ANNALES ACADEMIAE SCIENTIARUM FENNICAE

Series A

I. MATHEMATICA

479

ON SOME FAMILIES OF FORMAL LANGUAGES OBTAINED BY REGULATED DERIVATIONS

BY

ARTO SALOMAA

HELSINKI 1970
SUOMALAINEN TIEDEAKATEMIA

doi:10.5186/aasfm.1971.479

Communicated 11 September 1970

KESKUSKIRJAPAINO
HELSINKI 1970

Introduction

The customary hierarchy of formal language families $\mathcal{L}(i)$ consisting of languages of type i, $i = 0, 1, 2, 3$, is obtained by imposing restrictions on the form of productions. A natural generalization is to impose restrictions also on the use of productions as done in defining matrix grammars, [1], programmed grammars, [6], ordered grammars, [4], probabilistic grammars, [10], or periodically time-variant grammars, [9]. A common idea is that not every derivation leading from the initial symbol to a terminal word is acceptable, but rather there is a control device which lets through acceptable derivations only.

A convenient uniform way of describing restrictions on the use of productions is to introduce a control language C for a grammar G, [3], [7], [12], [14]. The notion of a control set in [5] is essentially different. The control language C is a set of finite strings of productions of G, referred to as control words. The language generated by G with control language C is the subset of the language generated by G, consisting of words which possess at least one derivation whose string of productions belongs to C. Thereby, two interpretations of control words are possible. In the narrow or non-checking interpretation, each letter of a control word has to be applied. In the broad or checking interpretation, one may specify some productions such that, whenever they occur in a control word and are not applicable at the corresponding step of the derivation, then we may move to the next production in the control word.

In this paper, we study language families $\mathcal{L}(i, j, 0)$ and $\mathcal{L}(i, j, 1)$ generated by type i grammars with type j control language, where $0 \leqq i, j \leqq 3$. The numbers 0 and 1 in the last argument place refer to non-checking and checking interpretation, respectively. Since the case $i = 2$ is separated into two subcases, we obtain altogether 40 language families. However, most of them coincide with some of the families $\mathcal{L}(i)$.

Definitions and a survey of results are given in Section 1. Sections 2 and 3 deal with families where the core productions are context-sensitive, i.e., $i = 1$. It is shown that every language in the family $\mathcal{L}(1, 3, 1)$ is context-sensitive. This is a generalization of the results in [3] and [12] concerning the family $\mathcal{L}(1, 3, 0)$, as well as the result concerning context-sensitive programmed grammars in [6]. It is then shown that

(1) $$\mathcal{L}(1\,,\,2\,,\,1) = \mathcal{L}(1\,,\,2\,,\,0)$$

and that every language in this family is recursive. It remains an open problem whether or not the family occurring in (1) contains properly the family of context-sensitive languages.

In Section 4, we investigate the family

(2) $$\mathcal{L}(2\,,\,3\,,\,0)$$

obtained using context-free core productions (including productions with the empty word on the right side) with a regular control language under non-checking interpretation. An operation characteristic for this family is introduced and its properties studied. A result concerning a subfamily of (2) is obtained. It is known, [6], [7], that if in (2) 0 is replaced by 1, the resulting family equals $\mathcal{L}(0)$. However, the size of the family (2) remains an open problem.

1. The families $\mathcal{L}(i\,,j\,,k)$. Let $G = (I_N\,,I_T\,,X_0\,,F)$ be a phrase structure grammar, where I_N is the set of nonterminals, I_T the set of terminals, X_0 the initial symbol and F the set of productions. Derivations according to G, the language $L(G)$ generated by G, as well as type i $(i = 0\,,\,1\,,\,2\,,\,3)$ grammars in the hierarchy obtained by imposing restrictions on F, are defined in the usual fashion, cf. [11, pp. 164—169]. The family of type i languages $(i = 0\,,\,1\,,\,2\,,\,3)$ is denoted by $\mathcal{L}(i)$.

Let

(3) $$\{f_1\,,\,\ldots\,,f_u\}$$

be a set of distinct labels for the productions in F and assume that F_1 is a subset of (3). Let

(4) $$X_0 = P_0 \underset{f_{j(0)}}{\Rightarrow} P_1 \underset{f_{j(1)}}{\Rightarrow} \ldots \underset{f_{j(r-1)}}{\Rightarrow} P_r\,,\ r \geqq 1\,,$$

be a derivation according to G, where for each i, $0 \leqq i < r$, the production labeled by $f_{j(i)}$ is $R_i \to S_i$ and either

(i) there exist Q_1 and Q_2 such that $P_i = Q_1R_iQ_2$ and $P_{i+1} = Q_1S_iQ_2$, or else

(ii) R_i is not a subword of P_i, $f_{j(i)} \in F_1$ and $P_i = P_{i+1}$.

