A RESULT ON THE STRUCTURE OF ETOL LANGUAGES

by

Andrzej Ehrenfeucht
Department of Computer Science
University of Colorado at Boulder
Boulder, Colorado

Grzegorz Rozenberg
Department of Mathematics
University of Antwerp, U.I.A.
Wilrijk, Belgium

ALL correspondence to second author.

ABSTRACT

A theorem on combinatorial structure of ETOL languages is proved. It is used to formulate various results allowing one to provide a number of languages that are not ETOL languages.

INTRODUCTION

One of the important research areas in formal language theory is the investigation of the (combinatorial) structure of languages in various language classes. This yields results of the form: "If K is a language belonging to the class of languages X then K has the following properties:..." Results of this kind are needed to understand the nature of languages in the given language class and to provide a tool for constructing languages that are not in the given class of languages (a task that is in general quite difficult).

This paper investigates the structure of ETOL languages (see [5] or [4]). Although ETOL languages play a central role in the theory of L systems (see e.g. [4] and [6]) we still have only few results describing their structure (see e.g. [1] and [3]). Hence, in our opinion, research in this direction must remain one of the main research streams in the theory of L systems.

We provide a direct result on the structure of ETOL languages and then formulate quite a number of its consequences all of which allow us to provide quite interesting examples of languages that are not ETOL languages.

We assume the reader to be familiar with rudiments of the theory of ETOL systems (e.g. in the scope of [4]).

PRELIMINARIES

To establish notation and terminology we recall now the notion of an ETOL system.

Definition. An ETOL *system* is a construct $G = (\Sigma,H,\omega,\Delta)$ where Σ is a finite nonempty alphabet, Δ is a subset of Σ, $\omega \in \Sigma^+$ and H is a finite set of finite substitutions on Σ (called *tables*). If each substitution from H is Δ-free than G is referred to as a *propagating* ETOL system (or EPTOL system for short). The *language* of G, denoted by L(G), is defined by $L(G) = \{x \in \Delta^* : x \in h_1 \ldots h_n(\omega)$ for some $n \geq 0$, $h_1,\ldots,h_n \in H\}$. (We write composition of functions in the left to right order, that is h_1 is applied first and h_n last. Also if $n = 0$ then $h_1 \ldots h_n$ is the identity mapping).□

A derivation in an ETOL system $G = (\Sigma,H,\omega,\Delta)$ is a precise description of how a word "is derived in G", that is, which words are obtained step-by-step, which tables are applied and how they map all occurrences of all the letters in intermediate words. This can be formalized, e.g. as in [4]. We will need only the following terminology and notation which we will introduce rather informally.

Let D be a derivation of x in G. Then the *full trace of* D, denoted as *ftrace* D, is a sequence $((x_1,h_1),(x_2,h_2),\ldots,(x_n,h_n),x_{n+1} = x)$ where $x_1 = \omega$, $x_2,\ldots,x_n$ are intermediate words obtained in D by using consecutively tables $h_1,\ldots,h_{n-1}$ and $x_{n+1} = x$ is the result of D, denoted as *res* D. The *control word of* D is the sequence of tables applied in D, hence it equals $h_1 \ldots h_n$.

It is very well known that for every ETOL system there exists (effectively) an equivalent propagating synchronized EPTOL system. Then it is rather trivial to obtain the following result which will be very useful in our considerations.

Theorem 1. There exists an algorithm which given any ETOL system produces an equivalent EPTOL system $G = (\Sigma,H,\omega,\Delta)$ such that

(1) $\omega \in \Sigma\backslash\Delta$,

(2) there exists a symbol F in $\Sigma\backslash(\Delta \cup \{\omega\})$ such that, for every a in Δ and every h in H, $h(a) = \{F\}$ and $h(F) = \{F\}$,

(3) for every a in Σ and every h in H, if $\alpha \in h(a)$ then either $\alpha \in \Delta^+$ or $\alpha = F$ or $\alpha \in (\Sigma\backslash(\Delta \cup \{F,\omega\}))^+$. □

We call F a *synchronization symbol of* G. Clearly we can always assume that G has only one synchronization symbol.

