DECIDABILITY OF THE EQUIVALENCE PROBLEM FOR DETERMINISTIC PUSHDOWN AUTOMATA

V. Yu. Meitus

UDC 510.53

Decidability of the equivalence problem is proved for deterministic pushdown automata. A comparison algorithm for two automata is described. The main theorem leads to solution of a number of open problems in the theory of program schemes and in formal language theory.

INTRODUCTION

The notion of deterministic pushdown automata (DPDA) was first introduced by Schützenberger [1] (although under a different name) and subsequently developed in [2, 3]. Context-free languages accepted in DPDA are easy to parse and many algorithms used in language theory are effective for this particular class of languages. All this has stimulated numerous studies of DPDA [1-19], which have produced a large volume of results and have highlighted the relationship between problems in DPDA theory and the theory of automata and transducers, the theory of program schemes, and language theory.

The equivalence problem is one of the best known open problems for DPDA: given two DPDA $\mathfrak{A}$ and $\mathfrak{B}$, can we decide that they accept the same language or not? This equivalence problem was first posed in [2], but so far it remains unsolved. We know that the equivalence problem for nondeterministic pushdown automata is undecidable. The inclusion problem for DPDA is undecidable [3], but a whole range of problems that are undecidable for nondeterministic pushdown automata are decidable for DPDA. For instance, the following problem is decidable for the DPDA $\mathfrak{A}$ and any finite automaton A: decide if the languages $\mathfrak{L}(\mathfrak{A})$ and $\mathfrak{L}(A)$ are equal. On the other hand, for a nondeterministic pushdown automaton even a weaker problem is undecidable: decide if an arbitrary context-free language accepted by this automaton is identical with the language Σ^* [3].

Analysis of the DPDA equivalence problem has led to a number of results for different DPDA subclasses which are connected with the equivalence problem [6-12]. A number of original methods and approaches to DPDA analysis have been developed. The problems identified in many studies are reducible to the DPDA equivalence problem, and the absence of a solution for this problem has restricted the development of certain lines of research in the theory of languages and transducers and the theory of program schemes. This accounts for the continuing interest that researchers show in the DPDA equivalence problem [14].

The present paper provides a complete proof of decidability of the equivalence problem for DPDA, which was originally obtained by the author back in 1982. The original proof was based on the comparison of the pair of categorical automata constructed from the given DPDAs. Subsequently the proof has been simplified and here we present a proof that does not require the theory of categorical automata.

The general outline of the proof is based on an algorithm that constructs a category system K from a given pair of DPDAs by the procedure Π and then analyzes the direct limit of the system K, $\lim_{\rightarrow} K$. As a corollary of the main theorem, we obtain solutions for a number of open problems in the theory of program schemes, automata theory, and language theory.

The algorithm proposed for the particular case of real-time strict DPDAs, i.e., DPDAs without ε-rules, was originally reported in [20].

Translated from Kibernetika i Sistemnyi Analiz, No. 5, pp. 20-45, September-October, 1992. Original article submitted December 9, 1991.

1060-0396/92/2805-0672$12.50

The paper consists of two parts. The first part introduces the basic definitions and considers the general scheme of the algorithm that defines the decision procedure; it outlines the method of proof and presents all the propositions (Secs. 1 and 2). The second part contains the proofs. Lemmas and theorems that are proved in [20] and that do not require any further extension to the general case are given without proof, with simple references to [20]. In these cases, the number of the corresponding result is preceded by [20] (thus, a reference to Lemma 1 of [20] is cited as Lemma [20].1). Except for these references, the paper does not require familiarity with [20]. All other results, including the decidability of the equivalence problem, are proved in full.

