ITERATION GRAMMARS AND LINDENMAYER AFL'S

Arto Salomaa
Department of Computer Science
University of Aarhus
Aarhus, Denmark

One of the first observations concerning L-systems was that the corresponding language families have very weak closure properties, in fact, many of the families are anti-AFL's, i.e., closed under none of the AFL operations. However, this phenomenon is due to the lack of a terminal alphabet rather than to parallelism which is the essential feature concerning L-systems. E.g., the family TOL of all TOL-languages is an anti-AFL, whereas the family ETOL is a full AFL. Later on we will see how L-systems can be used to convert language families with weak closure properties into full AFL's in a rather natural way.

The basic notion in this paper, K-iteration grammar, is a slight generalization of the notion introduced by van Leeuwen [57]. The motivation for such a notion is three-fold:

i) It provides a uniform framework for discussing OL-systems and all of their context-free generalizations.
ii) It shows the relation between OL-systems and (iterated) substitutions.
iii) It associates with each family K of languages (having certain mild closure properties) some full AFL's, obtained from K in the "Lindenmayer way".

We make the following conventions, valid throughout this paper. All language families discussed are non-trivial, i.e., they contain at least one language containing a non-empty word. (A language family is understood as in [102].) Two generative devices are termed <u>equivalent</u> if they generate the same language or else the language generated by one device differs from the language generated by the other through the empty word λ. (Thus in this sense, for any context-free grammar, there is an equivalent context-free grammar with no λ-rules.)

We introduce first some standard terminology and notations. Let K be a family of languages. A <u>K-substitution</u> is a mapping σ from some alphabet V into K. The mapping σ is extended to languages over V in the usual fashion. For language families K_1 and K_2, we define

(1) $\mathrm{Sub}(K_1, K_2) = \{\sigma(L) \mid L \in K_2 \text{ and } \sigma \text{ is a } K_1\text{-substitution}\}$.

If K_2 = OL or K_2 = TOL, families (1) are called <u>macro-OL</u> and <u>macro-TOL</u> families, respectively, and denoted by K_1MOL and K_1MTOL. Macros were introduced in [7] and [9], where especially the cases K_1 = F (the family of finite languages) and K_1 = R (the family of regular languages) were investigated. Using the fact (cf. [55]) that the family of EOL-languages is closed under arbitrary homomorphism, it is easy to show that

$$\text{FMOL} = \text{EOL}.$$

(There seems to be no short direct proof for the inclusion FMOL $\subseteq$ EOL.) Similarly, one can prove that

$$\text{FMTOL} = \text{RMTOL} = \text{ETOL}.$$

On the other hand, FMOL is properly included in RMOL because Herman's language

$$h^{-1}\{a^{2^n} \mid n \geqq 0\} \text{ with } h(a) = a,\ h(b) = \lambda,$$

is in the difference RMOL-FMOL, cf. [35]. The family RMOL is the smallest full AFL (and the smallest AFL) including the family OL, cf. [9] or [55]. It is also the closure of FMOL under inverse homomorphism.

We will now present the basic definition. Let K be a family of languages. A <u>K-iteration grammar</u> is a quadruple $G = (V_N, V_T, S, U)$, where V_N and V_T are disjoint alphabets (of nonterminals and terminals), $S \in V^+$ with $V = V_N \cup V_T$ (initial word) and $U = \{\sigma_1, \ldots, \sigma_n\}$ is a finite set of K-substitutions defined on V and with the property that, for each i and each $a \in V$, $\sigma_i(a)$ is a language over V. The language generated by such a grammar is defined by

(2) $$L(G) = \cup\, \sigma_{i_1} \ldots \sigma_{i_k}(S) \cap V_T^{*},$$

where the union is taken over all integers $k \geqq 1$ and over all ordered k-tuples $(i_1, \ldots, i_k)$ with $1 \leqq i_j \leqq n$. The family of languages generated by K-iteration grammars is denoted by K_{iter}. By $K_{iter}^{(t)}$ we denote the subfamily of K_{iter}, generated by such grammars, where U consists of at most t elements, for some $t \geqq 1$.