Also the following technical result on trees (that is ordered labeled trees) will be quite useful in our analysis of derivations in ETOL systems.

Lemma 1. Let T be a tree satisfying the following condition: if $e_1,\dots,e_k$ is a path in T such that labels of e_1 and e_k are equal, then each node $e_1,\dots,e_{k-1}$ has the out-degree equal 1. Let q be the number of labels used in T and let the out-degree of every node in T be bounded by p. Then the number of leaves of T is bounded by p^q.

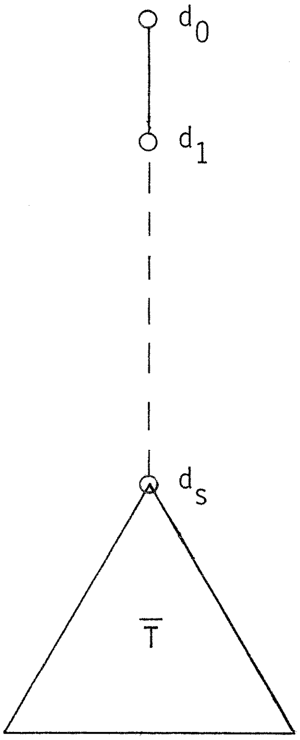
Proof. By induction on q.

$q = 1$. The result is obvious.

Let us assume that the claim holds for all trees satisfying the condition of the lemma and using no more than k labels.

$q = k+1$. Let c be the label of the root. If c does not label any other node of T then by the inductive assumption the number of leaves in T is bounded by $p \cdot p^k = p^{k+1} = p^q$. If c labels also another node in T then let $d_0,d_1,\dots,d_5$ be the longest path in T such that d_0 is the root of T and

d_0, d_s have the same label c. Then the condition required in the statement of the lemma implies that T must be of the form



where the tree $\overline{T}$ rooted at d_s is such that no node of it except for d_s is labeled by c. But then again (by the inductive assumption) p^q bounds the number of leaves in $\overline{T}$ and hence in T. □

We use standard language-theoretic terminology and notation and perhaps only the following notation needs an additional explanation. For a word x, $|x|$ denotes the length of x, *alph* x denotes the set of all letters occurring in x and, for an alphabet Θ, $\#_\Theta x$ denotes the number of occurrences of letters from Θ in x.

MAIN RESULT

In this section we prove the main result of this paper.

Theorem 2. Let K be an ETOL language over an alphabet Δ. Then there exists a positive integer constant k such that for every nonempty subset Θ of Δ and for every word x in K one of the following conditions holds:

(1) $\#_\Theta x \leq 1$,

(2) x contains a subword w such that $|w| \leq k$ and $\#_\Theta w \geq 2$,

(3) there exists an infinite subset M of K such that, for every y in M, $\#_\Theta y = \#_\Theta x$.

Proof. Since the above theorem trivially holds whenever K is finite let us assume that K is infinite. Let $G = (\Sigma, H, S, \Delta)$ be an ETOL system generating K and we assume that G satisfies the conclusion of Theorem 1 and F is the synchronization symbol of G.

Let $\overline{\Sigma} = \Delta \cup \{F, \overline{S}\} \cup \{[a,t] : a \in \Sigma \setminus (\Delta \cup \{F\})$ and $t \in \{0,1,2\}\}$ where $\overline{S}$ is a new symbol.

Let, for every h in H, $\overline{h}$ be the finite substitution on $\overline{\Sigma}^*$ defined by

$\overline{h}(\overline{S}) = \{[S,0],[S,1],[S,2]\}$,

$\overline{h}(a) = \{F\}$ for every a in $\Delta \cup \{F\}$,

and for every $a \in \Sigma \setminus (\Delta \cup \{F\})$:

$\overline{h}([a,0]) = \{F\} \cup \{\alpha \in \Delta^+ : \alpha \in h(a)$ and $\#_\Theta \alpha = 0\} \cup$

$\cup \{[b_1,0] \ldots [b_r,0] : b_1 \ldots b_r \in \Sigma \setminus (\Delta \cup \{F\})$ and $b_1 \ldots b_r \in h(a)\}$,