1. DETERMINISTIC PUSHDOWN AUTOMATA AND SYSTEMS OF EQUATIONS

A deterministic pushdown automaton (DPDA) is an ordered tuple of symbols $\mathfrak{A} = (S, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where S, Σ, Γ are finite sets (the state set of the automaton, the set of input symbols or the input alphabet, and the set of pushdown symbols or the pushdown alphabet), $q_0 \in S$ is the initial state, $Z_0 \in \Gamma$ is the starting pushdown symbol, $F \in S$ is the final state. The function δ: $\Sigma \times S \times \Gamma \to S \times \Gamma^*$ associates to each tuple of symbols $(a, \alpha, A) \in \Sigma \times S \times \Gamma$ (by assumption, the empty word $\varepsilon \in \Sigma$) at most one element of the set $S \times \Gamma^*$; if $a \neq \varepsilon$ and $\delta(a, \alpha, A) = (\beta, \gamma)$, then $\delta(\varepsilon, \alpha, A) = \varnothing$; if $\delta(\varepsilon, \alpha, A) = (\beta, \gamma)$, then $\delta(a, \alpha, A) = \varnothing$ for all $a \in \Sigma$, $a \neq \varepsilon$.

Each element θ of the graph of the function δ is usually written in the form θ: $(a, \alpha, A) \to (\beta, \gamma)$; it is called a rule of the automaton $\mathfrak{A}$. Rules of the form $(\varepsilon, \alpha, A) \to (\beta, \gamma)$ are called ε-rules. If $\gamma = A\gamma'$ in θ, then the application of the rule θ implies that an automaton scanning the pushdown symbol A in state α reads the input symbol a and goes to state β, writing the word γ' in the pushdown store above the symbol A; if $\gamma = \varepsilon$, then the automaton goes to state β erasing the symbol A from the pushdown list.

Using the rules of the automaton $\mathfrak{A}$, we can inductively define on the set $\Sigma^* \times S \times \Gamma^*$ the binary computation relation $\Rightarrow$:

1) $(ax, \alpha, \xi A) \Rightarrow (x, \beta, \xi\gamma)$ if the rule θ: $(a, \alpha, A) \to (\beta, \gamma)$ exists;

2) if $(ax, \delta, \xi) \Rightarrow (ay, \alpha, \zeta A)$ and the rule θ: $(a, \alpha, A) \to (\beta, \gamma)$ exists, then $(ax, \delta, \xi) \Rightarrow (y, \beta, \zeta\gamma)$;

3) there are no other pairs in the relation $\Rightarrow$.

Any pair of tuples in the relation $\Rightarrow$ is called a computation.

The language $\mathfrak{L}(\mathfrak{A})$ accepted by the DPDA $\mathfrak{A}$, is the set of those and only those words w in the alphabet Σ for which a computation of the form $(w, q_0, Z_0) \Rightarrow (\varepsilon, F, \gamma)$ exists, i.e., $\mathfrak{L}(\mathfrak{A}) = \{w | (w, q_0, Z_0) \Rightarrow (\varepsilon, F, \gamma)\}$. The DPDA $\mathfrak{A}$ is said to operate with an empty stack if for any word $w \in \mathfrak{L}(\mathfrak{A})$ and the computation $(w, q_0, Z_0) \Rightarrow (\varepsilon, F, \gamma)$ defined by this word, the string γ is empty, i.e., the starting pushdown symbol Z_0 is erased when the automaton passes to the final state.

The DPDA equivalence problem is stated as follows: for any pair of DPDAs $\mathfrak{A}$ and $\mathfrak{B}$, decide if the languages $\mathfrak{L}(\mathfrak{A})$ and $\mathfrak{L}(\mathfrak{B})$ are equal or not. If $\mathfrak{L}(\mathfrak{A}) = \mathfrak{L}(\mathfrak{B})$, then the automata $\mathfrak{A}$ and $\mathfrak{B}$ are equivalent. If $\mathfrak{L}(\mathfrak{A}) \neq \mathfrak{L}(\mathfrak{B})$, then the two automata are inequivalent. In what follows, we assume that the DPDAs being compared operate with an empty stack and contain ε-rules that erase only pushdown symbols, i.e., rules of the form $(\varepsilon, \alpha, A) \to (\beta, \varepsilon)$. Moreover, the right-hand side of each rule from δ has the form $(\beta, RT_m \ldots T_1)$, where $m \geq 0$, or (β, ε). It is easy to verify that these conditions do not restrict the generality of the result, while simplifying the constructions.

