BIT 19 (1979), 73–78

GRAMMAR FUNCTORS AND COVERS: FROM NON-LEFT-RECURSIVE TO GREIBACH NORMAL FORM GRAMMARS

ANTON NIJHOLT

Abstract.

Attention is paid to structure preserving properties of transformations from a non-left-recursive context-free grammar to a Greibach normal form grammar. It is demonstrated that such a transformation cannot only be ambiguity preserving, but also both cover and functor relations between grammars or their associated syntax-categories can be obtained from such a transformation.

Keywords: context-free grammar, cover, syntax-category, grammar functor, Greibach normal form.

1. Introduction.

In this paper we consider some structure preserving properties of transformations from non-left-recursive context-free grammars to context-free grammars in Greibach normal form (GNF). Two approaches are followed. In the syntax-categorical approach of transformations one is interested in the existence of a (grammar) functor between certain categories associated with each grammar. The existence of such a grammar functor leads to the conclusion that the syntactic structure has been preserved by the transformation. In the cover approach of transformations (see for example Gray and Harrison [3] and Aho and Ullman [1]) the primary interest is the relationship between the parses of the grammar which is obtained after the transformation and the parses of the original grammar. If such a simple relationship (a homomorphism) exists then the new grammar can be used for parsing and its parses can be mapped on the parses of the original grammar. The categorical approach for this special transformation to GNF has been discussed by Benson [2].

Preliminaries.

The length of a string α is denoted by $|\alpha|$; if $|\alpha| \geqq k$ then ${}^{(k)}\alpha$ denotes the first k symbols of α, else it denotes α; analogously, we have $\alpha^{(k)}$ for the last k symbols of α. The empty string will be denoted by ε.

Let $G = (N, \Sigma, P, S)$ be a context-free grammar (CFG). Elements of N (nonterminal symbols) will be denoted by upper-case Roman letters, elements of

Received July 10, 1978. Revised October 10, 1978.

Σ (terminal symbols) by lower-case Roman letters and elements of $V^* = (N \cup \Sigma)^*$ by lower-case Greek letters. The productions in $P \subseteq N \times V^+$ are assumed to be uniquely numbered. S is the startsymbol.

A leftmost derivation of $\alpha \in V^*$ from $A \in N$ is denoted by $A \overset{\pi}{\underset{l}{\Rightarrow}} \alpha$, where π (the left parse) is the sequence of numbers of the productions which are used, in the proper order, in this derivation. The CFG G is said to be non-left-recursive if there are no derivations of the form $A \overset{+}{\underset{l}{\Rightarrow}} A\alpha$. CFG G is said to be in Greibach normal form (GNF) if all productions are of the form $A \to a\alpha$, where $a \in \Sigma$ and $\alpha \in N^*$. The grammars in this paper are assumed to be proper (Aho and Ullman [1]). The language of a CFG G is denoted by $L(G)$.

For the concept of syntax category we follow Benson [2]. Let $G = (N, \Sigma, P, S)$ be a CFG. Then (V^*, P) generates the free strict monoidal category $S(G)$ which will be called the syntax category of G. Here, objects are elements of V^* and morphisms are derivations (or better, equivalence classes of similar derivations) from one object to the other.

A well-known relation for morphisms is the following:

$$(f_1 \circ g_1) + (f_2 \circ g_2) = (f_1 + f_2) \circ (g_1 + g_2)$$

where we assume that domains and codomains are defined properly. Here, $\circ$ denotes composition of morphisms (in general $\circ$ will be omitted) and $+$ stands for the concatenation of morphisms, that is, if $h_1: \alpha \to \beta$ and $h_2: \gamma \to \delta$, then $h_1 + h_2: \alpha\gamma \to \beta\delta$ stands for the derivation which is composed by first going from $\alpha\gamma$ to $\beta\gamma$ by h_1 and then from $\beta\gamma$ to $\beta\delta$ by h_2.

For each object α, the categorical identity at α is denoted by $i_\alpha: \alpha \to \alpha$.

2. Grammar functors and covers.

First we recall the definitions of left cover (Aho and Ullman [1]) and externally fixed grammar functor (Benson [2]).

DEFINITION 2.1. A CFG $G' = (N', \Sigma, P', S')$ is said to left cover a CFG $G = (N, \Sigma, P, S)$ if there exists a (cover-)homomorphism $\varphi: P' \to P^*$ such that

(i) if $S' \overset{\pi'}{\underset{l}{\Rightarrow}} w$, then $S \overset{\varphi(\pi')}{\underset{l}{\Longrightarrow}} w$, and
(ii) if $S \overset{\pi}{\underset{l}{\Rightarrow}} w$, then there exists π' such that $S' \overset{\pi'}{\underset{l}{\Rightarrow}} w$ and $\varphi(\pi') = \pi$.