$\overline{h}([a,1]) = \{F\} \cup \{\alpha \in \Delta^+ : \alpha \in h(a)$ and $\#_\Theta \alpha = 1\} \cup$

$\cup \{[b_1,t_1][b_2,t_2] \ldots [b_r,t_r] : b_1, \ldots, b_r \in \Sigma \setminus (\Delta \cup \{F\})$,

$b_1 \ldots b_r \in h(a)$ and for some $j \in \{1, \ldots, r\}$, $t_j = 1$ and $t_\ell = 0$ for $\ell \neq j\}$,

$\overline{h}([a,2]) = \{F\} \cup \{\alpha \in \Delta^+ : \alpha \in h(a) \text{ and } \#_\Theta\alpha > 1\} \cup$
$\cup \{[b_1,t_1][b_2,t_2]\ldots[b_r,t_r] : b_1,\ldots,b_r \in \Sigma\backslash(\Delta \cup \{F\}),\ b_1\ldots b_r \in h(a)$
and either, for some $j \in \{1,\ldots,r\}$, $t_j = 2$ or, for some
$j_1,j_2 \in \{1,\ldots,r\}$, $j_1 \neq j_2$, $t_{j_1} = t_{j_2} = 1\}$.

Finally let $\overline{G} = (\overline{\Sigma},\overline{H},\overline{S},\Delta)$ be the ETOL system where $\overline{H} = \{\overline{h}|h \in H\}$.

Note that $\overline{G}$ results from G by attaching to each letter a form $\Sigma\backslash(\Delta \cup \{F\})$ an index 0, 1 or 2 (resulting in the letter [a,0],[a,1] or [a,2] respectively). If [a,i] occurs in a successful derivation in $\overline{G}$ then the corresponding (occurrence of a) letter in the corresponding derivation in G will contribute to the result of this derivation (in G) no letters from Θ if i = 0, one occurrence of a letter from Θ if i = 1 and at least two occurrences of letters from Θ if i = 2. It is rather easy to see that $L(\overline{G}) = L(G)$.

Let us analyze derivations in $\overline{G}$.

If a derivation in $\overline{G}$ starts with the production $\overline{S} \rightarrow [S,0]$ or with the production $\overline{S} \rightarrow [S,1]$ then the result of this derivation will satisfy condition (1) of the statement of the theorem.

Thus let us assume that the first step of a derivation D in $\overline{G}$ uses production $\overline{S} \rightarrow [S,2]$. Let *ftrace* D = $((x_0,g_0),(x_1,g_1),\ldots,(x_{m-1},g_{m-1}),x_m = x)$ and let i be the largest integer such that x_i contains an occurrence of a type 2 letter (i.e. a letter of the form [a,2]). We have two cases to consider.

(i) There exist r,s in $\{i+1,\ldots,m-1\}$ such that *alph* x_r = *alph* x_s, s > r and an occurrence of a letter (say c) in x_r contributes to x_s a word of the form $\alpha c\beta$ with $\alpha\beta \neq \Lambda$.

Then for every $n \geq 1$ we change the derivation D to the derivation $D_{(n)}$ constructed as follows.

First we use the sequence of tables $g_0 \dots g_{r-1}$ in precisely the same way as in D; thus we get x_r. Then to x_r we apply the sequence of tables $(g_r \dots g_{s-1})^n$ in such a way that each occurrence of a letter, except for the given occurrence of c, contributes on each iteration of $g_r \dots g_{s-1}$ a maximal in length word that an occurrence of this letter contributes from x_r to x_s in D; the given occurrence of c is rewritten in such a way that in each iteration of $g_r \dots g_{s-1}$ it contributes $\alpha c \beta$. In this way after applying $(g_r \dots g_{s-1})^n$ to x_r we obtain a word z_n. Finally we apply $g_s \dots g_{m-1}$ to z_n in such a way that each occurrence of a letter in z_n is rewritten in such a way that it contributes a word of maximal length that was obtained from the corresponding letter in x_s when x_s is rewritten by $g_s \dots g_{m-1}$ in D.

