JOURNAL OF COMPUTER AND SYSTEM SCIENCES 15, 232–240 (1977)

Algebraic Properties of Derivation Words

GYÖRGY RÉVÉSZ

Computer and Automation Institute of Hungarian Academy of Sciences, 1502 Budapest, XI, Kende u. 13–17, Hungary

Received July 19, 1976; revised November 1, 1976

Derivation words introduced by Hart represent the canonical (leftmost) derivations in a phrase structure grammar and allow for a concrete realization to the categorical treatment of derivations due to Hotz. In this paper a simple self-embedding property of the domain and codomain functions of this realization will be established. This property can be used for simplifying most definitions and making the proofs much shorter.

1. INTRODUCTION

Derivation languages as defined by Hart [1] seem to be appropriate tools for algebraic representation of phrase structure derivations. They have a straightforward connection with the syntactical graphs of Loeckx [2] and are deterministic context-sensitive languages. But from the algebraic point of view it is more important that the composition and juxtaposition of derivations introduced by Hotz [3] can be given in terms of operations on derivation words. These operations are based on two functions defining the domain and the codomain of a derivation word.

In the present paper an interesting self-embedding property of these functions will be established which can be used for a more compact characterization of derivation languages and to render the proofs of the relevant theorems much shorter.

In Section 2 we recapitulate the original definitions given by Hart [1]. In Section 3 we present the alternative definitions and establish the self-embedding property of the domain and codomain functions. Also, we give the proofs of the basic theorems by using this property. Section 4 deals with the operations on derivation words where we slightly generalize the original definitions.

The reader may skip Section 2 if he is not interested in the connection with the original definitions. In either case it is assumed that the reader is familiar with the notions of phrase structure grammar and derivation in a phrase structure grammar.

In accordance with Hart [1] we slightly deviate from the standard notation of phrase structure grammars to the extent of assigning production names (or labels) to the elements of P. More precisely, let $G = (V_N, V_T, S, P)$ be a phrase structure grammar where V_N, V_T, S, and P are the set of nonterminal symbols, the set of terminal symbols, the

ISSN 0022-0000

initial symbol, and the set of production rules, respectively. Then each production rule in P will have a unique name, in symbols

$$p\colon u \to v \in P$$

where $u, v \in (V_N \cup V)_T^*$ and p is the name of the production rule. Hence the set of productions can be given in the form

$$P = \{p_1, p_2, \cdots, p_k\}$$

where each p_i stands for a specific production assigned to it. It is assumed that the production names are distinct symbols of the alphabet P such that $P \cap (V_N \cup V_T) = \varnothing$. For the sake of brevity $V_N \cup V_T$ will usually be denoted by V. The empty string will be denoted by λ and the length of a string w by $| w |$. Often we shall use mixed strings, that is, words over the alphabet $V \cup P$.

2. Old Definitions

If $p_i \in P$ with $p_i : a_1 a_2 \cdots a_m \to b_1 b_2 \cdots b_n$ for some $m \geqslant 1$, $n \geqslant 0$ (the a's and b's are in V), we say that the head stratification of p_i, $H(p_i)$, is m, and the tail stratification of p_i, $T(p_i)$, is n.

Definition 2.1. Let $G = (V_N, V_T, S, P)$ be a phrase structure grammar. The head sum of $x \in (V \cup P)^*$, $S_h(x)$ is defined as

(1) $S_h(\lambda) = 0$,

(2) $S_h(ax) = S_h(x) - 1$ if $a \in V$ and $S_h(x) > 0$,

(3) $S_h(ax) = S_h(x) + H(a)$ if $a \in P$, and

(4) $S_h(x)$ is undefined in all other cases.

The tail sum of $x \in (V \cup P)^*$, $S_t(x)$, is defined as

(1) $S_t(\lambda) = 0$,

(2) $S_t(xa) = S_t(x) - 1$ if $a \in V$ and $S_t(x) > 0$,

(3) $S_t(xa) = S_t(x) + T(a)$ if $a \in P$, and

(4) $S_t(x)$ is undefined in all other cases.