We do not distinguish between productions and their numbers. Notice that we have $\varphi: P' \to P^*$ instead of $\varphi: P' \to P$, that is, φ may map a production of P' on a (possibly empty) string of productions of P. From the definition it follows that $L(G') = L(G)$. Given a sentence $w \in L(G)$, its degree of ambiguity with respect to G, notation $\langle w, G \rangle$, is the number of left parses for w. It follows that $\langle w, G \rangle \leqq \langle w, G' \rangle$.

DEFINITION 2.2. Let $G' = (N', \Sigma', P', S')$ and $G = (N, \Sigma, P, S)$ be two context-free grammars. A grammar functor $F: S(G') \to S(G)$ is a functor which preserves concatenation (both for objects and morphisms) and the empty string, and which is defined as follows

(i) $F(A) \in V^*$ for all $A \in N'$,
(ii) $F(a) \in \Sigma^*$ for all $a \in \Sigma'$, and
(iii) $F(S') = S$.

F is externally fixed if $\Sigma' = \Sigma$ and $F(a) = a$ for all $a \in \Sigma'$.

Except for condition (i) Definition 2.2 is similar to the definition in Benson [2]. In that paper $F(A)$ is in N^*, for each $A \in N'$. To define a grammar functor $F\colon S(G') \to S(G)$ it is in general sufficient to define F on $V' = (N' \cup \Sigma')$ and on P'. Free generation takes care of the rest. In this paper we will only make use of externally fixed grammar functors. Henceforth grammar functor will usually stand for externally fixed grammar functor. From Definition 2.2. it follows that $L(G') \subseteq L(G)$.

Notice that without further restrictions on F we cannot relate $\langle w, G' \rangle$ to $\langle w, G \rangle$. A grammar functor F can be restricted to the HOM-sets of $S(G')$. Let $\alpha, \beta \in V'^*$, then the HOM-set of (α, β) is denoted by $S(G')(\alpha, \beta)$ and the restriction of F to this HOM-set is denoted by

$$F(\alpha, \beta)\colon S(G')(\alpha, \beta) \to S(G)(F(\alpha), F(\beta))\colon f \to F(f)\,.$$

In the case of covers we are only interested in the relationship between derivations from the start symbol to sentences (represented by parses) of G' and G. That is, in terms of functors, we are only interested in relations between $S(G')(S', w)$ and $S(G)(S, w)$ for each $w \in L(G')$.

The notion of cover as defined in Definition 2.1 can now be compared with a grammar functor F which has the property that for each $w \in L(G')$, $F(S', w)$ is a surjection. In this case $L(G') = L(G)$ and $\langle w, G' \rangle \geqq \langle w, G \rangle$.

For grammar functors we can go into more details. A grammar functor is said to be full if for each pair of objects $\alpha, \beta \in V'^*$, $F(\alpha, \beta)$ is surjective; it is said to be externally full if for each pair of objects $\alpha \in V'^*$ and $w \in \Sigma'^*$, $F(\alpha, w)$ is surjective. A grammar functor is faithful if for each pair of objects (α, β), $F(\alpha, \beta)$ is injective. Obviously, for covers we are interested in injectivity of $F(S', w)$. If a cover-homomorphism is such that $\langle w, G' \rangle = \langle w, G \rangle$ then we call the cover faithful.

3. A transformation to Greibach normal form.

The usual transformation (Aho and Ullman's algorithm 2.14) to a CFG G' in GNF from a non-left-recursive CFG G is in general not ambiguity preserving. That is, we can have the situation that $\langle w, G' \rangle$ is less than $\langle w, G \rangle$. In this case we can not expect to have a left cover from G' to G or a grammar functor F such that $F(S', w)$ is surjective for all $w \in L(G')$.

Despite this negative observation it is possible to obtain a CFG G' in GNF from a non-left-recursive grammar G in such a way that G' left covers G. A similar result can be obtained for grammar functors. To show this we will use a variant of the standard method.

The standard transformation to GNF is also used in Benson [3]. It is claimed that this yields an externally fixed grammar functor which is faithful and externally full. However, to obtain this result the transformation which is given yields a CFG G' in GNF which may have productions which are the same except for their index. One may consider this as introducing semantical ambiguity (Aho and Ullman [1]), that is, syntactical ambiguity is replaced by semantical ambiguity.

