INFORMATICA, 1997, Vol. 8, No. 1, 43–56

D-GRAPHS IN CONTEXT-FREE LANGUAGE THEORY

Larisa STANEVIČIENĖ

Moscow State University, Department of Computational Mathematics and Cybernetics
Akademika Vargi 8–112, Moscow, Russia
E-mail:stanev@ccas.ru

Abstract. The paper presents a proposed approach in the context-free language theory. The main new notion is a graph defining a pushdown automaton (PDA). Each vertex of such graph is a pair (state, stack symbol). Each edge corresponds to a "command" and is labelled by input portion being read by the command and by a "charge" describing the stack word transformation. Some paths of the graph represent PDA's computations. The finite automata are a case of the pushdown graphs. The paper contains some of the author's results based on the approach – the notion of a D-language extending the notion of Dyck's language and the theorem on a representation of a context-free language as a morphical image of the intersection of a D-language with a local set.

Key words: pushdown automata; graphic characterization of context-free languages; morphic characterization of context-free languages.

1. Introduction. There exist a lot of theorems on morphic representations of formal languages. On the opinion of A. Salomaa (1981), the results form an interesting part of the formal language theory. The present paper contains a new theorem on morphic characterization of context-free languages which is most natural in the sense that its proof generalizes evidently the proof of the theorem Salomaa (1981) on morphic characterization of regular languages.

The development utilizes the author's method of the context-free languages investigation. This method represents pushdown automata (PDAs) as graphs. The graphs include the finite automata as their own subclass. The pushdown automaton core is defined as a special subset of the paths of the graphic representation of the automaton. We prove the core finiteness. For the last goal, the notion of a D-language is convenient. It generalizes the notion of Dyck's set which is known of combinatorial algebra.

The core notion is very helpful for many problems. It displays the characterization of the context-free language by the context-free expression and serves, in ansemble with latter, for a proof of the regularity of the so-called singular languages.

Section 2 of the paper defines D-languages, the above-mentioned generalization. Intuitively, a D-language may be interpreted, similarly to Dyck's set, as the set of balanced nested strings of matching parentheses of n types, where $n \geqslant 1$. But now distinct parentheses can match to instances of the same symbol. PDA computations (after it is normalized by the manner of Section 3) are words of a D-language. Therefore the D-language notion is useful in PDA theory.

Sections 3 and 4 define respectively a D-graph representing a PDA and its core. The core finiteness is deduced from the statement of the Section 2. The Growth Theorem is established on the base of which cores may be regarded as a characterization of the context-free languages.

Section 5 considers the main theorem of the paper. It implies immediately the Chomsky–Schutzenberger theorem.

The history of the development is such. The first version of Sections 2–4 was appeared in Stanevičienė (1983). The easy of access journals contain only brief sketches of our method (Stanevičienė, 1989; Stanevičienė, 1994; Stanevičienė, 1996). Brief sketches in English are in conference proceedings (Stanevičienė, 1994; Stanevičienė, 1995). The first version of Section 5 see in Stanevičienė (1988). Any knowledge of the mentioned papers is not necessary to verify the presented here facts. This citing aims only at indicating our priority in extremely perspective branch of the context-free languages theory.

2. D-languages

Definition 1. Let $\Sigma_($ and $\Sigma_)$ be alphabets which do not overlap. Let $\mathcal{P}$ be any subset of $\Sigma_(\times \Sigma_)$ such that $\Sigma_(= \{a \mid \exists b\ (a,b) \in \mathcal{P}\}$, $\Sigma_) = \{b \mid \exists a\ (a,b) \in \mathcal{P}\}$. Let $G_{\mathcal{P}}$ be a grammar defined by the following rules: $S \to \Lambda \mid aSbS$, $(a,b) \in \mathcal{P}$. Then the set $\mathcal{P}$ is called a D-*set*, and the language $\mathcal{L}_{\mathcal{P}} = L(G_{\mathcal{P}})$ is called a D-*language* (over the D-set $\mathcal{P}$).

Note that every Dyck's set is a D-language but not vice versa.

Definition 2. Let $\xi \in \mathcal{L}_{\mathcal{P}}$. The *partition index* of the word ξ is defined as $parti(\xi) = \max\{n \mid \xi = \xi_1 \ldots \xi_n,\ \xi_i \in \mathcal{L}_{\mathcal{P}} - \{\Lambda\} \text{ for } i = 1, \ldots, n\}$. The *width* of ξ is defined as $width(\xi) = \max\{parti(\psi) \mid \psi \in \mathcal{L}_{\mathcal{P}},\ \xi = x\psi y \text{ for some } x \text{ and } y\}$.