Note that if S_h and S_t are defined, they are nonnegative. If $S_h(x) = 0$, then $x = ax'p$ for some $a \in V$, $x' \in (V \cup P)^*$, and $p \in P$, or else $x = \lambda$. If $S_t(x) = 0$ and $x \neq \lambda$, then either $x = px'a$ as above or $x = x'p$ for some $p \in P$ with $T(p) = 0$ and x' with $S_t(x') = 0$.

Definition 2.2. Let $G = (V_N, V_T, S, P)$ be a phrase structure grammar. A string $w \in (V \cup P)^*$ is said to have domain $a_1 a_2 \cdots a_m$, written $\mathrm{dom}(w) = a_1 a_2 \cdots a_m$, if and only if w can be written as

$$w = a_1 x_1 a_2 x_2 \cdots a_m x_m,$$

with $a_1, a_2, \ldots, a_m \in V$ and $S_t(x_1) = S_t(x_2) = \cdots = S_t(x_m) = 0$. w has codomain $b_1 \cdots b_n$, written $\mathrm{cod}(w) = b_1 b_2 \cdots b_n$, if w can be written as

$$w = y_0 b_1 y_2 b_2 \cdots y_{n-1} b_n$$

with $b_1, b_2, \ldots, b_n \in V$ and $S_h(y_0) = S_h(y_1) = \cdots = S_h(y_{n-1}) = 0$.

The definition of $\mathrm{dom}(w)$ and $\mathrm{cod}(w)$ is precise, for a word w has at most one such factorization. Also, if $\mathrm{dom}(w_1) = x_1$ and $\mathrm{dom}(w_2) = x_2$, then $\mathrm{dom}(w_1 w_2) = x_1 x_2$. Likewise, if $\mathrm{cod}(w_1) = y_1$ and $\mathrm{cod}(w_2) = y_2$, then $\mathrm{cod}(w_1 w_2) = y_1 y_2$. Thus, dom and cod are homomorphisms where they are defined.

DEFINITION 2.3. Let $G = (V_N, V_T, S, P)$ be a phrase structure grammar. A derivation word with domain $x \in V^*$ and codomain $y \in V^*$ is defined recursively as follows.

(1) If $w = a_1 a_2 \cdots a_m \in V^*$ $(m \geqslant 0)$, then w is a derivation word with $\mathrm{dom}(w) = \mathrm{cod}(w) = w$.

(2) If $w \in (V \cup P)^*$ is a derivation word with $w = w_1 w_2 w_3$, $\mathrm{cod}(w_1) = u$, $\mathrm{cod}(w_2) = a_1 a_2 \cdots a_m$, $\mathrm{cod}(w_3) = v$, and $\mathrm{dom}(w) = x$, and if p: $a_1 a_2 \cdots a_m \to b_1 b_2 \cdots b_n \in P$, then

$$z = w_1 w_2 p b_1 b_2 \cdots b_n w_3$$

is a derivation word with $\mathrm{dom}(z) = x$ and $\mathrm{cod}(z) = u b_1 \cdots b_n v$.

(3) Nothing else is a derivation word unless its being so follows from (1) and (2).

This definition is precise, and it is easy to check (using recursion) that the derivation words have the indicated domain and codomain as defined by Definition 2.2.

DEFINITION 2.4. Let $G = (V_N, V_T, S, P)$ be a phrase structure grammar and w_1, w_2 derivation words of G. Then the juxtaposition of w_1 with w_2 is the concatenation of w_1 and w_2, i.e., the product $w_1 w_2$ in the free monoid $(V \cup P)^*$.

DEFINITION 2.5. Let $G = (V_N, V_T, S, P)$ be a phrase structure grammar and w_1, w_2 derivation words with $\mathrm{dom}(w_2) = \mathrm{cod}(w_1)$. If $\mathrm{dom}(w_2) = b_1 \cdots b_n$ $(b_i \in V)$ with

$$w_1 = y_0 b_1 y_1 \cdots y_{n-1} b_n \qquad (S_h(y_i) = 0, i = 0, 1, \ldots, n-1)$$

and

$$w_2 = b_1 x_1 \cdots b_n x_n \qquad (S_t(x_i) = 0, i = 1, \ldots, n)$$

then the composition of w_2 with w_1, written $w_2 \circ w_1$, is

$$w_2 \circ w_1 = y_0 b_1 x_1 y_1 b_2 x_2 y_2 \cdots b_n x_n .$$

In the next section we shall see that the functions $H(x)$, $T(x)$, $S_h(x)$, and $S_t(x)$ are not needed here at all.

