF.

Claim 2. $L(G) = L(G')$.

Proof of Claim 2. It is clear that for any $w \in T^*$, $S \Rightarrow_l^* w$ in G' implies $S \Rightarrow_l^* w$ in G. For the converse consider $A \Rightarrow_l^n w$ in G. If $n = 1$, then trivially also $A \Rightarrow_l w$ in G'. Suppose that $A \Rightarrow_l^* w$ in G implies $A \Rightarrow_l^* w$ in G' be true for all derivations of length less than n in G. Factor the derivation $A \Rightarrow_l^n w$ in G to get $A \Rightarrow_l^{n_A^a} a\alpha$, where it is assumed that ${}^{(1)}w = a$. By construction $A \to a\alpha$ is in P'. Let $\alpha = A_1 A_2 \cdots A_m \in N^*$. Notice that according to the remark following Claim 1 we may assume $\alpha \in N^*$. Each A_i derives a subword of w, that is $A_i \Rightarrow_l^* w_i$ in G for $1 \leq i \leq m$ and $w = aw_1 w_2 \cdots w_m$. Since these derivations are of length less than n, $A_i \Rightarrow_l^* w_i$ in G'. The combination of $A \to aA_1 \cdots A_n$ is in P' and $A_i \Rightarrow_l^* w_i$ in G' gives $A \Rightarrow_l^* w$ in G'. Therefore also $S \Rightarrow_l^* w$ in G implies $S \Rightarrow_l^* w$ in G'. It follows that $L(G') = L(G)$. This proves Claim 2 and the proof of Theorem 4.1 is now complete.

Example 4.1. Consider the cfg G with only productions $S \to Aa$, $S \to Ab$, $A \to BbS$, $A \to BdS$, $A \to a$ and $B \to c$. G is a simple chain grammar. The transformation (followed by reduction) yields a cfg with productions $S \to aa$, $S \to ab$, $S \to cbSa$, $S \to cdSa$, $S \to cbSb$ and $S \to cdSb$ which is a cfg in pseudo-GNF. Replacing the terminals yields $S \to aA$, $S \to aB$, $S \to cBSA$, $S \to cDSA$, $S \to cBSB$, $S \to cDSB$, $A \to a$, $B \to b$ and $D \to d$, which is a simple chain grammar in GNF.

The next step in this section is the transformation from a simple chain grammar in GNF to a simple LL(1) grammar (see Def. 1.2). This transformation is a simple process of *left factoring* until for each $a \in T$ and $A \in N$ there remains at most one production $A \to a\alpha$, for some $\alpha \in V^*$. The idea of left factoring, which amounts to replacing productions as $A \to \alpha\beta$ and $A \to \alpha\gamma$, $\beta \neq \gamma$, by productions $A \to \alpha Q$, $Q \to \beta$ and $Q \to \gamma$, is well-known and appears in many papers, among others in [12, 23]. The definition of simple chain grammars is such that this process of left factoring can always be continued in such a way that the resulting grammar is a simple LL(1) grammar.

Notation. All productions with left-hand side $A \in N$ whose right-hand sides start with $\alpha \in V^*$ are in $P(A, \alpha)$. Hence $P(A, \alpha) = \{A \to \alpha\phi \in P \mid \phi \in V^*\}$. Let $A \to \alpha_1$ and $A \to \alpha_2$ be two productions in P. The longest string $\alpha \in TN^*$ (since we may assume that the simple chain grammars are in GNF) such that α is both a prefix of α_1 and α_2, is called the *common* prefix of $A \to \alpha_1$ and $A \to \alpha_2$. Similarly we can define the common prefix of a set of productions. For example, the common prefix of a set $P(A, a)$ is the longest string $\alpha \in aN^*$ such that α is a prefix of all right-hand sides of the productions in $P(A, a)$.

Transformation. Let $G = (N, T, P, S)$ be a simple chain grammar in GNF. The cfg G will be transformed to a simple LL(1) grammar $G' = (N', T, P', S)$. Initially set $N' = N$ and $P' = \emptyset$ (the empty set).