Definition 3. Let $\xi \in \mathcal{L}_\mathcal{P}$. The *nesting index* of ξ is defined as $strat(\xi) = \max\{n \mid \xi = x_1 \dots x_n y_n \dots y_1,\ x_i, y_i \notin \mathcal{L}_\mathcal{P},\ x_i y_i, x_i \dots x_n y_n \dots y_i \in \mathcal{L}_\mathcal{P}$ for $i = 1, \dots, n\}$. The *depth* of ξ is defined as $depth(\xi) = \max\{strat(\psi) \mid \psi \in \mathcal{L}_\mathcal{P},\ \xi = x\psi y$ for some x and $y\}$.

Theorem 1. *Let $\psi \in \mathcal{L}_\mathcal{P}$, $width(\psi) \leqslant w$, $depth(\psi) \leqslant d$. Let $g_{w,d}$ be defined inductively as follows: $g_{w,1} = 2w$, $g_{w,d} = (g_{w,d-1} + 2)w$ for $d > 1$. Then $\mid \psi \mid \leqslant g_{w,d}$.*

Proof. Let $parti(\psi) = n$. Then there exist $\psi_1, \dots, \psi_n \in \mathcal{L}_\mathcal{P} - \{\Lambda\}$ such that $\psi = \psi_1 \dots \psi_n$. Notice that $n \leqslant w$. Obviously, $\mid \psi \mid = \sum_{i=1}^{n} \mid \psi_i \mid$. We prove the theorem by induction on d. If $d = 1$, then $\mid \psi_i \mid = 2$ for $i = 1, \dots, n$. Hence, $\mid \psi \mid = 2n \leqslant 2w = g_{w,1}$. Let $d > 1$. Assume the fact for all words of which depth and width are no more than $d - 1$ and w respectively. Note that ψ_i, $i = 1, \dots, n$, has a form $a_i \xi_i b_i$, where $(a_i, b_i) \in \mathcal{P}$, $\xi_i \in \mathcal{L}_\mathcal{P}$, $depth(\xi_i) \leqslant d - 1$. ξ_i is a subword of ψ, therefore $width(\xi_i) \leqslant w$. By induction hypothesis $\mid \xi_i \mid \leqslant g_{w,d-1}$. Consequently, $\mid \psi_i \mid \leqslant g_{w,d-1} + 2$, and $\mid \psi \mid \leqslant (g_{w,d-1} + 2)w = g_{w,d}$.

3. D-graphs. Let us denote PDA $(K, \Sigma, \Gamma, Z_0, \delta, p_0, F)$ by M, as in Ginsburg (1966). Let us agree that: p, q are states; Λ is null word; $a \in \Sigma \cup \{\Lambda\}$; X, $Z \in \Gamma$; $\gamma \in \Gamma^*$, its right symbol is in the top of the stack.

Let us admit that:

(i) for every "command" $(p, a, Z) \to (q, \gamma)$ the stack word γ has one of the forms Λ, ZX, where $X \in \Gamma$, and if $Z = Z_0$, then $\gamma = Z_0 X$;

(ii) every final configuration has the form (f, Λ, Z_0) for some $f \in F$.

For such a "normal form PDA" the following theorem holds. Its proof is quite standard.

Theorem 2. *Let $L \subseteq \Sigma^*$ be a context-free language, and let $\perp \notin \Sigma$. Then:*

(i) $L = L(M)$ for a normal form PDA M;

(ii) if L is deterministic, then $L\perp = L(M)$ for a normal form DPDA M.

The rest of the paper is devoted to the consideration of the normal form PDAs.

Definition 4. Let $(p, x, \gamma'\gamma)$ be a configuration, and let $\gamma \neq \Lambda$. Then a pair (p, γ) will be called a collection.

Definition 5. Let: $(p, a, Z) \to (q, \gamma)$ be a command; $(p, \gamma' Z)$ be a collection; $\gamma\gamma' \neq \Lambda$; if $\gamma = \Lambda$, then $\Lambda \neq \gamma' = \gamma_1 Y$. Then we define an edge $\pi = (p, Z) \frac{a}{\mu(\pi)} (q, W)$ with an initial collection $bcol(\pi) = (p, \gamma' Z)$, a result

collection $ecol(\pi) = (q, \gamma_2)$, an initial vertex $beg(\pi) = (p, Z)$, a terminal vertex $end(\pi) = (q, W)$, a label $\omega(\pi) = a$, a charge $\mu(\pi)$ equal to $+X$, if $\gamma = ZX$ for some $X \in \Gamma$, and to $-Z$, if $\gamma = \Lambda$. Here if $\gamma = ZX$, then $\gamma_2 = \gamma' ZX$ and $W = X$, else $\gamma_2 = \gamma'$ and $W = Y$.

An edge is called charging or deleting respectively its charge.