We introduce some notation. Let $\mathfrak{A}$ be a DPDA. For each pushdown symbol $P \in \Gamma$, denote by W^P the set of rules of the automaton $\mathfrak{A}$, that erase the symbol P from the pushdown store, i.e., the set of rules of the form $(a, \alpha, P) \to (\beta, \varepsilon)$, where a may be the empty word. For a given pair $(\alpha, P) \in S \times \Gamma$, denote by $U^{\alpha P}$ the set of all rules of the automaton $\mathfrak{A}$, which, reading a nonempty symbol from Σ in the state α with the pushdown symbol P at the top, write the string γ of length greater than or equal to zero in the pushdown store, i.e., rules of the form $(a, \alpha, P) \to (\beta, P\gamma)$ $(a \neq \varepsilon, |\gamma| \geq 0)$. For each symbol $Q \in \Gamma$, define the set $\Phi^Q = \{(\alpha, a) | \exists[(a, \alpha, Q) \to (\rho, \varepsilon)] \in W^Q\}$ of pairs (state α, input symbol a) such that the set W^Q contains a rule with left-hand side of the form (a, α, Q).

For any rule θ: $(b, \alpha, Q) \to (\rho, \varepsilon) \in W^Q$, we denote the pair (α, b) by the symbol $[Q]$ if the specific values of α, b are irrelevant; the notation $[Q]_i$, $[Q]_j$ is used for different rules $\theta \in W^Q$. If the specific value of ρ is also irrelevant and only its relationship with a rule from W^Q matters, then we use the notation $\mu[Q]$. Thus, a representative rule of the set W^Q in general may be written as θ: $([Q], Q) \to (\mu[Q], \varepsilon)$. For particular values of b, α, Q, the state $\mu[Q]$ in the rule θ is denoted by the symbol $\mu(b, \alpha, Q)$.

Let us now construct the system of equations $\mathfrak{U}(\bar{X})$ from the given DPDA $\mathfrak{U}$. For each pair $(\alpha, Q) \in S \times \Gamma$, we define the set of variables $X_{\beta b}^{\alpha Q}$, where $(\beta, b) \in \Phi^Q$. With the variable $X_{\beta b}^{\alpha Q}$ we associate the language $\hat{X}_{\beta b}^{\alpha Q} = \{xb\}$ – the set of words in the alphabet Σ with the computation $(x, \alpha, Q) \Rightarrow (\varepsilon, \beta, Q)$ that never erases the symbol Q from the pushdown store. In other words, if the automaton $\mathfrak{U}$ starts processing in the state α with the pushdown symbol Q at the top, then reading the word x from the input tape it writes and erases symbols above the symbol Q in the pushdown store and finally passes to the state β, where it scans the same symbol Q.

The only exception are variables of the form $X_F^{\alpha Z_0}$ for which the set of words $\hat{X}_F^{\alpha Z_0}$, if it is nonempty, may contain words that take the automaton to the final state F with erasure of the starting pushdown symbol Z_0, i.e., the automaton operates with an empty stack.

The construction of the system $\mathfrak{U}(\bar{X})$ starts with the variable $X_F^{q_0 Z_0}$, where q_0 and F are the initial and the final state of the automaton $\mathfrak{U}$ and Z_0 is the starting pushdown symbol. For each variable $X_{\beta b}^{\alpha Q}$, the system of equations $\mathfrak{U}(\bar{X})$ includes the following equation:

$$X_{\beta b}^{\alpha Q} = \bigvee_{\substack{\theta \in U^{\alpha Q} \\ \theta: (a, \alpha, Q) \to (\rho, QT_m \ldots T_1)}} \bigvee_{i=1}^{m} \bigvee_{[(\alpha_i, b_i)] \in \Phi^{T_i}} a X_{\alpha_1 b_1}^{\rho T_1} X_{\alpha_2 b_2}^{\mu(\alpha_1 b_1, T_1) T_2} \ldots X_{\beta b}^{\mu(\alpha_m b_m, T_m) Q} \vee b \delta_{\alpha\beta}. \tag{1.1}$$