Then the word

$$f_{j(0)}f_{j(1)}\ldots f_{j(r-1)}$$

over the alphabet (3) is termed a *control word* of the derivation (4).

Thus, a control word of a derivation indicates which productions have been applied in the derivation. Thereby, »applying» a production f either

means actual rewriting according to f (cf. point (i) above), or checking that such a rewriting is not possible and that $f \in F_1$ (cf. point (ii)). For productions in the set $F - F_1$, only the alternative (i) is possible.

Let C be a language over the alphabet (3). Then

$$L_C(G, F_1) \tag{5}$$

is defined to be the subset of $L(G)$ consisting of words which possess at least one derivation whose control word is in C. (5) is referred to as the language generated by the pair (G, F_1) with the *control language* C. If G is a type i grammar and C a type j language, we say that (5) is a type $(i, j, 1)$ language. If, furthermore, the set F_1 is the empty set $\emptyset$ (i.e., the application of a production f always means actual rewriting according to f), then we say that (5) is of type $(i, j, 0)$. Thereby, j ranges through the numbers $0, 1, 2, 3$, and i ranges through the numbers $0, 1, 2, 2 - \lambda, 3$. The difference between the types 2 and $2 - \lambda$ is that productions of the form $X \to \lambda$, where λ is the empty word, are allowed in type $2 - \lambda$ only if X is the initial symbol and does not occur on the right side of any production, whereas all context-free productions are allowed in type 2. (It is well known that the generative capacity of grammars of types 2 and $2 - \lambda$ is the same. However, it is not the same for grammars with a control language.)

If $F_1 = \emptyset$, we are dealing with the *narrow* or *non-checking* interpretation of control words. Otherwise, we speak of the *broad* or *checking* interpretation. The family of languages of the type (i, j, k) is denoted by

$$\mathfrak{L}(i, j, k). \tag{6}$$

Thus, by definition, there are 40 language families of the form (6). However, many of them coincide. By definition (cf. also Remark 1 in [7]), the following inclusions are obvious, for all i, j, k, i_1, j_1:

$$\mathfrak{L}(i) \subseteq \mathfrak{L}(i, j, k), \tag{7}$$

$$\mathfrak{L}(j) \subseteq \mathfrak{L}(i, j, k), \tag{8}$$

$$\mathfrak{L}(i, j, 0) \subseteq \mathfrak{L}(i, j, 1), \tag{9}$$

$$\mathfrak{L}(i, j, k) \subseteq \mathfrak{L}(i, j_1, k) \text{ if } j \geqq j_1, \tag{10}$$

$$\mathfrak{L}(i, j, k) \subseteq \mathfrak{L}(i_1, j, k) \text{ if } i \geqq i_1 \text{ and not both } i = 2 \text{ and } i_1 = 1. \tag{11}$$

In (11) it is understood that $3 \geqq 2 - \lambda \geqq 2 \geqq 1 \geqq 0$. The additional assumption concerning i and i_1 is necessary because, as will be seen below,

$$\mathfrak{L}(2, 3, 1) = \mathfrak{L}(0) \text{ and } \mathfrak{L}(1, 3, 1) = \mathfrak{L}(1).$$

For the definition of *programmed grammars*, *matrix grammars* and *periodically time-variant grammars* the reader is referred to [6], [1] and [9], respectively, or to [7]. The family of languages generated by programmed grammars with type i core productions ($i = 0, 1, 2, 2 - \lambda, 3$) is denoted by $\mathcal{P}(i, 1)$. If, in addition, all failure fields are empty, the corresponding family is denoted by $\mathcal{P}(i, 0)$. The notations $\mathcal{M}(i, k)$ and $\mathcal{T}(i, k)$ are used for the families generated by matrix grammars and periodically time-variant grammars. Thereby, as in the definition of the family (6), $k = 1$ indicates the broad sense of the application of productions, whereas $k = 0$ indicates the narrow sense.

We now give a summary of the known inclusions between the families introduced. It is fairly easy to prove (cf. [7]) that, for all i and k,

$$\mathcal{T}(i, k) \subseteq \mathcal{M}(i, k) \subseteq \mathcal{P}(i, k) \subseteq \mathcal{L}(i, 3, k) . \tag{12}$$

It is established in [9] that

$$\mathcal{T}(i, k) = \mathcal{M}(i, k) = \mathcal{P}(i, k) = \mathcal{L}(i, 3, k) , \tag{13}$$

for $i = 2, 2 - \lambda$ and $k = 0, 1$. (In fact, we are going to see that the equations (13) hold true for all values of i and k.) The following relations are established in [6]:

$$\mathcal{P}(2, 1) = \mathcal{L}(0) , \tag{14}$$

$$\mathcal{L}(2) \subset \mathcal{P}(2 - \lambda, k) \subset \mathcal{L}(1) , \quad k = 0, 1 , \tag{15}$$

$$\mathcal{P}(1, 1) = \mathcal{L}(1) , \tag{16}$$

where $\subset$ denotes proper inclusion.