Definition 6. In the triple $T = ((p, \gamma), \Pi, (p', \gamma'))$ let the members (p, γ) and (p', γ') be collections, $m \geqslant 0$, an edge sequence $\Pi = \pi_1 \ldots \pi_m$ satisfies the following: $bcol(\pi_1) = (p, \gamma)$, $ecol(\pi_m) = (p', \gamma')$, $ecol(\pi_{i-1}) = bcol(\pi_i)$ for $i = 2, ..., m$. Then T will be called a computation (of an automaton) with an initial collection $bcol(T) = (p, \gamma)$, a result collection $ecol(T) = (p', \gamma')$; vertices of the given edges are called computation vertices, and $beg(T) = beg(\pi_1)$, $end(T) = end(\pi_m)$; T has a label $\omega(T) = \omega(\pi_1) \ldots \omega(\pi_m)$ and a charge $\mu(T)$, which is defined by the following actions (i)–(iii):

(i) assign $\mu(\pi_1) \ldots \mu(\pi_m)$ to w;

(ii) while the equality $w = w_1 + X - X w_2$ holds for some $X \in \Gamma$, $w_1, w_2 \in \Gamma^*$ assign $w_1 w_2$ to w;

(iii) assign w to $\mu(T)$.

Definition 7. Let (p, γ) be a collection, $\gamma = \gamma' Z$. Then the triple $T = ((p, \gamma), , (p, \gamma))$, where the second member is empty, is called an empty computation; T has a vertex (p, Z); a label and a charge of T are empty.

Definition 8. Let $\gamma = W_1 \ldots W_n$, $n \geqslant 1$, $W_i \in \Gamma$ for $i = 1, \ldots, n$. Then a charge $+W_1 \ldots + W_n$ is noted by $+\gamma$, and a charge $-W_n \ldots - W_1$ is noted by $-\gamma$. Let us also note the empty charge by $+\Lambda$ or $-\Lambda$.

Let a computation T have an empty charge (charge $-\gamma$, $+\gamma$, where $\gamma \neq \Lambda$). Then T is called neutral (deleting, charging – respectively).

Definition 9. Let $\mathcal{T}$ be the computation set of a PDA M. Then we define a binary relation $\sim_M \subseteq \mathcal{T} \times \mathcal{T}$ (or $\sim$, if M is clear): $(T_1, T_2) \in \sim$ iff $T_1 = ((p, \gamma_1), \Pi, (q_1, \gamma_1')) \in \mathcal{T}$, $T_2 = ((p_2, \gamma_2), \Pi, (q_2, \gamma_2')) \in \mathcal{T}$ for some collections (p_1, γ_1), (q_1, γ_1'), (p_2, γ_2), (q_2, γ_2') and an edge sequence Π, where the emptiness of Π implies the following: $(p_1, \gamma_1) = (q_1, \gamma_1')$, $(p_2, \gamma_2) = (q_2, \gamma_2')$, $p_1 = p_2$, $\gamma_1 = \gamma\gamma_2 \wedge \gamma_1' = \gamma\gamma_2'$ or $\gamma_2 = \gamma\gamma_1 \wedge \gamma_2' = \gamma\gamma_1'$ for some γ. Hence $(T_1, T_2) \in \sim$ iff T_1 and T_2 have the same edge sequence or are both empty and have the same vertex.

Evidently, $\sim$ is an equivalence relation. The shortest member T of an equivalence class is uniquely defined by its edge sequence (by its vertex, if empty).

In an investigation of intrinsic properties of an automaton it is enough to consider only such computations. So do we in the remaining part of the paper.

Definition 10. Vertices (edges) of an automaton computations are called vertices (edges) of the automaton; (p_0, Z_0) is called an input vertex; (f, Z_0) is called an output vertex iff $f \in F$.

Definition 11. A computation T is called a sentence iff $beg(T)$ is input vertex and $end(T)$ is output vertex.

The following assertion is obvious for normal form PDA M.

Theorem 3. $L(M) = \{\omega(T) \mid T$ *is a sentence of the automaton* $M\}$.

Thus we have a graph representation of the pushdown automaton.

Definition 12. Let a computation $T = T_0\pi_1T_1T_2T_3\pi_3T_4$ be such that π_1, π_3 are edges, $\mu(T_2) = \mu(T_1T_2T_3) = \mu(\pi_1T_1T_2T_3\pi_3) = \Lambda$, $\mu(\pi_1T_1) = +\gamma$, $\mu(T_3\pi_3) = -\gamma$ for some $\gamma \in \Gamma^+$. Then we will say that π_1T_1 and $T_3\pi_3$ form a nest $(\pi_1T_1, T_2, T_3\pi_3)$ in T. When no ambiguity can arise, we write briefly "form a nest (in T)". The element π_1T_1 of the nest is called an opening computation.