Let R be the set consisting of all pairs (A, a) such that there is at least one production $A \to a\alpha$, for some $\alpha \in V^*$, in P. Hence

$$R = \{(A, a) \mid A \to a\alpha \in P \text{ for some } \alpha \in V^*\}.$$

The elements of R are numbered in an arbitrary way. Starting with the first element we shall consider for each element $(A, a) \in R$ the set $P(A, a)$. Initially $P(A, a) = \{A \to a\phi \in P \mid \phi \in N^*\}$. The set $P(A, a)$ is not fixed but will change in the course of the computation.

Step 1. (i) Let $|P(A, a)| = 1$. Then add the only production of $P(A, a)$ to P'. If all the elements of R have been considered go to Step 2. Otherwise start again with the next element of R.

(ii) Let $|P(A, a)| > 1$. Consider $\alpha \in aN'^*$ such that α satisfies

(a) α is a common prefix of at least two productions in $P(A, a)$, and

(b) there are no productions $A \to a\phi$ and $A \to a\psi$, $\phi \neq \psi$, in $P(A, a)$ with common prefix α' such that α is a proper prefix of α'.

If $|P(A, \alpha)| = n$, then denote its elements by $\{A \to \alpha X_i\phi_i \mid 1 \leq i \leq n\}$. Replace in $P(A, a)$ the subset $P(A, \alpha)$ by the single production $A \to \alpha[A\alpha, X_1\phi_1, \ldots, X_n\phi_n]$, where $[A\alpha, X_1\phi_1, \ldots, X_n\phi_n]$ is a newly introduced nonterminal which is added to N'. Repeat Step 1.

Step 2. For each newly introduced nonterminal of the form $Q = [B\beta, Y_1\psi_1, \ldots, Y_m\psi_m]$ add to P' for each i, $1 \leq i \leq m$, the set of productions $\{Q \to \gamma\psi_i \mid Y_i \to \gamma \in P'\}$.

Step 3. Remove all useless nonterminals and productions.

Remark. In general α in Step 1 (ii) is not uniquely determined. If there is more than one such α, then it does not matter which one is taken first. Notice that since G is in GNF the strings $Y_i\psi_i$, $1 \leq i \leq m$, are in N'^*. A newly introduced nonterminal $[B\beta, Y_1\psi_1, \ldots, Y_m\psi_m]$ is associated by $B\beta$ with the productions in $P(B, \beta)$ from which it is obtained. This association is not necessary but it is done to facilitate the proof which will follow.

Example 4.2. Consider the simple chain grammar in GNF with the following list of productions:

$$\begin{array}{llll} S \to cA & A \to aBD & A \to aAB & D \to e \\ A \to aBCBD & A \to aBA & A \to f & B \to b \\ A \to aBCBA & A \to aACA & D \to d & C \to c. \end{array}$$

The subsequent results of Step 1 on $P(A, a)$ can be given in the following order:

(1) For $\alpha = aBCB$. $A \to aBCBD$ and $A \to aBCBA$ are replaced by $A \to aBCBQ_0$, where $Q_0 = [AaBCB, D, A]$.

(2) For $\alpha = aB$. $A \to aBCBQ_0$, $A \to aBD$ and $A \to aBA$ are replaced by $A \to aBQ_1$, where $Q_1 = [AaB, CBQ_0, D, A]$.

(3) For $\alpha = aA$. $A \to aACA$ and $A \to aAB$ are replaced by $A \to aAQ_2$, where $Q_2 = [AaA, CA, B]$.

(4) For $\alpha = a$ (the common prefix of $P(A, a)$). $A \to aAQ_2$ and $A \to aBQ_1$ are replaced by $A \to aQ_3$, where $Q_3 = [Aa, AQ_2, BQ_1]$.