Remark 1. The programmed grammars defined in [6] operate under so-called »leftmost interpretation», i.e., always the leftmost occurrence of a string is rewritten. However, (14) — (16) hold true also if an arbitrary occurrence of a string may be rewritten. This is the »free interpretation» of [6]. In this paper, as well as in [7], the programmed grammars are assumed to operate under free interpretation.

It is obvious by Church's Thesis that the family (6) equals the family $\mathcal{L}(0)$ whenever $i = 0$ or $j = 0$. It is shown in [14] and, independently, in [3] that

$$\mathcal{L}(3, 1, 0) = \mathcal{L}(0) . \tag{17}$$

Remark 2. In the proof of (17), it is essential that the type 3 core productions include productions of the form $X \to Y$, where X and Y are nonterminals. If such productions are excluded and the resulting type is denoted by 3_2, we obtain

$$\mathcal{L}(3_2 , 1 , 0) = \mathcal{L}(1) .$$

Such a distinction between types 3 and 3_2 does not lead into different language families if the control language is of type 0, 2 or 3.

By (9) — (11), (13), (14) and (17), the family (6) equals the family $\mathcal{L}(0)$ for each of the following 26 triples (i , j , k):

$(0 , 0 , 0)$, $(0 , 0 , 1)$, $(0 , 1 , 0)$, $(0 , 1 , 1)$, $(0 , 2 , 0)$, $(0 , 2 , 1)$,
$(0 , 3 , 0)$, $(0 , 3 , 1)$; $(1 , 0 , 0)$, $(1 , 0 , 1)$, $(2 , 0 , 0)$, $(2 , 0 , 1)$,
$(2 - \lambda , 0 , 0)$, $(2 - \lambda , 0 , 1)$, $(3 , 0 , 0)$, $(3 , 0 , 1)$; $(3 , 1 , 0)$, $(3 , 1 , 1)$,
$(2 - \lambda , 1 , 0)$, $(2 - \lambda , 1 , 1)$, $(2 , 1 , 0)$, $(2 , 1 , 1)$, $(1 , 1 , 0)$, $(1 , 1 , 1)$;
$(2 , 3 , 1)$, $(2 , 2 , 1)$.

In the next section, it will be shown that

$$\mathcal{L}(1 , 3 , 0) = \mathcal{L}(1 , 3 , 1) = \mathcal{L}(1) . \tag{18}$$

(This result concerning $\mathcal{L}(1 , 3 , 0)$ was established also in [3] and [14].) According to [7],

$$\mathcal{L}(3 , 3 , 0) = \mathcal{L}(3 , 3 , 1) = \mathcal{L}(3) , \tag{19}$$

$$\mathcal{L}(3 , 2 , 0) = \mathcal{L}(3 , 2 , 1) = \mathcal{L}(2) . \tag{20}$$

By (18) — (20) and the list of 26 families, only the following 8 families may be different from the families $\mathcal{L}(i)$:

$$\mathcal{L}(2 - \lambda , 3 , 0) , \mathcal{L}(2 - \lambda , 3 , 1) , \mathcal{L}(2 , 3 , 0) , \mathcal{L}(2 , 2 , 0) , \tag{21}$$
$$\mathcal{L}(2 - \lambda , 2 , 0) , \mathcal{L}(2 - \lambda , 2 , 1) , \mathcal{L}(1 , 2 , 0) , \mathcal{L}(1 , 2 , 1) .$$

It follows by (15) and (13) that both of the families

$$\mathcal{L}(2 - \lambda , 3 , 0) \text{ and } \mathcal{L}(2 - \lambda , 3 , 1) \tag{22}$$

properly contain the family of context-free languages and are properly contained in the family of context-sensitive languages. It will be shown that

$$\mathcal{L}(1 , 2 , 0) = \mathcal{L}(1 , 2 , 1)$$

(this equation is denoted by (1) in the introduction).

The notion of an *abstract family of languages*, abbreviated AFL, is defined as in [5]. Using the results of [7], the following theorem can be obtained. (In fact, the proof of the assertion concerning the closure of $\mathcal{L}(2 - \lambda , 3 , 1)$ under restricted homomorphism requires a slight modification of the methods of [7]. We will present it in detail in a forthcoming paper about scattered context languages.)

Theorem 1. *Each of the families* (21) *is closed under each of the following operations: union, catenation, restricted homomorphism, intersection with regular languages and* λ*-free regular substitution. Furthermore, the families*

$$\mathscr{L}(2 - \lambda , 3 , 1) , \quad \mathscr{L}(2 - \lambda , 2 , 1) , \quad \mathscr{L}(1 , 2 , 0) \tag{23}$$

are closed under catenation closure. Consequently, each of the families (23) *is an AFL.*

It is an open problem whether or not the families (21) other than those in (23) are closed under catenation closure and, consequently, whether or not they are AFL's. Some of the families (21) are closed under some additional operations like intersection, substitution and arbitrary homomorphism.