Emphasize that the pair (π_1, π_3), where the edges π_1 and π_2 are considered in Definition 12, may be interpreted as an element of a D-set. Further, from Definition 11 it follows that every automaton sentence is a word of the D-language over the alphabet of the automaton edges. By this reason the graphs, which are defined in this section, are called D-graphs.

4. Core of a PDA

Definition 13. Let T be a nonempty computation, let $beg(T) = end(T)$. Then T is called a cycle.

Definition 14. A computation having nonempty label will be called a reading computation.

Definition 15. Let T be a neutral computation of a PDA M, w and d be nonnegative integers. Let T satisfy the following: if $T_0, \ldots, T_{m+1}$ are such that $T = T_0 \ldots T_{m+1}$ and T_i is a neutral cycle for $i = 1, \ldots, m$, then $m \leqslant w$; if T_{11}, $\ldots$, T_{1m}, $T_{21}, \ldots, T_{2(m+1)}$, $T_{31}, \ldots, T_{3m}$ are such that $T_{2(i+1)} = T_{1i}T_{2i}T_{3i}$ and (T_{1i}, T_{2i}, T_{3i}) is a nest with cyclic T_{1i}, T_{3i} for $1 \leqslant i \leqslant m$, then $m \leqslant d$. Then T is called a (w, d)-canon of M.

Denote the set of M's sentences being (w, d)-canons by $Core(M, w, d)$.

For the sake of brevity we write simply "canon" and "$Core(M)$" instead of "(w, d)-canon" and "$Core(M, w, d)$" respectively.

It is easy to prove the following proposition.

Lemma 1. *Let m, n, $i_1, \ldots, i_{mn}$ be natural numbers and $1 \leqslant i_j \leqslant m$ for $1 \leqslant j \leqslant mn$. Then at least n of the numbers $i_1, \ldots, i_{mn}$ coincide.*

The proof of the following lemma is based on the fact that edges forming a nest in a computation may be considered as matching parentheses, and so a neutral computation is a word of a D-language.

Lemma 2. *Let m be the number of the vertices of a PDA M. The width and the depth of M's canon are bounded by $(w+1)m$ and $(d+1)m^2$ respectively.*

Proof. Assume that T is a canon and $width(T) = s > (w+1)m$. From Definition 2 it follows that T has a subcomputation $T_1 \ldots T_s$ in which T_i, $i = 1, \ldots, s$, is neutral and nonempty. By Lemma 1 the inequality $s > (w+1)m$ implies the existence of a vertex P and numbers $1 \leqslant i_1 < \ldots < i_{w+2} \leqslant s$ such that $beg(T_{i_j}) = P$ for $1 \leqslant j \leqslant w+2$. Then the subcomputations $T_{i_j} \ldots T_{i_{j+1}-1}$ are neutral cycles for $1 \leqslant j \leqslant w+1$. Consequently, T does not satisfy Definition 15, contrary to the assumption that T is a canon.

Now assume that T is a canon and $depth(T) = s > (d+1)m^2$. From Definition 3 it follows that T has a subcomputation $T_{11} \ldots T_{1s} T_{2s} \ldots T_{21}$ in which T_{1i} and T_{2i}, $i = 1, \ldots, s$, form a nest. The inequality $s > (d+1)m^2$ implies the existence of vertices P, Q and numbers $1 \leqslant i_1 < \ldots < i_{d+2} \leqslant s$ such that $beg(T_{1i_j}) = P$ and $end(T_{2i_j}) = Q$ for $1 \leqslant j \leqslant d+2$. Then $T_{1i_j} \ldots T_{1(i_{j+1}-1)}$ and $T_{2(i_{j+1}-1)} \ldots T_{2i_j}$ are cycles and form a nest in T for $1 \leqslant j \leqslant d+1$. Consequently, T does not satisfy Definition 15, contrary to the assumption that T is a canon.

Definition 16. Let a computation T contain a neutral cycle or cycles forming a nest in T. Then T is called a composite computation. A composite computation having no proper composite subcomputation is called a formantis.

Let us define binary relations $\updownarrow_M, \downarrow_M$ on the computation set of a PDA M: $(T, T') \in \updownarrow_M$ (*with a history* $\langle T_1, T_2, T_3, T_4, T_5 \rangle$) iff $T = T_1T_2T_3T_4T_5$, $T' = T_1T_3T_5$, $T_2T_3T_4$ is a formantis, T_2 and T_4 form a nest in it; $(T, T') \in \downarrow_M$ (*with a history* $\langle T_1, T_2, \Lambda, \Lambda, T_5 \rangle$) iff $T = T_1T_2T_5$, $T' = T_1T_5$, T_2 is a cyclic formantis.

We refer to the union $\Downarrow_M = \updownarrow_M \cup \downarrow_M$ as a *reduction relation*, and to the $\Uparrow_M = \Downarrow_M^{-1}$ as a *growth relation*.