The results of Step 2 for Q_0, Q_1, Q_2 and Q_3 are

$$\begin{array}{llll} Q_0 \to d & Q_1 \to cBQ_0 & Q_1 \to f & Q_3 \to fQ_2 \\ Q_0 \to e & Q_1 \to d & Q_2 \to cA & Q_3 \to bQ_1 \\ Q_0 \to aQ_3 & Q_1 \to e & Q_2 \to b & \\ Q_0 \to f & Q_1 \to aQ_3 & Q_3 \to aQ_3Q_2. & \end{array}$$

Theorem 4.2. *Each simple chain grammar can be transformed to an equivalent simple* LL(1) *grammar.*

Proof. By Theorem 4.1 we may assume that $G = (N, T, P, S)$ is a simple chain grammar in GNF. Let $G' = (N', T, P', S)$ be the cfg which is obtained by the preceding transformation. The proof that G' is a simple LL(1) grammar which is equivalent to G is divided into three claims.

Claim 1. Let $Q = [A\alpha, X_1\phi_1, \ldots, X_n\phi_n]$ be a newly introduced nonterminal. Then each X_i, $1 \leq i \leq n$, is in N and if $i \neq j$, where $1 \leq i, j \leq n$, then $X_i \neq X_j$ and $X_i \not\equiv X_j$.

Proof of Claim 1. Observe that the prefix α in Step 1 (ii) is always in TN^* (that is, it does not contain newly introduced nonterminals). Moreover, since all the productions in $P(A, \alpha)$ are considered at the same time we can not have productions $A \to \alpha Q'\phi$ for some newly introduced Q' and $\phi \in N'^*$ and $A \to \alpha B\psi$ for some $B \in N$ and $\psi \in N'^*$. Thus each $X_i\phi_i$ in Q has $X_i \in N$. Moreover, for $i \neq j$, $X_i \neq X_j$ since otherwise the α which was chosen was not the longest applicable prefix as is demanded in Step 1 (ii)(b). Since $X_i, X_j \in N$ there exist productions $A \to \alpha X_i\phi$ and $A \to \alpha X_j\psi$ in P, for some ϕ and ψ in V^*. For $i \neq j$ we have $X_i \neq X_j$ and since P is the set of productions for the simple chain grammar G, we have $X_i \not\equiv X_j$.

Claim 2. G' is a simple LL(1) grammar.

Proof of Claim 2. We have to show that for each nonterminal A in N' and each terminal $a \in T$ there is at most one production $A \to a\alpha$ in P', for some $\alpha \in V'^*$. A set $P(A, a)$, where $A \in N$ and $a \in T$, is reduced to only one production whose right-hand side has as prefix the common prefix of $P(A, a)$. After Step 1 has been performed, for each $A \in N$ and $a \in T$ there is at most one production $A \to a\alpha$ in P' for some $\alpha \in N'^*$.

In Step 2 productions are introduced for the new nonterminals of the form $Q = [A\alpha, X_1\phi_1, \ldots, X_n\phi_n]$. Since by Claim 1 $X_i \not\equiv X_j$ for $i \neq j$ we can not have $X_i \to a\gamma$ and $X_j \to a\gamma'$ for some $a \in T$ and $\gamma, \gamma' \in V'^*$. Therefore, for any newly introduced Q and for any $a \in T$ there is also at most one production in $P(Q, a)$. This concludes the proof that G' is a simple LL(1) grammar.

Claim 3. $L(G) = L(G')$.

Proof of Claim 3. In Fig. 1 the transformation is illustrated. Only local transformations as presented in this figure are performed. Therefore the transformation is language preserving.

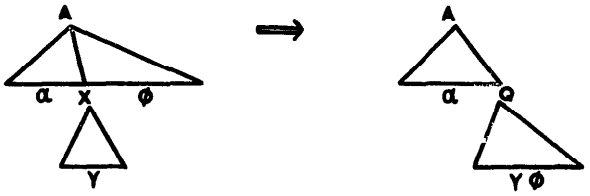

Fig. 1. Transformation to simple LL(1) grammars.

From Claim 2 and Claim 3 it follows that the transformation yields an equivalent simple LL(1) grammar.