2. The family $\mathscr{L}(1 , 3 , 1)$. By (7) and (9),

$$\mathscr{L}(1) \subseteq \mathscr{L}(1 , 3 , 0) \subseteq \mathscr{L}(1 , 3 , 1) . \tag{24}$$

We will now establish the equations (18). Thus, regular control languages do not increase the generative capacity of context-sensitive grammars, not even under the broad interpretation of control words.

A grammar with the end marker $\#$ and derivations of the form

$$\# X_0 \# \Rightarrow \# P_1 \# \Rightarrow \ldots \Rightarrow \# P_r \#$$

is defined in the usual fashion, [11, p. 202]. The notion of a control language is extended to concern grammars with an end marker.

Lemma. *Any language of the form* $L_C(G , \emptyset)$, *where* C *is regular (context-free) and* G *a context-sensitive grammar with an end marker, belongs to the family* $\mathscr{L}(1 , 3 , 0)$ $(\mathscr{L}(1 , 2 , 0))$.

The proof of the Lemma, being similar to the corresponding proof concerning context-sensitive grammars, [11, pp. 202—203], is omitted. In fact, the only additional information needed is the result that the family of regular (context-free) languages is closed under regular substitution.

We will now prove that

$$\mathscr{L}(1 , 3 , 0) = \mathscr{L}(1 , 3 , 1) . \tag{25}$$

By (24) and the Lemma, it suffices to prove that any language L of the form

$$L = L_C(G , F_1) , \quad F_1 \neq \emptyset , \tag{26}$$

where

$$G = (I_N , I_T , X_0 , F) \tag{27}$$

is a context-sensitive grammar and C is regular, satisfies the equation

$$L = L_{C_1}(G_1, \emptyset), \tag{28}$$

for some context-sensitive grammar G_1 with the end marker $\#$ and regular language C_1.

A grammar G_1 and control language C_1 satisfying (28) will now be defined. The set of nonterminals of G_1 is the union

$$I_N \cup \{\#\} \cup \{[\alpha, f] \mid \alpha \in I_N \cup I_T, f \in F_1\}.$$

The terminal alphabet of G_1 is I_T, and the initial symbol X_0.

Consider a production $P \to Q$, labeled by $f \in F_1$. Assume that $P = x_1 \ldots x_r$, $r \geqq 1$, where each x_i is a letter of $I_N \cup I_T$. We denote by A_f the set consisting of all productions

$$[\alpha, f]\beta \to \alpha[\beta, f], \quad \alpha, \beta \in I_N \cup I_T, \quad \alpha \neq x_1,$$
$$[x_1, f]\beta \to x_1[\beta, f], \quad \beta \in I_N \cup I_T - \{x_2\},$$
$$[x_1, f]x_2\beta \to x_1[x_2, f]\beta, \quad \beta \in I_N \cup I_T - \{x_3\},$$
$$\ldots$$
$$[x_1, f]x_2 \ldots x_{r-1}\beta \to x_1[x_2, f] \ldots x_{r-1}\beta, \quad \beta \in I_N \cup I_T - \{x_r\},$$

and by E_f the set consisting of all productions

$$[\alpha, f]\# \to \alpha\#, \quad \text{if not both } r = 1 \text{ and } \alpha = x_1,$$
$$[\alpha_1, f]\alpha_2\# \to x_1\alpha_2\#, \quad \alpha \in I_N \cup I_T, \quad r > 2,$$
$$\ldots$$
$$[\alpha_1, f]\alpha_2 \ldots \alpha_{r-1}\# \to \alpha_1\alpha_2 \ldots \alpha_{r-1}\#, \quad \alpha_i \in I_N \cup I_T, \quad r > 2.$$

Furthermore, let B_f consist of the productions

$$\#\alpha \to \#[\alpha, f], \quad \alpha \in I_N \cup I_T.$$

The production set of G_1 is the union of F and the sets A_f, B_f and E_f, where f ranges over F_1.

Having completed the definition of G_1, we now introduce a regular substitution φ on the elements of F by

$$\varphi(f) = \begin{cases} f \text{ if } f \in F - F_1, \\ f \cup B_f W(A_f) E_f \text{ if } f \in F_1. \end{cases}$$

Thereby, $W(A_f)$ stands for the set of all words (including λ) over A_f. We now define

$$C_1 = \varphi(C). \tag{29}$$

Then C_1 is regular and (28) holds true. In fact, the definition of φ makes it possible, for productions $f \in F_1$, either to rewrite according to f or check that f is not applicable. For the latter purpose, one begins with a production in B_f and introduces a nonterminal $[\alpha, f]$. Productions in A_f are then applied to move nonterminals of this form towards the right end, where they can be eliminated by E_f. If P appears as a subword, no production in E_f can become applicable, and the derivation terminates. This completes the proof of the equation (25).