Definition 17. Let T, T' be M's computations. Let either $T' = T$ or for some $k > 0$ exist a sequence $H = (\langle T_{11}, T_{12}, T_{13}, T_{14}, T_{15} \rangle, \ldots, \langle T_{k1}, T_{k2},$

T_{k3}, T_{k4}, $T_{k5}\rangle$) such that $T' = T_{11}T_{13}T_{15}$, $T = T_{k1}T_{k2}T_{k3}T_{k4}T_{k5}$, $(T', T_{11}T_{12}T_{13}T_{14}T_{15}) \in \Uparrow_M$ with the history $\langle T_{11}, T_{12}, T_{13}, T_{14}, T_{15}\rangle$ and $(T_{(i-1)1}T_{(i-1)2}T_{(i-1)3}T_{(i-1)4}T_{(i-1)5}, T_{i1}T_{i2}T_{i3}T_{i4}T_{i5}) \in \Uparrow_M$ with the history $\langle T_{i1}, T_{i2}, T_{i3}, T_{i4}, T_{i5}\rangle$. Then H is called *a genealogy (from the ancestor T')* of T, k is called *the length* of the genealogy. If $T = T'$, then the length of T's genealogy from the ancestor T' is equal to 0.

The following theorem is an immediate consequence of Lemma 2 and Theorem 1.

Theorem 4. *Let m be the number of the vertices of a PDA M. The length of M's canon is bounded by $g_{(w+1)m,(d+1)m^2}$. Consequently, $Core(M)$ is a finite set.*

Next we establish that each of a PDA's sentences has an ancestor that is a canon (moreover, a noncomposite canon).

Lemma 3. *Any composite computation contains a formantis.*

Proof. It is clear that any composite computation contains neutral composite subcomputations. Let us choose a subcomputation T having the minimal length possible between the neutral composite subcomputations of a composite computation. Assume that T is not a formantis. Then, by Definition 16, two cases are possible:

(i) $\exists$ (T_0, T_1, T_2: $T_1 \neq T = T_0T_1T_2 \wedge$ (T_1 is a neutral cycle);

(ii) $\exists$ (T_0, T_1, T_2, T_3, T_4): $T_1T_2T_3 \neq T = T_0T_1T_2T_3T_4 \wedge$ (T_1 and T_3 is a pair of cycles forming a nest in T). In each of the cases there exists a neutral composite subcomputation having a lesser length than T, contrary to the choice of T.

Define two functions mapping computations in sets of computations.

Let T be a computation. Let $\mathcal{S}_1 = \{T_1T_3 \mid \exists$ (a neutral cycle T_2) $T = T_1T_2T_3$ $\}$, $\mathcal{S}_2 = \{T_1T_3T_5 \mid \exists$ (cycles T_2, T_4) (T_2, T_3, T_4) is a nest in $T = T_1T_2T_3T_4T_5\}$. Then $DelCycle(T)$ is equal to $\{T\}$, if $\mathcal{S}_1 = \emptyset$, and else it is equal to $\mathcal{S}_1$. If $\mathcal{S}_2 = \emptyset$, then $DelPair(T) = \{T\}$, else $DelPair(T) = \mathcal{S}_2$.

The following functions map some factorizations of a computation in a set of computations.

Let $TT'T''$ be a computation. Let $\mathcal{S}_3 = \{T_1T_3T_5 \in DelPair(TT'T'') \mid \exists$ (T_2, T_4, T_1', T_5') ($T = T_1T_2T_1'$, $T'' = T_5'T_4T_5$, $T_3 = T_1'T'T_5'$) $\}$. If $\mathcal{S}_3 = \emptyset$, then $PreservingDel(T, T', T'') = \{TT'T''\}$, else $PreservingDel(T, T', T'') = \mathcal{S}_3$.

Let $m \geqslant 1$, $T = T_1T_1' \ldots T_{m-1}T_{m-1}'T_m$ be a computation. Let $\mathcal{S}$ denote the union of two following sets:

$$\{T_1T_1' \ldots T_{i-1}T_{i-1}'\hat{T}_iT_i' \ldots T_{m-1}T_{m-1}'T_m \mid 1 \leqslant i \leqslant m,$$

$$\hat{T}_i \in DelCycle(T_i) \cup DelPair(T_i)\},$$

$$\{T_1T_1' \ldots T_{i-1}T_{i-1}'\hat{T}_iT_i' \ldots T_{j-1}T_{j-1}'\hat{T}_jT_j' \ldots T_{m-1}T_{m-1}'T_m \mid 1 \leqslant i < j \leqslant m,$$

$$\hat{T}_iT_i' \ldots T_{j-1}T_{j-1}'\hat{T}_j \in PreservingDel(T_i, T_i' \ldots T_{j-1}T_{j-1}', T_j)\}.$$

If $\mathcal{S} = \emptyset$, then $reduction(T_1, T_1', \ldots, T_{m-1}, T_{m-1}', T_m) = \{T\}$, else this function has the value $\cup_{\hat{T}_1T_1' \ldots \hat{T}_{m-1}T_{m-1}'\hat{T}_m \in \mathcal{S}} reduction(\hat{T}_1, T_1', \ldots, \hat{T}_{m-1}, T_{m-1}', \hat{T}_m)$.