Example 4.3. We give the results of the transformation applied on two simple chain grammars. Let G_1 be the cfg with productions

$$S \to aSA \qquad A \to bS$$
$$S \to aA \qquad A \to c.$$

Then the transformation yields

$$S \to aQ \qquad Q \to c$$
$$Q \to aQA \qquad A \to bS$$
$$Q \to bS \qquad A \to c.$$

Let G_2 be the cfg with productions

$$\begin{array}{lll} S \to aSA & A \to dC & B \to b \\ S \to aA & A \to b & C \to c. \\ A \to dB & A \to c & \end{array}$$

Then the transformation yields

$$\begin{array}{lll} S \to aQ_0 & Q_1 \to b & A \to b \\ Q_0 \to aQ_0A & Q_1 \to c & A \to c. \\ Q_0 \to dQ_1 & A \to dQ_1 & \end{array}$$

Since each simple LL(1) grammar is a simple chain grammar and since it is decidable whether two simple LL(1) grammars are equivalent [13], we have the following corollary.

Corollary 4.1. (i) *The simple chain grammars generate exactly the class of simple* LL(1) *(or simple deterministic) languages.*

(ii) *It is decidable whether two simple chain grammars are equivalent.*

In the second part of this section we take a more general look at transformations from simple chain grammars to simple LL(1) grammars. Some of the difficulties which appear if we try to describe structure-preserving properties of transformations of context-free grammars are already present at the level of simple chain grammars. First we need some preliminaries on deterministic pushdown transducers and on covers.

Definition 4.1. A *deterministic pushdown transducer* (dpdt for short) is an eight-tuple $P = (Q, T, \Gamma, \Delta, \delta, q_0, Z_0, F)$ where Q is a finite set of *states*. T, Γ, and Δ are alphabets and δ is a mapping from $Q \times (T \cup \{\varepsilon\}) \times \Gamma$ to $Q \times \Gamma^* \times \Delta^*$ such that if $\delta(q, a, Z)$ is defined, then $\delta(q, \varepsilon, Z)$ is undefined and if $\delta(q, \varepsilon, Z)$ is defined, then $\delta(q, a, Z)$ is undefined for all $a \in T$. Further, $q_0 \in Q$ is the *initial state*, $Z_0 \in \Gamma$ is the *start symbol*, and $F \subseteq Q$ is the set of *accepting states*. A *configuration* of P is a four-tuple (q, w, α, y) in $Q \times T^* \times \Gamma^* \times \Delta^*$. If $\delta(q, a, Z) = (r, \alpha, z)$ we write $(q, ax, Z\gamma, y) \vdash (r, x, \alpha\gamma, yz)$. In the usual way the move $\vdash$ is extended to $\vdash^+$ and $\vdash^*$. The *translation* defined by P is the set

$$\tau(P) = \{(x, y) \mid (q_0, x, Z_0, \varepsilon) \overset{*}{\vdash} (q, \varepsilon, \alpha, y) \text{ for some } q \in F \text{ and } \alpha \in \Gamma^*\}.$$

The language accepted by P is the set $L(P) = \{x \mid (x, y) \in \tau(P)\}$. $L(P)$ is said to be a *deterministic language*. A dpdt is said to be a *simple* dpdt if it has only one state, it has no ε-rules (i.e., rules of the form $\delta(q, \varepsilon, Z) = \ldots$, for some $Z \in \Gamma$ and $q \in Q$) and after the input is accepted the pushdown stack is empty.

We assume the reader is familiar with these concepts and we do not go into details. The second condition will turn out to be essential for the main result of the remainder of this section.

Definition 4.2. Let $G = (N, T, P, S)$ and $G' = (N', T, P', S')$ be two cfg's. Let x, $y \in$ {'left', 'right'} and let $h : P'^* \to P^*$ be a homomorphism such that

(i) if π' is an x-parse for $w \in L(G')$ with respect to G', then $h(\pi')$ is a y-parse for w with respect to G, and

(ii) if π is a y-parse for $w \in L(G)$ with respect to G, then there exists π' such that $h(\pi') = \pi$ and π' is an x-parse of w with respect to G'.