3. New Definitions and the Self-Embedding Property

Definition 3.1. Let $G = (V_N, V_T, S, P)$ be a phrase structure grammar. The domain of a string $w \in (V \cup P)^*$, $\mathrm{dom}(w)$, is defined recursively as follows.

(1) If $w \in V^*$, then $\mathrm{dom}(w) = w$,

(2) if $\mathrm{dom}(xy)$ is defined and $p: u \to v \in P$, then $\mathrm{dom}(xpvy) = \mathrm{dom}(xy)$.

The codomain of a string $w \in (V \cup P)^*$, $\mathrm{cod}(w)$, is defined recursively as follows.

(1) If $w \in V^*$, then $\mathrm{cod}(w) = w$,

(2) if $\mathrm{cod}(xy)$ is defined and $p: u \to v \in P$, then $\mathrm{cod}(xupy) = \mathrm{cod}(xy)$.

Note that $\mathrm{dom}(w)$ and $\mathrm{cod}(w)$ are in V^* whenever they are defined. It is easy to show by induction on the number of production names occurring in w that these definitions are precise. For, if w has two different decompositions $x_1 p_1 v_1 y_1$ and $x_2 p_2 v_2 y_2$, then the substrings $p_1 v_1$ and $p_2 v_2$ may not overlap, that is, either $w = xp_1 v_1 z p_2 v_2 y$ or $w = xp_2 v_2 z p_1 v_1 y$ for some $x, z, y \in (V \cup P)^*$. Therefore

$$\mathrm{dom}(w) = \mathrm{dom}(x_1 y_1) = \mathrm{dom}(x_2 y_2) = \mathrm{dom}(xzy)$$

which is precise by the induction hypothesis. The argument is quite similar for $\mathrm{cod}(w)$.

These functions have the self-embedding property expressed by the next lemma.

Lemma 3.1. *If* $w_1, w_2, w_3 \in (V \cup P)^*$, *then*

$$\mathrm{dom}(w_1 w_2 w_3) = \mathrm{dom}(w_1\, \mathrm{dom}(w_2) w_3),$$

and

$$\mathrm{cod}(w_1 w_2 w_3) = \mathrm{cod}(w_1\, \mathrm{cod}(w_2) w_3)$$

whenever the right-hand side of the corresponding equation is defined.

Proof. This will be shown by induction on the number of production names occurring in w_2. Basis: If $w_2 \in V^*$, then the assertion is trivial. Induction step: Suppose that w_2 contains n production names ($n \geqslant 1$) and $\mathrm{dom}(w_1\, \mathrm{dom}(w_2) w_3)$ is defined. This implies that $\mathrm{dom}(w_2)$ is also defined, hence $w_2 = xpvy$ for some $x, y \in (V \cup P)^*$ and $p: u \to v \in P$ with $\mathrm{dom}(w_2) = \mathrm{dom}(xpvy) = \mathrm{dom}(xy)$. Then

$$\mathrm{dom}(w_1 w_2 w_3) = \mathrm{dom}(w_1 xpvyw_3) = \mathrm{dom}(w_1 xyw_3)$$

by definition and

$$\mathrm{dom}(w_1 xyw_3) = \mathrm{dom}(w_1\, \mathrm{dom}(xy) w_3)$$

by the induction hypothesis which completes the proof.

The second equation of the lemma can be shown by a similar argument.

Definition 3.2. Let $G = (V_N, V_T, S, P)$ be a phrase structure grammar. The derivation language of G, $D(G)$, is defined recursively as follows.

(1) If $w \in V^*$ then $w \in D(G)$,

(2) if $xy \in D(G)$ with $\mathrm{cod}(x)$, $\mathrm{cod}(y)$ being defined and $p: u \to v \in P$ such that $\mathrm{cod}(x) = zu$ for some $z \in V^*$, then $xpvy \in D(G)$.