Lemma 4. *Let $(T, T') \in \Downarrow_M$ with a history $\langle T_1, T_2, T_3, T_4, T_5\rangle$. Then there exists a canon $T_0 = T_{01}T_2T_3T_4T_{05}$ such that $beg(T_0) = beg(T)$, $end(T_0) = end(T)$.*

Proof. Let $T_0 \in reduction(T_1, T_2T_3T_4, T_5)$. Then there exist T_{01} and T_{05} such that $T_0 = T_{01}T_2T_3T_4T_{05}$; further, T_{01}, T_{05} do not contain neutral cycles or cycles forming a nest in T_0. Observe that $beg(T_{0j}) = beg(T_j)$, $end(T_{0j}) = end(T_j)$ for $j = 1, 5$.

Verify that T_0 is a canon. Recall that $T_2T_3T_4$ is the formantis. Because of this and the construction of T_0 we see that at least one of two cycles forming a nest in T_0 has a common nonempty subcomputation with $T_2T_3T_4$. Similarly, each T_0's neutral cycle contains $T_2T_3T_4$. Consequently, T_0 is (1,1)-canon at most. Thus, T_0 satisfies Definition 15.

Theorem 5. (The Growth Theorem) *Let M be a PDA. For everyone of its sentences T there exists $T_0 \in Core(M)$ such that $(T_0, T) \in \Uparrow_M^*$. In each element $\langle T_1, T_2, T_3, T_4, T_5\rangle$ of T's genealogy from the ancestor T_0, the formantis $T_2T_3T_4$ is contained in some element of $Core(M)$.*

Proof. It suffices to consider composite sentences. Let T be a composite sentence of a PDA M. By Lemma 3 T contains a formantis. Consequently, two cases arise:

(i) $T = T_1T_2T_3$ and T_2 is a cyclic formantis (then $(T, T_1T_3) \in \downarrow_M$);

(ii) $T = T_1T_2T_3T_4T_5$, $T_2T_3T_4$ is a formantis and T_2, T_4 are cycles forming a nest in T (then $(T, T_1T_3T_5) \in \updownarrow_M$).

Observe that in any case there exists T' such that $(T, T') \in \Downarrow_M$. As $|T'| < |T|$, this fact implies the existence of a sequence $T_0, \ldots, T_k$ such that $k > 0$,

T_0 is a canon, $T_k = T$, $(T_i, T_{i-1}) \in \Downarrow_M$ for $i = 1, \ldots, k$. Thus, $(T_0, T) \in \Uparrow_M^k$. It is easily seen that $beg(T_0) = beg(T)$, $end(T_0) = end(T)$. Consequently, $T_0 \in Core(M)$ and the first part of the theorem holds. The second part holds by Lemma 4.

It is important to emphasize that $Core(M)$ exhibits properties of M and is extremely helpful for answering numerous questions. In particular, $Core(M)$ is easily transformed into a context-free grammar equivalent to M and inheriting M's peculiarities.

5. A morphism theorem. Let Δ be an alphabet, $x \in \Delta^+$, $x = ay = zb$ for some $a, b \in \Delta$, $y, z \in \Delta^*$. Then we write $first(x) = a$, $last(x) = b$.

Let

$$Pairs(M) = \{(\pi_1, \pi_2) | \exists T \ (\pi_1, T, \pi_2) \text{ is a nest in a M's sentence}\}.$$

Our reduction technique (cf. the proof of Lemma 4) helps to prove the equality

$$Pairs(M) = \{(\pi_1, \pi_2) | \exists T \ (\pi_1, T, \pi_2)$$

is a nest in an element of $Core(M, 1, 1)\}$. The same technique is useful in the proof of the following assertion. Every computation of the length 2 is contained in an element of $Core(M, 2, 2)$. Indeed, if $T_1\pi_1 T \pi_2 T_2$ is a sentence and $(\pi_1, \pi_2) \in Pairs(M)$ (or T is empty), then $reduction(T_1, \pi_1, T, \pi_2, T_2) \subset Core(M, 1, 1)$ (respectively, $Core(M, 2, 2)$).

Note that every neutral computation is a word of the D-language $\mathcal{L}_{Pairs(M)}$.