According to [3] and [14],

$$\mathcal{L}(1) = \mathcal{L}(1, 3, 0).$$

Consequently, the equations (18) hold true. Taking into account the inclusions (12) and the results concerning ordered grammars in [7], we may state the following theorem. The second and third sentences of the theorem have been established also in [6] and [4], respectively.

Theorem 2. *Regular control languages do not increase the generative capacity of context-sensitive grammars, not even under the broad interpretation of control words. A language generated by a programmed context-sensitive grammar is context-sensitive. A langugage generated by an ordered context-sensitive grammar is context-sensitive. A language generated by a context-sensitive matrix grammar is context-sensitive. A language generated by a periodically time-variant context-sensitive grammar is context-sensitive.*

3. The family $\mathcal{L}(1, 2, 1)$. We will first establish the following

Theorem 3. *For a context-sensitive grammar with a context-free control language, the broad interpretation of control words does not increase the generative power, i.e., the equation* (1) *holds true.*

Proof. The inclusion

$$\mathcal{L}(1, 2, 0) \subseteq \mathcal{L}(1, 2, 1)$$

follows by (9). The reverse inclusion is established exactly as the corresponding inclusion in the proof of equation (25) in Section 2. In fact, by the Lemma, it suffices to prove that any language of the form (26), where (27) in context-sensitive and C context-free, satisfies the equation (28), for some context-sensitive grammar G_1 with an end marker and context-free language C_1. The grammar G_1 and the substitution φ are defined exactly as in Section 2, and C_1 is defined by (29). Then C_1 will be context-free, whence Theorem 3 follows.

Theorem 4. *Every language in the family* $\mathcal{L}(1, 2, 1)$ *is recursive. Consequently, the family* $\mathcal{L}(1, 2, 1)$ *is properly included in the family* $\mathcal{L}(0)$.

Proof. By Theorem 3, it suffices to prove that every language in the family $\mathcal{L}(1, 2, 0)$ is recursive. Assume that

$$L = L_C(G , \emptyset) ,$$

where

$$G = (I_N , I_T , X_0 , F) , \quad I = I_N \cup I_T ,$$

is context-sensitive and C is context-free.

We note first that, for any word Q over I, the collection of all control words (under narrow interpretation) corresponding to derivations according to G of the form

$$Q \Rightarrow \ldots \Rightarrow Q \tag{30}$$

constitutes a regular language, denoted by $R(G , Q)$, which can be effectively constructed from G and Q. This can be shown by the following argument. Each intermediate word in a derivation (30) is of the same length as Q. We now construct a finite directed graph possessing a node for each word of $\lg(Q)$ over I. For each Q_1 and Q_2 (not necessarily distinct), there is an edge, labeled by f, from the node labeled by Q_1 to the node labeled by Q_2 exactly in case Q_1 directly yields Q_2 by an application of the production f. (Note that multiple edges are possible.) The node labeled by Q is the only initial and the only final node. Then $R(G , Q)$ is the language represented by this graph and, hence, regular.

It now follows that, for any word P over I_T, the collection of control words $f_1 f_2 \ldots f_u$ of derivations of P (according to G)

$$X_0 = P_0 \underset{f_1}{\Rightarrow} P_1 \underset{f_2}{\Rightarrow} P_2 \Rightarrow \ldots \underset{f_u}{\Rightarrow} P_u = P , \quad u \geqq 1 , \tag{31}$$

is a regular language $R_1(G , P)$. For there is only a finite set E of derivations (31), where the words P_i are distinct, [11, pp. 171—172]. Let (31) be an element of E. We define

$$\varphi_i = \begin{cases} R(G , P_{i-1})\{f_i\} & \text{if } R(G , P_{i-1}) \neq \emptyset , \\ \{f_i\} , & \text{otherwise} \end{cases} \qquad (i = 1 , \ldots , u)$$

and form the catenation $\varphi_1 \varphi_2 \ldots \varphi_u$. Then $R_1(G , P)$ is the (finite) union of these catenations, corresponding to different elements of E.

To decide, whether or not a given word P over I_T belongs to L, we first form the language $R_1(G , P)$. By [11, pp. 183—184], the intersection

$$C \cap R_1(G , P) \tag{32}$$

is context-free and, consequently, its emptiness is decidable. $P \in L$ if and only if the intersection (32) is not empty. This completes the proof.

Remark 3. It is an immediate consequence of Theorem 4 and the inclusion (11) that the families $\mathcal{L}(2 - \lambda, 2, 0)$ and $\mathcal{L}(2 - \lambda, 2, 1)$ are recursive. By complexity theory, one can strengthen Theorem 4 to the form: The family $\mathcal{L}(1, 2, 1)$ is properly included in the family of recursive languages. It remains an open problem whether or not this family properly includes the family of context-sensitive languages.