(3) Nothing else belongs to $D(G)$.

The elements of $D(G)$ are called derivation words of G.

EXAMPLE 3.1. Let $G = (V_N, V_T, S, P)$ be a phrase structure grammar with $V_N = \{S, A, B\}$, $V_T = \{a, b\}$, and $P = \{p_1 : S \to SAb, p_2 : bAb \to aBa, p_3 : Aa \to AB, p_4 : BB \to aBb\}$. Then the derivation word

$$w = Sp_1Sp_1SAbAbp_2ap_3ABBa$$

represents the derivation given by the syntactical graph in Fig. 1.

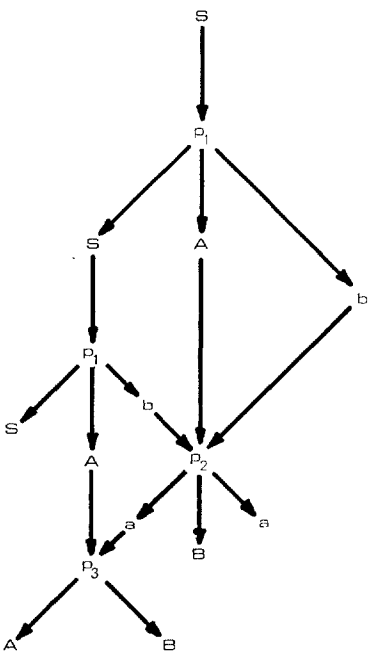

FIG. 1. Syntactical graph of Example 3.1.

To show the domain and the codomain of this derivation word, we parenthesize its substrings in both ways.

$$\text{domain: } S(p_1S(p_1SAb)\,Ab)(p_2a(p_3AB)\,Ba),$$

$$\text{codomain: } (Sp_1)(Sp_1)\,S(A(bAbp_2)\,ap_3)\,ABBa.$$

As can be seen in this example the domain and the codomain correspond to the starting and the ending nodes, respectively. This is true in general for derivation words as it is shown below. Actually, derivation words can be considered as morphisms of a category whose objects are the strings in V^*. First we show the following lemma.

LEMMA 3.2. *If* $w \in D(G)$ *then both* $\mathrm{dom}(w)$ *and* $\mathrm{cod}(w)$ *are defined.*

Proof. The proof is given by induction on the number of production names occurring in w. Basis: For $w \in V^*$ the assertion is trivial. Induction step: Assume that the lemma is true for derivation words containing at most n production names and let w contain $n + 1$ of them. According to the definition of $D(G)$, w must be of the form $xpvy$ where xy contains n production names. Hence,

$$\mathrm{dom}(w) = \mathrm{dom}(xpvy) = \mathrm{dom}(xy)$$

which is defined by the induction hypothesis. On the other hand,

$$\mathrm{cod}(w) = \mathrm{cod}(\mathrm{cod}(x)\, pvy) = \mathrm{cod}(zupvy)$$

for some $z \in V^*$ with $p: u \to v \in P$. Then

$$\mathrm{cod}(zupvy) = \mathrm{cod}(z\, \mathrm{cod}(up)\, vy) = \mathrm{cod}(zvy)$$

where zvy contains at most n production names and this completes the proof.

Note that $\mathrm{dom}(w)$ or $\mathrm{cod}(w)$ may be defined even if $w \notin D(G)$. Now we can show the basic theorem which justifies the name of derivation words.

THEOREM 3.1. *If* $w \in D(G)$ *then* $\mathrm{dom}(w) \overset{*}{\Rightarrow}_G \mathrm{cod}(w)$.