The different OL-families can now be easily characterized within this framework. Consider the special case K = F. Then

$$K_{iter}^{(1)} = F_{iter}^{(1)} = \text{EOL} = \text{FMOL}.$$

(Note that it suffices to choose, for each $a \in V$, $\sigma(a)$ to be the language consisting of the right sides of the productions with a on the left side.) Similarly,

$$F_{iter} = \text{ETOL}\ (= \text{FMTOL} = \text{RMTOL}).$$

The families with D and/or P are characterized as follows. D means that the σ's are homomorphisms, P means that the σ's are λ-free. Thus, EPDTOL is the subfamily of F_{iter}, obtained by such grammars where all substitutions σ are λ-free homomorphisms.

If one wants to consider families without E (OL, TOL, etc.), then one simply assumes that V_N is empty (which means that the intersection with V_T^* in (2) is superfluous). Note that in the general case the generative capacity is not affected by assuming that $S \in V_N$. Finally, the macro-families KMOL and KMTOL are obtained by K-iteration grammars satisfying the following condition. There is a sub-alphabet V_I of V_N such that, for each i and each $a \in V_I$, $\sigma_i(a)$ is a finite language over V_N. Furthermore, for each i and each $a \in V_T$, $\sigma_i(a)$ is empty and, for each i and each $a \in V_N - V_I$, $\sigma_i(a)$ is a language (in K) over the alphabet V_T. (Here it is assumed that K contains all finite languages.)

Thus, all context-free L-systems find their counterpart in this formalism. Note, however, that so far (apart from regular macros) one has not considered in the theory of L-systems cases more general than $K = F$.

The basic tool needed in proofs for closure results is the following Theorem 1. We say that a K-iteration grammar is λ-free iff each of the substitutions σ_i is λ-free.

Theorem 1. ([55], [57], [103]) Assume that K is a language family closed under finite substitution and intersection with regular languages. Then for each K-iteration grammar, there is an equivalent λ-free K-iteration grammar.

Applying standard AFL-theory and the technique used to prove Medvedev's Theorem for finite automata, one can establish the following results:

Theorem 2. Assume that K satisfies the hypothesis of Theorem 1 and, furthermore, contains all regular languages. Then all of the families K_{iter}, $K_{iter}^{(t)}$, for any $t \geq 1$, KMOL and KMTOL are full AFL's.

Thus, Theorem 2 can be applied whenever K is a cone (also called a full trio). Since the full AFL's associated with K are obtained by parallel rewriting, they are naturally called Lindenmayer AFL's. Apart from the obvious inclusions

$$\mathrm{KMOL} \subseteq \mathrm{KMTOL} \subseteq K_{iter}, \quad \mathrm{KMOL} \subseteq K_{iter}^{(1)} \subseteq K_{iter}^{(2)} \subseteq \dots \subseteq K_{iter},$$

very little is known about these AFL's, e.g., about the strictness of the inclusions.

It is shown by van Leeuwen ("Notes on pre-set pushdown automata", this volume) that $R_{iter}^{(1)}$ equals the family of languages accepted by pre-set pushdown automata. (In van Leeuwen's terminology, R_{iter} could be called "hyper-algebraic multi-extension of regular languages".)

A natural notion from the point of view of L-systems in AFL-theory is that of a hyper-AFL. By definition, a family K satisfying the hypothesis of Theorem 2 is a hyper-AFL iff $K_{iter} = K$. Hyper-AFL's are discussed in the paper by P.A. Christensen (this volume). This approach shows that, among the L-families, the family ETOL has a very interesting mathematical property.

Iteration grammars have been generalized by Derick Wood ("A note on Lindenmayer systems, spectra and equivalence", McMaster University Computer Science Technical Report No. 74/1) to cover L-languages with interactions. He also gives an example of how the uniform framework of iteration grammars can be used to generalize specific results. The example concerns the ultimate periodicity of spectra in EOL- and ETOL-systems, [20], [27]. Wood's result shows that the specific results mentioned depend only on the method of iterated substitution and not at all on the finiteness of the substitutions.