If in (i) and (ii) both x and y are replaced by 'left', then G' is said to *left-cover* G. If x is replaced by 'left' and y is replaced by 'right', then we say that G' *left-to-right covers* G. If both x and y are replaced by 'right' then G' is said to *right-cover* G.

Now consider the following relationships between covers and dpdt's. For more details the reader is referred to [17]. Suppose the dpdt R acts as a left parser for the cfg G', that is, for each $w \in L(G')$ R accepts w and R produces a left parse of w with respect to G' and if $w \notin L(G')$, then w is not accepted. Let G' left cover a cfg G with the associated cover-homomorphism h. For each $q, r \in Q$, $a \in T$, $Z \in \Gamma$, $\alpha \in \Gamma^*$ and $y \in \Delta^*$ such that $\delta(q, a, Z) = (r, \alpha, y)$, replace this rule by $\delta(q, a, Z) = (r, \alpha, h(y))$. Then it is clear that the resulting dpdt is a left parser for G. In case G' left-to-right covers G, then the resulting dpdt is a right parser for G.

Armed with these observations we can attack the cover problems. There exist cfg's for which there is no dpdt which acts as a left parser ([17]). One can easily verify that one of these grammars is the simple chain grammar G with productions $S \to aEc$, $S \to aEd$, $E \to aEb$ and $E \to ab$ (Example 2.3).

Now suppose G is left covered by some simple LL(1) grammar G'. It is obvious how to construct a (simple) dpdt for G' which acts as a left parser. However, since G' left covers G we can replace the rules of this dpdt in such a way that we obtain a left parser for G. But this is in contradiction with the property of G that there is no such left parser. Hence we can conclude that any transformation from simple chain grammars to simple LL(1) grammars will not, in general, yield a left cover.

Now we consider the possibility of a left-to-right cover. From a point of view of parsing this is the interesting case. Instead of parsing with respect to the simple chain grammar (to obtain right parses) we would like to parse with respect to the simple LL(1) grammar (which would yield left parses). Now, if the simple LL(1) grammar left-to-right covers the simple chain grammar, then we can parse with respect to the simple LL(1) grammar and by applying the cover-homomorphism on the left parses we can obtain the right parses with respect to the simple chain grammar. Unfortunately the transformation which we gave in the first part of this section does not yield such a left-to-right cover.

Example 4.4. Consider the cfg G_1 with (labeled) productions

1. $S \to aBC$
2. $S \to aBD$
3. $B \to aB$
4. $B \to d$
5. $B \to e$
6. $C \to c$
7. $D \to d$.

Left-factoring G_1 in the way we did in this section does not yield a left-to-right cover. In this case left-factoring yields the cfg G_1' with productions

1. $S \to aBQ$
2. $B \to aB$
3. $B \to d$
4. $B \to e$
5. $Q \to c$
6. $Q \to d$.

One can easily verify that left parses with respect to G_1' can not be mapped by any cover-homomorphism on right parses of G_1.

Clearly this example does not exclude the possibility of the existence of a transformation (different from ours) from simple chain grammars to simple LL(1) grammars which will yield such a left-to-right cover. We can, however, use the same type of reasoning as given above to show the non-existence of such a transformation. First we have the following claim which again makes use of the cfg G_1 of Example 4.4.

Claim. *There does not exist a* dpdt *which acts as a right parser for* G_1 *and which has no* ε*-rules.*