Proof. Again this is shown by induction on the number of production names in w. Basis: For $w \in V^*$ the assertion is trivial. Induction step: Let w contain n production names ($n \geqslant 1$). Then $w = xpvy$ for some $x, y \in (V \cup P)^*$, $p \in P$, and $v \in V^*$ as required by the definition of $D(G)$. Hence

$$\mathrm{dom}(w) = \mathrm{dom}(xpvy) = \mathrm{dom}(xy)$$

and

$$\mathrm{dom}(xy) \overset{*}{\underset{G}{\Rightarrow}} \mathrm{cod}(xy)$$

by the induction hypothesis. Further, by Lemma 3.1 we have

$$\begin{aligned}\mathrm{cod}(xy) &= \mathrm{cod}(\mathrm{cod}(x)\, y) = \mathrm{cod}(\mathrm{cod}(x)\, \mathrm{cod}(y)) = \mathrm{cod}(x)\, \mathrm{cod}(y) \\ &= zu\, \mathrm{cod}(y) \overset{*}{\underset{G}{\Rightarrow}} zv\, \mathrm{cod}(y) = \mathrm{cod}(zv\, \mathrm{cod}(y)) = \mathrm{cod}(zvy) \\ &= \mathrm{cod}(z\, \mathrm{cod}(up) vy) = \mathrm{cod}(zupvy) = \mathrm{cod}(\mathrm{cod}(x)\, pvy) = \mathrm{cod}(w)\end{aligned}$$

and this completes the proof.

COROLLARY. *The language generated by the grammar G can be given as*

$$L(G) = \{\mathrm{cod}(w) \mid w \in D(G), \mathrm{dom}(w) = S, \mathrm{cod}(w) \in V_T^*\}.$$

We conclude this section by an important characterization theorem to the effect that the conversion of Lemma 3.2 is also true.

THEOREM 3.2. *$w \in D(G)$ if and only if both* $\mathrm{dom}(w)$ *and* $\mathrm{cod}(w)$ *are defined.*

Proof. We have to show only the if part which will be done by induction on the number of production names occurring in w. Basis: For $w \in V^*$ the assertion is trivial. Induction step: Let w contain n production names ($n \geqslant 1$) and both $\mathrm{dom}(w)$ and $\mathrm{cod}(w)$ be defined. Then there must occur some p in w such that $w = xpvy$ and $\mathrm{dom}(w) = \mathrm{dom}(xpvy) = \mathrm{dom}(xy)$. If there is more than one such p then we choose the rightmost one. In this case y must be in V^* and thus, $\mathrm{cod}(y)$ must be defined. (Namely, the computation of $\mathrm{dom}(w)$ is performed by successive cancellation of production names occurring in w together with a possibly empty substring from the right context of the given p. Hence, the rightmost p in w may be canceled first.)

For this particular p we have $\mathrm{cod}(w) = \mathrm{cod}(xpvy) = \mathrm{cod}(\mathrm{cod}(xp)\, vy)$ provided that $\mathrm{cod}(xp)$ is defined. But, this must be the case, otherwise in the course of the computation of $\mathrm{cod}(w)$ we could not cancel this p at all. Further, $\mathrm{cod}(xp) = \mathrm{cod}(\mathrm{cod}(x)\, p)$ therefore, $\mathrm{cod}(x)$ must be also defined with $\mathrm{cod}(x) = zu$, $z \in V^*$, and $p: u \to v \in P$. This means that $\mathrm{cod}(xy)$ is also defined for $\mathrm{cod}(xy) = \mathrm{cod}(\mathrm{cod}(x)\, \mathrm{cod}(y)) = \mathrm{cod}(x)\, \mathrm{cod}(y)$.

Now, $xy \in D(G)$ by the induction hypothesis and thus, $w = xpvy \in D(G)$ by the definition of $D(G)$.

4. OPERATIONS ON DERIVATION WORDS

For the juxtaposition of derivation words we will use the same definition as given in Section 2. (See Definition 2.4.)

THEOREM 4.1. *If w_1 and w_2 are in $D(G)$ then the juxtaposition of w_1 with w_2 is also in $D(G)$.*

Proof. By Lemma 3.2 $\mathrm{dom}(w_1)$, $\mathrm{dom}(w_2)$, $\mathrm{cod}(w_1)$, and $\mathrm{cod}(w_2)$ are all defined. Therefore

$$\mathrm{dom}(w_1 w_2) = \mathrm{dom}(\mathrm{dom}(w_1)\, \mathrm{dom}(w_2)) = \mathrm{dom}(w_1)\, \mathrm{dom}(w_2),$$

and

$$\mathrm{cod}(w_1 w_2) = \mathrm{cod}(\mathrm{cod}(w_1)\, \mathrm{cod}(w_2)) = \mathrm{cod}(w_1)\, \mathrm{cod}(w_2).$$

Hence by Theorem 3.2 $w_1 w_2 \in D(G)$ which completes the proof.