The following two lemmas are immediate consequences of the definitions of a computation and $Pairs(M)$.

Lemma 5. *Let T be a neutral computation and $(\pi_1, \pi_2) \in Pairs(M)$. Let $\pi_1 first(T\pi_2)$ and $last(\pi_1 T)\pi_2$ be computations. Then $\pi_1 T \pi_2$ is a neutral computation.*

Lemma 6. *Let T_1 and T_2 be nonempty neutral computations. Let $last(T_1)$ $first(T_2)$ be a computation. Then $T_1 T_2$ is a neutral computation.*

Let E be the set of all M's edges. Let

$$A = \{\pi \in E | beg(\pi) = (p_0, Z_0)\},$$
$$B = \{\pi \in E | \exists f \in F \ end(\pi) = (f, Z_0)\},$$
$$C = \{\pi\pi' \in E^2 | \pi\pi' \text{ is not a computation}\}.$$

Then

$$R = (AE^* \cap E^*B) - E^*CE^*$$

is a local set (Salomma, 1981).

Lemma 7. *Let $x \in \mathcal{L}_{Pairs(M)} \cap R$. Then every subword $y \in \mathcal{L}_{Pairs(M)} - \{\Lambda\}$ of x is a neutral computation.*

Proof. By induction on the length m of the nonempty subword y. If $m = 2$, then the assertion follows from R's definition.

Let $k > 1$. Suppose the assertion holds for every nonempty subword of a length lesser than $2k$ and consider subwords of the length $2k$. The following cases are possible.

Case 1. Subword under consideration has the form ayb, where $(a, b) \in Pairs(M)$, $y \in \mathcal{L}_{Pairs(M)}$. By induction hypothesis y is a neutral computation. The condition $x \in \mathcal{L}_{Pairs(M)} \cap R$ implies that the pair (a, b) and the word y satisfy Lemma 5. Hence, ayb is a neutral computation.

Case 2. Subword under consideration has the form yz, where $y, z \in \mathcal{L}_{Pairs(M)} - \{\Lambda\}$. By induction hypothesis y and z are neutral computations. The condition $x \in \mathcal{L}_{Pairs(M)} \cap R$ implies that $last(y)first(z)$ is a computation. Hence, by Lemma 6 yz is a neutral computation.

Theorem 6. *Let $T \in \mathcal{L}_{Pairs(M)}$. T is a nonempty sentence iff $T \in R$.*

Proof. It is easy seen that every nonempty sentence belongs to R. Let $T \in \mathcal{L}_{Pairs(M)} \cap R$. Then T is a neutral computation by Lemma 7. The condition $T \in R$ implies $first(T) \in A$ and $last(T) \in B$. Consequently, T is a sentence.

The following theorem is main.

Theorem 7. *Let $L \subseteq \Sigma^*$ be a context-free language. Then there exist a D-language $\mathcal{L}$, a local set R, and a fine morphism (Salomaa, 1981) h such that*

$$L = h(\mathcal{L} \cap R).$$

Proof. A context-free language L is accepted by a PDA M. Let a local set R be defined as above. Let a morphism

$$h : E^* \to \Sigma^*$$

be defined by the following formulas

$$h(\pi) = \omega(\pi), \ \pi \in E.$$

By Theorem 6 the set of M's sentences is $\mathcal{L}_{Pairs(M)} \cap R$. By Theorem 3 $L = h(\mathcal{L}_{Pairs(M)} \cap R)$.

Let $\mathcal{L}_{\mathcal{P}}$ be a Dyck's language over a D-set $\mathcal{P} \subset \sum_{(} \times \sum_{)}$, where

$$|\Sigma_{(}| = |\Sigma_{)}| = |\mathcal{P}| = |Pairs(M)|.$$

Let $\Delta = \Sigma_{(} \cup \Sigma_{)}$ and

$$\phi : Pairs(M) \to \mathcal{P}$$

be a bijection.

From the definition of the function ϕ it follows that the function

$$\zeta : \Delta \to E,$$

given by the formulas (remember that each symbol of the alphabet Δ enters in a pair of the D-set $\mathcal{P}$):

$$\forall (a, b) \in \mathcal{P}\ \zeta(a) = a',\ \zeta(b) = b',\ \phi(a', b') = (a, b),$$

is a surjection. Note that the function ζ induces a very fine (Salomaa, 1981) (mapping every symbol into a symbol) morphism

$$\zeta : \Delta^* \to E^*.$$

The definition of the function ϕ implies that the function

$$\Phi : \mathcal{L}_{Pairs(M)} \to \mathcal{L}_{\mathcal{P}},$$

given by the formulas

$$\begin{aligned}
&\Phi(\Lambda) = \{\Lambda\},\\
&\Phi(\pi_1 T \pi_2) = a\Phi(T)b,\ T \in \mathcal{L}_{Pairs(M)},\\
&(\pi_1, \pi_2) \in Pairs(M),\ (a, b) = \phi(\pi_1, \pi_2),\\
&\Phi(T_1 T_2) = \Phi(T_1)\Phi(T_2),\ T_1, T_2 \in \mathcal{L}_{Pairs(M)},
\end{aligned}$$

is a bijection.