Thus the control word of $D_{(n)}$ is $g_0 \dots g_{r-1}(g_r \dots g_{s-1})^n g_s \dots g_{m-1}$ and clearly $\#_\Theta \mathit{res} D_{(n)} = \#_\Theta x$. From our assumption on r,s it follows that, for $n > 1$, $|x| < |\mathit{res} D_{(n)}| < |\mathit{res} D_{(n+1)}|$. Consequently if (i) holds then the condition (3) of the conclusion of the theorem holds.

(ii) There do not exist r,s in $\{i+1,\dots,m-1\}$ such that $\mathit{alph}\, x_r = \mathit{alph}\, x_s$, $s > r$ and x_r contains an occurrence of a letter, say c, which contributes to x_s a word of the form $\alpha c \beta$ with $\alpha\beta \neq \Lambda$.

Let us consider an occurrence of a letter of type 2 ([a,2] say) in x_i. We will show that its contribution to $x_m = x$ is not longer than a certain constant dependent on $\overline{G}$ only.

Let E be a subderivation tree rooted at the given occurrence of [a,2] in x_i. Let us relabel it in such a way that each node in it with a label d gets relabeled by $(d, \mathit{alph}\, x_j)$ where the node corresponds to the occurrence in the word x_j from D, $i \leq j \leq m-1$. In this way we obtain the tree $\overline{E}$ with the root labeled by $([a,2], \mathit{alph}\, x_i)$ which satisfies the assumption of Lemma 1.

Our construction of $\overline{E}$ from E implies that $\overline{E}$ does not use more than $q = \#\overline{\Sigma} \cdot 2^{\#\overline{\Sigma}}$ labels and the out degree of every node in $\overline{E}$ is bounded by $p = \max \{|\alpha| :$ there exist $\overline{h}$ in $\overline{H}$ and a in $\overline{\Sigma}$ such that $\alpha \in \overline{h}(a)\}$. Consequently Lemma 1 implies that if we set $k = p^q$ with p,q as above then condition (2) of the statement of the theorem holds.

This completes the proof of the theorem. □

APPLICATIONS

In this section we prove several corrollaries of Theorem 2. They allow one to provide various examples of languages that are not ETOL languages.

As a direct application of Theorem 2 we can demonstrate the following example of a language which is not an ETOL language.

Corollary 1. $K = \{(ab^n)^m : m \geq n \geq 1\}$ is not an ETOL language.

Proof. Let $\Delta = \{a,b\}$ and $\Theta = \{a\}$. Consider conditions (1), (2) and (3) of the statement of Theorem 2 and let us check them for the words of the form $(ab^n)^m$, $m \geq n \geq 1$ with $m \geq 2$. Then (1) obviously does not hold. Moreover for every positive integer k words of the form $(ab^{k+1})^{k+1}$ do not satisfy (2). Finally for every word x in K the set of words y in K such that $\#_\Theta x = \#_\Theta y$ is finite, and so (3) does not hold.

Consequently Theorem 2 implies that K is not an ETOL language. □

Before we show another application of Theorem 2 we recall a notion from [2].

Definition. Let K be a nonempty language over an alphabet Δ and let Θ be a nonempty subset of Δ. We say that Θ *is clustered in* K if there exist positive integer constants n, $m \geq 2$ such that for every word x in K with $\#_\Theta x \geq n$ there exists a subword y of x such that $|y| \leq m$ and $\#_\Theta y \geq 2$.

Theorem 3. Let K be an ETOL language over an alphabet Δ and let Δ_1, Δ_2 be a partition of Δ. If there exists a function ψ from nonnegative integers into nonnegative integers such that, for every x in K, $\#_{\Delta_2} x < \psi(\#_{\Delta_1} x)$ then Δ_1 is clustered in K.

Proof. The existence of such a function ψ implies that, for every x in K, the set of all words y such that $\#_{\Delta_1} y = \#_{\Delta_1} x$ is finite. Hence K must satisfy either condition (1) or condition (2) of the statement of Theorem 2, which implies that Δ_1 is clustered in K. □

In particular the above result yields the following example of a language that is not an ETOL language.