Proof. (sketch). $L(G_1) = L_1 \cup L_2 \cup L_3 \cup L_4$, where $L_1 = \{a^{n+1}dc \mid n \geq 0\}$, $L_2 = \{a^{n+1}ec \mid n \geq 0\}$, $L_3 = \{a^{n+1}dd \mid n \geq 0\}$ and $L_4 = \{a^{n+1}ed \mid n \geq 0\}$. The sets of right parses for these sets are $R_1 = \{4\ 3^n\ 6\ 1 \mid n \geq 0\}$ $R_2 = \{5\ 3^n\ 6\ 1 \mid n \geq 0\}$, $R_3 = \{4\ 3^n\ 7\ 2 \mid n \geq 0\}$ and $R_4 = \{5\ 3^n\ 7\ 2 \mid n \geq 0\}$, respectively. For any dpdt which acts as a right parser the first symbol on the output tape should be a 4 or a 5, depending on whether $a^{n+1}d$ or $a^{n+1}e$ is a prefix of the string which has to be parsed. In both cases the dpdt can not emit this first symbol until the d or e has been read. After a^{n+1} has been read there are only two symbols left on the input tape while an unbounded amount of output symbols must be generated. Therefore the dpdt needs ε-rules.

Now suppose there exists a simple LL(1) grammar G_1' which left-to-right covers G_1. It is obvious how to construct a (simple) dpdt for G_1' which acts as a left parser and which has no ε-rules. Since G_1' left-to-right covers G_1 we can change the rules of this dpdt in such a way that we obtain a right parser (without ε-rules) for G_1. This contradicts the claim and therefore we must conclude that there is no such left-to-right cover.

Corollary 4.2. *There is no transformation from simple chain grammars to simple* LL(1) *grammars which yields a left-cover or a left-to-right cover.*

This negative result for left-to-right covers can readily be extended to simple LL(1) grammars, that is, there is no transformation from simple LL(1) grammars to simple LL(1) grammars which yields a left-to-right cover; the cfg with productions $S \to aB$, $B \to aB|b|c$ is a simple LL(1) grammar and cannot be left-to-right covered with a simple LL(1) grammar. A proof similar to the argument used above is straightforward.

As a last result of this section we consider the construction of dpdt which acts as a right parser for a simple chain grammar.

Construction 4.1. Let $G = (N, T, P, S)$ be a simple chain grammar. Let the elements of P be numbered and let Δ be the set which consists of these numbers. Let $R = (\{q\}, T, \Gamma, \Delta, \delta, q, [S], \{q\})$, be a dpdt, where

$$\Gamma = \{[A\alpha] \mid A \to \alpha\beta \text{ in } P \text{ for some } A \in N \text{ and } \alpha, \beta \in V^*\}.$$

Instead of $[A\varepsilon]$ we write $[A]$. The function δ is defined in the following way:

(i) For each $i = A \to \alpha$ in P let $\delta(q, \varepsilon, [A\alpha]) = (q, \varepsilon, i)$.

(ii) Let $[A\alpha] \in \Gamma$ such that $A \to \alpha\beta$ in P for some $\beta \neq \varepsilon$. If $X_0X_1 \cdots X_n \in CH^{(1)}(\beta)$, then

(a) if $X_0 = X_n \in T$, then $\delta(q, X_n, [A\alpha]) = (q, [A\alpha X_0], \varepsilon)$, otherwise

(b) $\delta(q, X_n, [A\alpha]) = (q, [X_{n-1}X_n] \cdots [X_0X_1][A\alpha X_0], \varepsilon)$[1].

The proof that this construction indeed yields a well-defined dpdt which acts as a right parser is straightforward and is therefore left to the reader.

5. Conclusions

In this paper we have introduced a proper subclass of the LR(0) grammars, the class of simple chain grammars. We have shown that every simple chain grammar is prefix-free and their languages coincide with the simple deterministic languages. Problems concerning covers were investigated and a deterministic pushdown transducer which acts as a right parser was constructed.

We presented a transformation from a simple chain grammar to a simple deterministic grammar. Another transformation, which does not utilize the properties of simple chain grammars (and which can be used for any non-left-recursive grammar)

[1] Notice that the top of the pushdown stack is on the left.

will be used in [21]. For cover results which are not in this paper the reader is referred to [20]. Extension of the definition of a simple chain grammar with look-ahead will give rise to questions concerning relationships with classes of grammars defined in [9] and [22].