We consider proper and non-left-recursive grammars which may have single productions (i.e. of the form $A \to B$). Such a grammar is said to be very proper if no single production has left-hand side S. Note that due to this condition a situation $S \overset{\pi}{\Longrightarrow} a$, $S \overset{\varrho}{\Longrightarrow} a$ with $a \in \sum$ and $\pi \neq \varrho$ cannot occur. In this situation a cover, for any transformation to GNF, cannot be defined.

The transformations which are used in this section are simple and only require straightforward proofs. Therefore detailed proofs are omitted. Since the cover relation is transitive we can obtain the cover result by composing different steps. The first algorithm deals with single productions. After the algorithms there is a short discussion on the grammar functor result.

In the algorithms we start with the productions of P. If $A \to \alpha$ is the ith production in P then we write $A \to \alpha\langle i \rangle$. In the subsequent steps of the algorithm new productions are introduced. The cover-homomorphism will be defined implicitly in the algorithm with the notation $A \to \alpha\langle \pi \rangle$, that is, a newly obtained production $A \to \alpha$ will be mapped on a string π of productions of P.

Algorithm 3.1.

Input. A very proper non-left-recursive CFG $G = (N, \Sigma, P, S)$.

Output. A CFG $G' = (N', \Sigma, P', S)$ such that G' has no single productions and G' left covers G. Moreover, the cover is faithful.

Method. Let $\bar{P}$ be the subset of P which consists of all single productions. Initially set $P_1 = P - \bar{P}$ and $N' = N$. Now there are three steps.

(i) Let $B \in N$, $B \overset{\pi'}{\Longrightarrow} C \overset{i}{\Longrightarrow} \gamma$ in G, with $\pi' \neq \varepsilon$ and where $|\gamma| \geqq 2$ or $\gamma \in \Sigma$. Set $\pi = \pi' i$. Add $[B\pi] \to \gamma\langle \pi \rangle$ to P_1 and $[B\pi]$ to N'.

(ii) Set $P' = \emptyset$. Define a homomorphism h: $N' \cup \Sigma \to N \cup \Sigma$, by defining $h(X) = X$ for each $X \in N \cup \Sigma$, and $h([A\pi]) = A$ for each $[A\pi] \in N' - N$. For each production $C \to \gamma\langle \pi \rangle$ in P_1 (hence, $C \in N'$ and $\gamma \in (N \cup \Sigma)^+$) add the set $\{C \to \gamma'\langle \pi \rangle \mid C \to \gamma\langle \pi \rangle$ in P_1 and $h(\gamma') = \gamma\}$ to P'.

(iii) Remove all useless symbols. ∎

In the following algorithm we assume that in the right-hand sides of the productions of the input grammar a terminal symbol can only occur in the leftmost position. This can be done without loss of generality; for example, a production $A \to \alpha a \beta \langle \pi \rangle$, $\alpha \neq \varepsilon$, can be replaced by $A \to \alpha H_a \beta \langle \pi \rangle$ and $H_a \to a \langle \varepsilon \rangle$.

Algorithm 3.2.

Input. A proper, non-left-recursive CFG $G = (N, \Sigma, P, S)$ where all productions are of the form $A \to \alpha$, with $\alpha \in \Sigma N^* \cup NN^+$.

Output. A CFG $G' = (N', \Sigma, P', S)$ in GNF, G' left covers G and the cover is faithful.

Method. Let $\bar{P}$ be the subset of P which consists of all productions of the form $A \to a\alpha$, $\alpha \in N^*$. Set $P_1 = \bar{P}$ and $N' = N$. There are three steps.

(i) For each $A \in N$ and $a \in \Sigma$, if in G, $A \overset{\pi'}{\underset{l}{\Rightarrow}} C\alpha' \overset{i}{\underset{l}{\Rightarrow}} a\alpha$, where $\pi' \neq \varepsilon$, $C \in N$ and $\alpha', \alpha \in N^+$, then add $A \to a[^{(1)}\alpha\pi]\alpha^{(r-1)}\langle \pi \rangle$ to P_1. Here, $\pi' i = \pi$, $|\alpha| = r$, and $[^{(1)}\alpha\pi]$ is a newly introduced nonterminal which is added to N'.

(ii) Set $P' = P_1$. Let $[A\pi]$ be a newly introduced nonterminal symbol. Then, for each $\gamma \in \Sigma N'^*$ such that $A \to \gamma\langle \varrho \rangle$ is in P_1, add the production $[A\pi] \to \gamma\langle \varrho \rangle$ to P'.