Corollary 2. $K = \{w \in \{a,b\}^* : \#_b w = 2^{\#_a w}\}$ is not an ETOL language.

Proof. If we take $\Delta = \{a,b\}$, $\Delta_1 = \{a\}$, $\Delta_2 = \{b\}$ and ψ defined by $\psi(n) = 2^n + 1$, then if K would be an ETOL language then $\{a\}$ must be clustered in K. Obviously $\{a\}$ is not clustered in K and so K is not an ETOL language. □

For our next application of Theorem 2 we need a definition first.

Definition. Let K be a nonempty language over an alphabet Δ and let Δ_1, Δ_2 be a partition of Δ. We say that Δ_1, Δ_2 *are* K-*equivalent* if for every x,y in K the following holds:

$\#_{\Delta_1} x = \#_{\Delta_1} y$ if and only if $\#_{\Delta_2} x = \#_{\Delta_2} y$. □

As a direct corollary of Theorem 2 we get the following result.

Theorem 4. Let K be an ETOL language over an alphabet Δ and let Δ_1, Δ_2 be a partition of Δ. If Δ_1, Δ_2 are K-equivalent then both Δ_1 and Δ_2 are clustered in K. □

In particular the above result yields the following example of a language which is not an ETOL language.

Corollary 3. $K = \{x \in \{a,b\}^+ : \#_a x = 2^n$ and $\#_b x = 3^n$ for some $n \geq 0\}$ is not an ETOL language.

Proof. Obviously neither $\{a\}$ nor $\{b\}$ are clustered in K but $\{a\},\{b\}$ are K-equivalent. Thus Theorem 4 implies that K is not an ETOL language. □

One of the operation considered in formal language theory is the shuffle operation defined as follows.

Definition. Let K_1, K_2 be languages over alphabets Δ_1 and Δ_2 respectively. The *shuffle of* K_1 *and* K_2, denoted as $K_1 \perp K_2$, is defined by

$$K_1 \perp K_2 = \{x_1 y_1 x_2 y_2 \ldots x_r y_r : r \geq 1,\ x_1, \ldots, x_r \in \Delta_1^*,\ y_1 \ldots y_r \in \Delta_2^*,\ x_1 \ldots x_r \in K_1 \text{ and } y_1 \ldots y_r \in K_2\}.$$ □

The class of ETOL languages has quite strong closure properties, e.g., it forms an AFL (see [4]). We will show now that this class of languages is not closed with respect to shuffle operator.

Theorem 5. The class of ETOL languages is not closed with respect to shuffle operation.

Proof. Take $K_1 = \{a^{2^n} b^{3^n} : n \geq 0\}$ and $K_2 = \{b^{3^n} a^{3^n} : n \geq 0\}$ and let $K = K_1 \perp K_2$. Obviously $\{a\},\{b\}$ are K-equivalent but since $\{a\}$ is not clustered in K, Theorem 4 implies that K is not an ETOL language. □

REFERENCES

[1] A. Ehrenfeucht and G. Rozenberg, On proving that certain languages are not ETOL, Acta Informatica, 6, pp. 407-415, 1976.

[2] A. Ehrenfeucht and G. Rozenberg, The number of occurrences of letters versus their distribution in some ETOL languages, Information and Control, 26, pp. 256-271, 1975.

[3] A. Ehrenfeucht and G. Rozenberg, On the (combinatorial) structure of L languages without interactions, Proceedings of the VIIth Ann. Symp. on the Theory of Computing, Albuquerque, pp. 137-144, 1975.

[4] G.T. Herman and G. Rozenberg, Developmental systems and languages, North-Holland, Amsterdam, 1975.

[5] G. Rozenberg, Extension of tabled OL systems and languages, International Journal of Computer and Information Sciences, 2, pp. 311-334, 1973.

[6] G. Rozenberg and A. Salomaa, The mathematical theory of L systems, in : J. Tou (ed.), Advances in Information Systems Science, v. 6, Plenum Press, New York, pp. 161-206, 1976.

Acknowledgments. The second author is very much indebted to IBM Belgium and NFWO foundation for supporting his stay at the University of Colorado at Boulder which made the work on this and other papers possible.