Acknowledgment

I am indebted to Prof. L.A.M. Verbeek for introducing me to the issues of parsing and covers. I wish to thank the referees for many helpful suggestions.

References

[1] A.V. Aho and J.D. Ullman, *The Theory of Parsing, Translation and Compiling*, Vols. 1 and 2 (Prentice Hall, Englewood Cliffs, NJ 1972 and 1973).
[2] M.E. Conway, Design of a separable transition-diagram compiler, *Comm. ACM* **6** (1963) 396–408.
[3] F.L. DeRemer, Simple LR(k) grammars, *Comm. ACM* **14** (1971) 453–460.
[4] M.M. Geller, S.L. Graham and M.A. Harrison, Production prefix parsing, in: J. Loeckx, Ed., *Automata, Languages and Programming*, Lecture Notes in Computer Science **14** (Springer, Berlin, 1974) 232–241.
[5] M.M. Geller and M.A. Harrison, Characteristic parsing: A framework for producing compact deterministic parsers, I and II, *J. Comput. System Sci.* **14** (1977) 265–317 and 318–345.
[6] M.M. Geller and M.A. Harrison, On LR(k) grammars and languages, *Theoret. Comput. Sci.* **4** (1977) 245–276.
[7] M.M. Geller, M.A. Harrison and I.M. Havel, Normal forms of deterministic grammars, *Discrete Math.* **16** (1976) 313–322.
[8] J.Gray and M.A. Harrison, On the covering and reduction problems for context-free grammars, *J. ACM* **19** (1972) 385–395.
[9] M. Hammer, A new grammatical transformation into LL(k) form, in: *Conference Record of the Sixth Annual ACM Symposium on Theory of Computing* (1974) 266–275.
[10] M.A. Harrison and I.M. Havel, Strict deterministic grammars, *J. Comput. System Sci.* **7** (1973) 237–277.
[11] M.A. Harrison and I.M. Havel, On the parsing of deterministic languages, *J. ACM* **21** (1974) 525–548.
[12] D.E. Knuth, Top-down syntax analysis, *Acta Informat.* **1** (1971) 79–110.
[13] A.J. Korenjak and J. E. Hopcroft, Simple deterministic languages, in: *IEEE Conference Record of the Seventh Annual Symposium on Switching and Automata Theory* (1966) 34–46.
[14] J.Kràl, Bottom up versus top down syntax analysis revised, Research Rep. UVT 10-11/74, Institute of Computation Technique, Technical University of Prague (1974).
[15] J. Kràl and J. Demner, A note on the number of states of the DeRemer's recognizer. *Information Processing Lett.* **2** (1973) 22–23.
[16] D.B. Lomet, Automatic generation of multiple exit parsing subroutines in : J. Loeckx, Ed., *Automata, Languages and Programming*, Lecture Notes in Computer Science **14** (Springer, Berlin, 1974) 214–231.
[17] A. Nijholt, On the covering of parsable grammars, *J. Comput. System Sci.* **15** (1977) 99–110.
[18] A. Nijholt, Simple chain grammars, in: A. Salomaa and M. Steinby, Eds., *Automata, Languages and Programming*, Lecture Notes in Computer Science **52** (Springer, Berlin, 1977) 352–364.
[19] A. Nijholt, Cover results and normal forms, in: J. Gruska, Ed., *Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science **53** (Springer, Berlin, 1977) 420–429.

[20] A. Nijholt, On the parsing and covering of simple chain grammars, in: G. Ausiello and C. Böhm, Eds., *Automata, Languages and Programming*, Lecture Notes in Computer Science **62** (Springer, Berlin 1978) 330–344.

[21] A. Nijholt, A left part theorem for grammatical trees, *Discrete Math.* **25** (1979) 51–63.

[22] E. Soisalon-Soininen, Characterization of LL(k) languages by restricted LR(k) grammars, Ph.D. Thesis, Rep A-1977-3, Dept. of Comput. Sci., University of Helsinki.

[23] D. Wood, The theory of left factored languages, *Comput. J.* **12** (1969) 349–356 and **13** (1970) 55–62.