For every subword T of an element of $\mathcal{L}_{Pairs(M)}$ denote by $\Phi(T)$ the following set $\{x_2 | (\exists T_1, T_2, T_3)\ (T_2 = T,\ T_1 T_2 T_3 \in \mathcal{L}_{Pairs(M)},\ \Phi(T_1 T_2 T_3) = x_1 x_2 x_3,\ |T_i| = |x_i| \text{ for } i = 1, 2, 3)\}$.

Let

$$A' = \cup_{\pi \in A} \Phi(\pi),\ B' = \cup_{\pi \in B} \Phi(\pi),$$
$$C' = \Delta^2 - \cup_{\pi_1 \pi_2 \in E^2 - C} \Phi(\pi_1 \pi_2).$$

Then

$$R' = (A'\Delta^* \cap \Delta^* B') - \Delta^* C' \Delta^*$$

is a local set, and

$$\{\Phi(T) | T \text{ is a sentence of } M\} = \mathcal{L}_{\mathcal{P}} \cap R'.$$

Now the following equality is evident

$$L(M) = h(\zeta(\mathcal{L}_{\mathcal{P}} \cap R')),$$

where h was defined in the proof of Theorem 7. As ζ is a very fine morphism, the morphism

$$h' = h\zeta : \Delta^* \to \Sigma^*$$

is fine.

So constructed Dyck's language $\mathcal{L}_{\mathcal{P}}$, the local set R', and the fine morphism h' make clear that Theorem 7 implies

Consequence (Chomsky–Schutzenberger theorem). Let $L \subseteq \Sigma^*$ be a context-free language. Then there exist a Dyck language $\mathcal{L}$, a local set R, and a morphism h such that $L = h(\mathcal{L} \cap R)$.

REFERENCES

Salomaa, A. (1981). *Jewels of Formal Language Theory.* Rockville.

Stanevičienė, L.I. (1983). *Lectures on LR(k)-parsing.* Moscow State University. (in Russian).

Stanevičienė, L.I. (1989). On an instrument for studying context-free languages. *Cybernetics,* **4**, 135–136 (in Russian).

Stanevičienė, L.I. (1994). A recursively unsolvable problem in deterministic pushdown acceptors. *Reports of the Russian Academy of Sciences,* **6**, 744–746.

Stanevičienė, L.I. (1996). A recursively unsolvable problem. *Proceedings of the Higher Educational Establishments. Mathematics,* **6**(409), 1–8.

Stanevičienė, L.I. (1994). On an approach in context-free language theory. In *Proc. of the Second International Conf. Current Problems of Fundamental Sciences.* Moscow.

Stanevičienė, L.I. (1995). On the equivalence of pushdown automaton vertices. In *Third International Conference of Women-Mathematicians. Abstracts.* Russia, Voronezh.

Stanevičienė, L.I. and B.F. Melnikov (1988). *An Extension of Minimal Linear Languages.* Moscow State University (in Russian).

Ginsburg, S. (1966). *The Mathematical Theory of Context-Free Languages.* McGraw-Hill, New York.

Received November, 1996

L. Stanevičienė received the Degree of Candidate of Physical and Mathematical Sciences from the Computer Center of the USSR Academy of Sciences in 1972. She is a senior tutor of the Computational Mathematics and Cybernetics Department of the Moscov State University. Her research interests include formal language theory and its applications.

D-GRAFAI NEPRIKLAUSANČIŲ NUO KONTEKSTO KALBŲ TEORIJOJE

Larisa STANEVIČIENĖ

Straipsnyje nagrinėjami darbo su nepriklausančiomis nuo konteksto kalbomis metodai, kuriuose automatas su stekine atmintimi aprašomas grafu. Kiekvieną tokio grafo viršūnę atitinka pora (būsena, steko alfabeto ženklas), o kiekvieną lanką – „komanda ". Lankai žymimi komandos skaitomų įvesties duomenų porcijomis ir „krūviu", aprašančiu, kaip transformuoti steko viršūnėje esantį žodį. Automato su stekine atmintimi atliekami skaičiavimai aprašomi tam tikrais keliais grafe. Baigtiniai automatai yra atskiras tokių grafų atvejis. Staripsnyje aprašyta Diko kalbas praplečiančių D kalbų notacija ir teorema apie nepriklausančios nuo konteksto kalbos vaizdavimą D kalbos ir lokaliosios aibės piūvio morfiniu vaizdu.