(iii) Remove all useless symbols. ∎

In the following Corollary the results of the two algorithms are collected.

COROLLARY 3.1. Each very proper non-left-recursive CFG $G = (N, \Sigma, P, S)$ is left covered by a CFG G' in GNF. The cover is faithful.

With some simple observations we show the existence of an externally fixed and externally full and faithful grammar functor $H: S(G') \to S(G)$ for any proper and non-left-recursive CFG G which is transformed by our Algorithms 3.1 and 3.2 to a CFG G' in GNF. Notice that faithfulness and external fullness are preserved under functor composition. Since in each production $C \to \gamma\langle \pi \rangle$ which appears in the transformations π stands for a corresponding leftmost derivation in the image grammar (notice that for less simple left covers this is not always necessary) we can use the notation π also to denote the corresponding morphism. To avoid confusion we will use $\bar{\pi}$ for the morphism. For example, a leftmost derivation

$$A \overset{1}{\underset{l}{\Rightarrow}} BD \overset{2}{\underset{l}{\Rightarrow}} CcD \overset{3}{\underset{l}{\Rightarrow}} bcD$$

has a corresponding left parse 123, while the corresponding morphism of $S(G)$ is $1 \circ (2 + i_D) \circ (3 + i_{cD})$. This morphism follows uniquely from the string 123.

CLAIM 3.1. Algorithm 3.1 yields a CFG G' without single productions and such that there exists an externally fixed and externally full and faithful grammar functor $H: S(G') \to S(G)$.

PROOF. We have to define H on the objects and the morphisms of $S(G')$. For the objects it is sufficient to define H on V', which is done by letting $H(X)=X$ for each $X \in V$ and $H([B\pi])=B$ for each $[B\pi] \in N'-N$. For the morphisms it is sufficient to define H on P', which is done by letting $H(A \to \alpha'\langle\pi\rangle)=\bar{\pi}$ for each production $A \to \alpha'\langle\pi\rangle$ in P'.

Notice that this is a proper definition. If $A \to \alpha'\langle\pi\rangle$ is in P' then π represents a leftmost derivation from $H(A)$ to $H(\alpha')$, since H on N' coincides with homomorphism h in step (ii) of the algorithm, and π runs from $h(A)$ to $h(\alpha')$. Now the proof can proceed in a straight forward way. ∎

The condition mentioned before Algorithm 3.2 can also be handled functorially. For instance, for the given example the functor H should satisfy $H(H_a)=a$, $H(A \to \alpha H_a \beta)=A \to \alpha a \beta$ and $H(H_a \to a)=i_a$. This can be generalized in an obvious way and clearly such a functor is faithful and externally full.

CLAIM 3.2. Algorithm 3.2 yields a CFG G' in GNF such that there exists an externally fixed, externally full and faithful grammar functor H: $S(G') \to S(G)$.

PROOF. The method which is used in Algorithm 3.2 is functorial as well. For each newly introduced nonterminal of the form $[{}^{(1)}\alpha\pi]$ define $H([{}^{(1)}\alpha\pi])={}^{(1)}\alpha$. Furthermore, define for each newly introduced production $A \to a[{}^{(1)}\alpha\pi]\alpha^{(r-1)}\langle\pi\rangle$ in step (i), $H(A \to a[{}^{(1)}\alpha\pi]\alpha^{(r-1)}\langle\pi\rangle)=\bar{\pi}$. Moreover, for each production $[A\pi] \to \gamma\langle\varrho\rangle$, newly introduced in step (ii), $H([A\pi] \to \gamma\langle\varrho\rangle)=\bar{\varrho}$. For all other nonterminal symbols and productions, H is the identical functor. Now the proof can proceed in a straight forward way. ∎

REFERENCES

1. A. V. Aho and J. D. Ullman, *The theory of parsing, translation and compiling*, Vol. 1 and 2., Prentice-Hall, Englewood Cliffs, N.J. 1972 and 1973.
2. D. B. Benson, *Some preservation properties of normal form grammars*, Siam J. of Comput. 6 (1977), 381–402.
3. J. N. Gray and M. A. Harrison, *On the covering and reduction problems for context-free grammars*, J. Assoc. Comput. Mach. 19 (1972), 675–698.

DEPARTMENT OF MATHEMATICS
VRIJE UNIVERSITEIT
AMSTERDAM
THE NETHERLANDS