DEFINITION 4.1. Let $w_1, w_2 \in (V \cup P)^*$ with $\mathrm{cod}(w_1) = \mathrm{dom}(w_2)$ and $\mathrm{cod}(W_1) \neq \lambda$ if $W_1 \neq \lambda$. Then the composition of w_2 with w_1, written $w_2 \circ w_1$, is defined recursively as follows.

(1) If $w_1 = \lambda$ then $w_2 \circ w_1 = w_2$,

(2) if for some $v \in V^+$ and $u_1, u_2, z_1, z_2 \in (V \cup P)^*$ $w_1 = u_1 v z_1$ with $\operatorname{cod}(w_1) = v \operatorname{cod}(z_1)$ and $w_2 = u_2 v z_2$ with $\operatorname{dom}(u_2) = \lambda$, then $w_2 \circ w_1 = u_2 u_1 v (z_2 \circ z_1)$.

Note that $\operatorname{cod}(u_1) = \lambda$ always holds here. Namely, we have

$$\operatorname{cod}(w_1) = \operatorname{cod}(u_1 v z_1) = \operatorname{cod}(u_1 v \operatorname{cod}(z_1)) = \operatorname{cod}(u_1 \operatorname{cod}(w_1)) = \operatorname{cod}(u_1) \operatorname{cod}(w_1)$$

provided that $\operatorname{cod}(u_1)$ is defined. But this must be the case since $\operatorname{cod}(w_1)$ is defined and any truncation of w_1 from the right preserves the computability of the codomain function. (Symmetrically, left truncation preserves the computability of the domain function.)

It is easy to see that for derivation words the above definition is equivalent to Definition 2.5. Actually, by a repeated application of recursion scheme (2) we obtain the decompositions

$$\begin{aligned} w_1 &= u_{1,1} v_1 u_{1,2} v_2 \cdots u_{1,k} v_k z_{1,k}, \\ w_2 &= u_{2,1} v_1 u_{2,2} v_2 \cdots u_{2,k} v_k z_{2,k}, \\ w_2 \circ w_1 &= u_{2,1} u_{1,1} v_1 \cdots u_{2,k} u_{1,k} v_k (z_{2,k} \circ z_{1,k}) \end{aligned}$$

where $\operatorname{cod}(u_{1,i}) = \operatorname{dom}(u_{2,i}) = \lambda$ for $i = 1, \ldots, k$ and $\operatorname{dom}(z_{2,k}) = z_{1,k} = \lambda$. Here we may choose each v_i to consist of only one letter, that is, we may insert redundant empty $u_{1,i}$ and $u_{2,i}$ strings between the letters of the v's. Now, if w_2 is a derivation word then $u_{2,1}$ must be λ and thus, we obtain the same result as with Definition 2.5.

On the other hand, w_1 and w_2 need not be derivation words. If, for instance, $p\colon AB \to CD \in G$ and w_1, w_2 are derivation words with $\operatorname{cod}(w_1) = \operatorname{dom}(w_2)$ then $pCDw_2 \circ w_1$ is also defined though $pCDw_2$ is not a derivation word since $\operatorname{cod}(pCDw_2)$ is undefined.

The important features of the composition follow immediately from our definition as we will show below.