4. The family $\mathcal{L}(2, 3, 0)$. According to (13) and (14),

$$\mathcal{L}(2, 3, 1) = \mathcal{L}(0).$$

However, no nontrivial results are known about the size of the family $\mathcal{L}(2, 3, 0)$. Intuitively, the presence of 1 in the last argument place corresponds to jump instructions of Turing machines. Consequently, the replacement of 1 by 0 should considerably decrease the size of the family.

In derivations of languages in the family $\mathcal{L}(2, 3, 0)$, the essential thing is the number of nonterminals rather than their mutual order. This leads us to the following definitions.

Two words P and Q over an alphabet I are termed *letter-equivalent* if, for each $x \in I$, both P and Q contain the same number of occurrences of x (i.e., P is obtained from Q by a permutation of letters). Two languages L_1 and L_2 over I are termed letter-equivalent if, for each $P_1 \in L_1$, there is a letter-equivalent $P_2 \in L_2$, and vice versa.

For the notion of the *index* of a context-free grammar, the reader is referred to [8] or [13]. The notion is readily extended to pairs (G, C), where G is a context-free grammar and C a regular control language. It should be noted that although the index of G is finite, the index of (G, C) may still be infinite, for some regular C. A simple example of this is provided by G consisting of the productions

$$f_1 : X_0 \to X_0X_0, \quad f_2 : X_0 \to x$$

(where x is the only terminal) and C defined by the regular expression $f_1^* f_2^*$.

It is well-known that, for each context-free language, there is a letter-equivalent regular language. This result is now extended to concern a subfamily of $\mathcal{L}(2, 3, 0)$.

Theorem 5. *For each language of finite index in the family* $\mathcal{L}(2, 3, 0)$, *there is a letter-equivalent regular language.*

Proof. Assume that $L = L_C(G, \emptyset)$, where $G = (I_N, I_T, X_0, F)$ is context-free and C regular and, furthermore, the index of (G, C) equals a natural number k. For a word Q over $I_N \cup I_T$, we denote by $\delta(Q)$ the word obtained by erasing all terminals in Q, and by $\gamma(Q)$ the word obtained by erasing all nonterminals in Q. Consequently, for any word

$P \in L$, there is a derivation (31) with $f_1 f_2 \dots f_u \in C$ and $\delta(P_i) \leqq k$, $i = 0, \dots, u$. By the assumption, C is accepted by a finite deterministic automaton $(F, S, s_0, S_1, \varphi)$, where F is the alphabet, S the state set, s_0 the initial state, S_1 the final state set and φ the transition function.

Consider the (finite) collection of words over I_N with length $\leqq k$. We choose one representative from each class of letter-equivalent words, and denote the resulting set by E. We now construct a finite directed graph possessing a node for each element of the product set $E \times S$. There is an edge e from the node labeled by (Q_1, s_1) to the node labeled by (Q_2, s_2) (where the Q's and s's are not necessarily distinct) if and only if there is a production $X \to P$, labeled by f, in F such that each of the following conditions is satisfied: (i) X occurs in Q_1; (ii) Q_2 is letter-equivalent to a word obtained from Q_1 by replacing some occurrence of X by $\delta(P)$; (iii) $\varphi(s_1, f) = s_2$. Furthermore, e is labeled by $\gamma(P)$. (Note that multiple edges are possible.) The node labeled by (X_0, s_0) is initial in the graph, and each node labeled by (λ, s_1), where $s_1 \in S_1$, is final. Let

$$\{P_1, \dots, P_m\} \tag{33}$$

be the collection of all words over I_T appearing as labels of the edges e, and L_1 the language represented by our graph. It is easy to verify that L_1 is letter-equivalent to L. Furthermore, L_1 is regular over the alphabet (33) and, consequently, regular over I_T. Hence, Theorem 5 follows.

As an immediate corollary we obtain the following

Theorem 6. *Every language over a one-letter alphabet which belongs to the family* $\mathfrak{L}(2, 3, 0)$ *and possesses a finite index is regular.*

Theorem 6 is a special case of the following

Conjecture. The family $\mathfrak{L}(2, 3, 0)$ contains no nonregular languages over one letter.

Remark 4. To prove this conjecture it suffices, by (13), to consider matrix grammars. To point out some of the difficulties involved in a proof, let us consider a context-free matrix grammar

$$G_M = (\{X\}, \{x\}, X, M_1, \dots, M_k),$$

where each M_i is a finite sequence of $j(i)$ productions

$$X \to P_1, X \to P_2, \dots, X \to P_{j(i)}. \tag{34}$$

(For simplicity, we have assumed that there is only one nonterminal.) Denote

$$a_i = \lg(\delta(P_1 P_2 \dots P_{j(i)})) - j(i); \; b_i = \lg(\gamma(P_1 P_2 \dots P_{j(i)})).$$