THEOREM 4.2 (Composition Theorem). *If w_1 and w_2 are derivation words with $\operatorname{cod}(w_1) = \operatorname{dom}(w_2)$ then $\operatorname{dom}(w_2 \circ w_1) = \operatorname{dom}(w_1)$ and $\operatorname{cod}(w_2 \circ w_1) = \operatorname{cod}(w_2)$.*

Proof. For $w_1 = \lambda$ the assertion is trivial. Otherwise recursion scheme (2) must be applied finitely many times. Thus, we have

$$\begin{aligned} \operatorname{dom}(w_2 \circ w_1) &= \operatorname{dom}(u_{2,1} u_{1,1} v_1 \cdots u_{2,k} u_{1,k} v_k (z_{2,k} \circ z_{1,k})) \\ &= \operatorname{dom}(\operatorname{dom}(u_{2,1})\, u_{1,1} v_1 \cdots \operatorname{dom}(u_{2,k})\, u_{1,k} v_k \operatorname{dom}(z_{2,k})) \\ &= \operatorname{dom}(u_{1,1} v_1 \cdots u_{1,k} v_k) = \operatorname{dom}(w_1). \end{aligned}$$

Similarly,

$$\begin{aligned} \operatorname{cod}(w_2 \circ w_1) &= \operatorname{cod}(u_{2,1} \operatorname{cod}(u_{1,1})\, v_1 \cdots u_{2,k} \operatorname{cod}(u_{1,k})\, v_k z_{2,k}) \\ &= \operatorname{cod}(u_{2,1} v_1 \cdots u_{2,k} v_k z_{2,k}) = \operatorname{cod}(w_2). \end{aligned}$$

THEOREM 4.3 (Juxtaposition Theorem). $x_2 y_2 \circ x_1 y_1 = (x_2 \circ x_1)(y_2 \circ y_1)$ *whenever the latter is defined.*

Proof. If $x_1 = \lambda$ then the assertion is trivial. Let $x_1 = u_1 v z_1$ and $x_2 = u_2 v z_2$ according to (2) of Definition 4.1. Then

$$x_2 y_2 \circ x_1 y_1 = u_2 u_1 v(z_2 y_2 \circ z_1 y_1) = u_2 u_1 v(z_2 \circ z_1)(y_2 \circ y_1) = (x_2 \circ x_1)(y_2 \circ y_1)$$

by definition and by the induction hypothesis.

Clearly the operations of composition and juxtaposition can be extended to more than two operands and analoguous theorems will hold for this case.

Finally, let us consider the same example as given by Hart [1]. (See Example 3.1 in his paper.) The rules of the grammar are

$$p_1 : AB \to CAB, \qquad p_2 : C \to BA, \qquad p_3 : BAA \to CBA,$$

and

$$w_1 = ABp_1CABp_1Cp_2BAAB,$$
$$w_2 = Cp_2BABp_1CABAAp_3CBAB,$$

are derivation words with $\mathrm{cod}(w_1) = \mathrm{dom}(w_2) = CBAAB$. Now, we will have the decompositions

$$w_1 = u_1 v z_1 \text{ , where } u_1 = ABp_1 \text{ , } v = C, \; z_1 = ABp_1Cp_2BAAB,$$

and

$$w_2 = u_2 v z_2 \text{ , } \qquad \text{where } \; u_2 = \lambda, \; v = C, \; z_2 = p_2BABp_1CABAAp_3CBAB.$$

It can be observed that z_2 is not a derivation word here, but $z_2 \circ z_1$ is still defined. The repeated application of recursion scheme (2) will yield

$$w_2 \circ w_1 = ABp_1Cp_2BAABp_1Cp_2Bp_1CABAAp_3CBAB$$

which is again a derivation word with $\mathrm{dom}(w_2 \circ w_1) = \mathrm{dom}(w_1)$ and $\mathrm{cod}(w_2 \circ w_1) = \mathrm{cod}(w_2)$. This means, of course, that

$$AB = \mathrm{dom}(w_1) \overset{*}{\underset{G}{\Rightarrow}} \mathrm{cod}(w_2) = BCACBAB.$$

References

1. J. M. Hart, Derivation languages and syntactical categories, *Inform. Contr.* **28** (1975), 204–220.
2. J. Loeckx, The parsing of general phrase structure grammars, *Inform. Contr.* **16** (1970), 443–464.
3. G. Hotz, Eindeutigkeit und Mehrdeutigkeit formaler Sprachen, *Elektron. Informationsverarbeit. Kybernetic* **2** (1966), 235–246.