If M_i is applied y_i times, $i = 1, \ldots, k$, in a derivation leading to a terminal word P, we obtain

$$\sum_{i=1}^{k} a_i y_i = -1 \tag{35}$$

and $P = x^v$, where

$$v = \sum_{i=1}^{k} b_i y_i . \tag{36}$$

It follows by the theory of systems of linear Diophantine equations that the language L_1 consisting of all words x^v, with v defined by (36), for some nonnegative solution

$$(y_1, \ldots, y_k) \tag{37}$$

of (35), is regular. The language $L(G_M)$ generated by G_M is a subset of L_1. But it may be a proper subset of L_1. This is due to the fact that although a_i is positive, in applying M_i the number of X's may still decrease if there are some productions $X \to \lambda$ beginning the sequence (34). Consequently, every solution (37) of (35) does not lead to a word x^v in $L(G_M)$, and the difference $L_1 - L(G_M)$ may be nonregular. The problem is quite the same in the general case where there are more than one nonterminals.

Remark 5. Stotskij, [14], has shown that the language

$$\{a^m b^n \mid 1 \leqq m , \quad 1 \leqq n \leqq 2^m\}$$

belongs to the family $\mathfrak{L}(2 - \lambda, 3, 0)$ and, hence, to the family $\mathfrak{L}(2, 3, 0)$ but it is not letter-equivalent to any regular language. Consequently, by Theorem 5, there are languages of infinite index in the family $\mathfrak{L}(2, 3, 0)$. If the above conjecture is true, there are no such languages over one letter.

We have seen (Theorem 1) that the family $\mathfrak{L}(2, 3, 0)$ is closed under a number of operations. Following Stotskij, [14], we shall now introduce another operation which is typical for this family.

The *quasi-intersection* of a language L_1 with a language L_2, in symbols, $L_1 \overline{\cap} L_2$ is the subset of L_1 consisting of all words P_1 such that there is a letter-equivalent word P_2 in L_2.

It follows that quasi-intersection is associative but not commutative. It is commutative in the sense that the languages $L_1 \overline{\cap} L_2$ and $L_2 \overline{\cap} L_1$ are letter-equivalent. The intersection of two languages is contained in their quasi-intersection. Quasi-intersection is idempotent and distributive over union, both from the left and from the right.

Theorem 7. *The family* $\mathfrak{L}(2, 3, 0)$ *is closed under quasi-intersection.*

Proof. Consider two languages

$$L = L_C(G\,, \emptyset) \quad \text{and} \quad L' = L_{C'}(G'\,, \emptyset)\,,$$

where C and C' are regular, and

$$G = (I_N\,, I_T\,, X_0\,, F) \quad \text{and} \quad G' = (I'_N\,, I'_T\,, X'_0\,, F')$$

are context-free grammars such that $I_N \cap I_{N'} = \emptyset$. Denote

$$I_1 = \{x_1 | x \in I_T \cup I'_T\}$$

and let, for each $P \in W(I_N \cup I_T)$, P_1 be the word obtained from P by replacing each letter $x \in I_T$ by the corresponding indexed letter x_1. Let F_1 be the set obtained from F by replacing the right side P of every production by P_1. The labels of the productions are left unaltered in the transition from F to F_1.

Consider a labeled production

$$f : X \to y_1 y_2 \ldots y_r\,, \quad y_i \in I'_N \cup I'_T$$

in F'. Let $y_{i_1}\,, \ldots\,, y_{i_k}$ be the letters of I'_T on the right side ($k \geqq 0$). Then f is replaced by the sequence of labeled productions

$$\begin{aligned} &f_0 : X \to \delta(y_1 y_2 \ldots y_r)\,,\\ &f_1 : (y_{i_1})_1 \to y_{i_1}\,,\\ &\ldots\\ &f_k : (y_{i_k})_1 \to y_{i_k}\,. \end{aligned}$$

The replacement is made for every $f \in F'$, and the resulting set of productions is denoted by F_2. (If the original production is $f : X \to \lambda$, it is replaced by $f_0 : X \to \lambda$.) Let C_1 be the language obtained from C' by replacing each letter f by the corresponding catenation $f_1 \ldots f_k f_0$. We introduce a new initial symbol Y_0 and the labeled production

$$g_0 : Y_0 \to X_0 X'_0\,.$$

It is easy to see that

$$L \barwedge L' = L_{C_2}(G_2\,, \emptyset)\,, \tag{38}$$

where

$$G_2 = (I_N \cup I'_N \cup I_1 \cup \{Y_0\}\,, I_T \cup I'_T\,, Y_0\,, \{g_0\} \cup F_1 \cup F_2)$$

and

$$C_2 = \{g_0\}\, CC_1\,.$$

In fact, we obtain first the collection of words of the form $P_1 X'_0$, where $P \in L$. The indices 1 and the nonterminal X'_0 are eliminated if and only

if a word letter-equivalent to P belongs to L'. This proves (38) and Theorem 7.

Remark 6. Some full AFL's, for instance, $\mathscr{L}(3)$ and $\mathscr{L}(2)$ are not closed under quasi-intersection. This follows because

$$(a^*b^*c^*) \,\overline{\cap}\, ((abc)^*abc) = \{a^n b^n c^n \mid n \geqq 1\}\,.$$

Remark 7. Theorems 5 and 7 have been established for the family $\mathscr{L}(2-\lambda\,,\,3\,,\,0)$ by Stotskij, [13], [14]. The above proof of Theorem 7 remains unaltered for any of the first six families (21).

Remark 8. We have pointed out that the family $\mathscr{L}(2\,,\,3\,,\,0)$ equals the family of languages generated by programmed grammars with context-free core productions and empty failure fields. A more general type, called an *appearance answering* context-free programmed grammar, has been introduced in [2]. However, it is easy to see that this generalization possesses the same generative capacity as context-free programmed grammars with arbitrary success and failure fields.

Remark 9. Friant, [3], has considered grammars which, in addition to a control language, have restrictions on the use of productions obtained by generalizing the ordering of productions, [4]. Thereby, the application of productions is understood in the narrow sense. It is an open problem to generalize the results to the case where the application is understood in the broad sense.

5. Conclusion. The following two tables summarize the results concerning the mutual relations between the families $\mathscr{L}(i\,,\,j\,,\,k)$ and $\mathscr{L}(i)$. (In addition, cf. (7) — (11).) Thereby, $\mathscr{R}$ denotes the family of recursive languages.

$$\mathscr{L} = \mathscr{L}(i\,,\,j\,,\,0)$$

i \ j	0	1	2	3
0	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$
1	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L}(1) \subseteq \mathscr{L} \subset \mathscr{R}$	$\mathscr{L} = \mathscr{L}(1)$
2	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L}(2) \subset \mathscr{L}$	$\mathscr{L}(2) \subset \mathscr{L}$
$2-\lambda$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L}(2) \subset \mathscr{L} \subset \mathscr{R}$	$\mathscr{L}(2) \subset \mathscr{L} \subset \mathscr{L}(1)$
3	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(2)$	$\mathscr{L} = \mathscr{L}(3)$

$$\mathscr{L} = \mathscr{L}(i\,,j\,,1)$$

i \ j	0	1	2	3
0	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$
1	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L}(1) \subseteq \mathscr{L} \subset \mathscr{R}$	$\mathscr{L} = \mathscr{L}(1)$
2	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$
$2 - \lambda$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L}(2) \subset \mathscr{L} \subset \mathscr{R}$	$\mathscr{L}(2) \subset \mathscr{L} \subset \mathscr{L}(1)$
3	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(0)$	$\mathscr{L} = \mathscr{L}(2)$	$\mathscr{L} = \mathscr{L}(3)$

In our estimation, the most interesting open problems are (i) to characterize the family $\mathscr{L}(2\,,3\,,0)$ and (ii) to determine whether or not $\mathscr{L}(1) = \mathscr{L}(1\,,2\,,0)$.

University of Turku
Finland

References

[1] Ábráham, S.: Some questions of phrase structure grammars. - Computational Linguistics 4 (1965), 61—70.

[2] Cohen, R. S. and Nash, B. O.: Parallel leveled grammars. - IEEE Conf. Record of 1969 Tenth Ann. Symp. on Switching and Automata Theory, 263—276.

[3] Friant, J.: Grammaires ordonnees — grammaires matricielles. - Université de Montréal (1968), 43 pp.

[4] Friš, I.: Grammars with partial ordering of the rules. - Information and Control 12 (1968), 415—425.

[5] Ginsburg, S. and Spanier, E. H.: Control sets on grammars. - Math. Systems Theory 2 (1968), 159—177.

[6] Rosenkrantz, D. J.: Programmed grammars and classes of formal languages. - J. Assoc. Comput. Mach. 16 (1969), 107—131.

[7] Salomaa, A.: On grammars with restricted use of productions. - Ann. Acad. Sci. Fennicæ, Ser. A I 454 (1969), 32 pp.

[8] —»— On the index of a context-free grammar and language. - Information and Control 14 (1969), 474—477.

[9] —»— Periodically time-variant context-free grammars. - Information and Control to appear.

[10] —»— Probabilistic and weighted grammars. - Information and Control 15 (1969), 529—544.

[11] —»— Theory of Automata. - Pergamon Press (1969), 276 pp.

[12] Stotskij, E. D.: Some restrictions on derivations in context-sensitive grammars. - Akad. Nauk SSSR Nauchno-Techn. Inform. Ser. 2 (1967), 35—38. (Russian.)

[13] —»— The notion of index in generalized grammars. - Ibid. (1969), 16—17. (Russian.)

[14] —»— Generative grammars with regulated derivations. - Ibid. (1968), 28—31. (Russian.)

Printed November 1970