The product of variables in (1.1) defines concatenation and the disjunction defines the union of the corresponding languages. Details of the general theory linking pushdown automata, CF-languages, systems of equations, and their minimal solutions can be found in [13]. A substantive discussion explaining the construction of Eq. (1.1) and the system $\mathfrak{U}(\bar{X})$ can be found in [20]. The collection of all variables entering the system $\mathfrak{U}(\bar{X})$ will be denoted by the symbol $\bar{X}$.

Note that the disjunctive expression in Eq. (1.1) is representable in the form $\bigvee_{\theta \in U^{\alpha Q}} a \mathfrak{U}_{\beta b}^{\rho QT_m \ldots T_1}$, where the expression $\mathfrak{U}_{\beta b}^{\rho QT_m \ldots T_1}$ replaces the disjunction $\bigvee_{i=1}^{m} \bigvee_{(\alpha_i, b_i) \in \Phi^{T_i}} X_{\alpha_1 b_1}^{\rho T_1} \ldots X_{\beta b}^{\mu(\alpha_m, b_m, T_m) Q}$. The string of pushdown symbols $\xi = QT_m \ldots T_1$ is called the characteristic of the expression $\mathfrak{U}_{\beta b}^{\rho QT_m \ldots T_1}$, which may be written in the form $\mathfrak{U}_{\beta b}^{\rho \xi}$ or $\mathfrak{U}^{\rho \xi}$, if the particular value of the pair β, b is irrelevant.

For each variable $X \in \bar{X}$ we define the norm $\|X\| = \min_{x \in \hat{X}} |x|$, where $|x|$ is the length of the word x in the language $\hat{X}$. The notion of norm is also extended to disjunctive polynomials: $\| \vee X_{i1} \ldots X_{ik_i} \| = \min_i \| X_{i1} \ldots X_{ik_i} \| = \min_i \Sigma_{j=1}^{k_i} \| X_{ij} \|$.

THEOREM 1. Let $\hat{X}_F^{q_0 Z_0}$ be the first component of the minimal solution $\hat{\bar{X}}$ of the system $\mathfrak{U}(\bar{X})$. Then $\hat{X}_F^{q_0 Z_0} = \mathfrak{L}(\mathfrak{U})$, where $\mathfrak{L}(\mathfrak{U})$ is the language accepted by the DPDA $\mathfrak{U}$.

Theorems on minimal solutions of disjunctive systems of equations are proved by well-known standard methods, and the proof is therefore omitted.

The following lemma provides a prefix property of the variables from $\bar{X}$.

LEMMA 1. For any variables $X_{\beta b}^{\alpha T}$ and $X_{\beta' b'}^{\alpha T}$, if $x \in \hat{X}_{\beta b}^{\alpha T}$, then there does not exist a word $y \in \Sigma^*$, $y \neq \varepsilon$, such that $xy \in \hat{X}_{\beta' b'}^{\alpha T}$.

The proof of Lemma 1 is identical with the proof of Lemma [20].1.

2. THE PROCEDURE Π, ε-RULES, AND MAIN RESULTS

The DPDA equivalence problem is solved by applying a comparison algorithm to a pair of DPDAs. The comparison algorithm constructs from the automata $\mathfrak{U}$ and $\mathfrak{B}$ a category system K whose elements are the categories $\mathfrak{C}_i$ $(i \geq 0)$ and the morphisms are functors between these categories. The system K is constructed by the procedure Π, which includes the operations of replacement, aggregation, decomposition, splitting, substitution, and reduction.

With each category $\mathfrak{C}_i$ $(i \geq 0)$ of the system K we associate the set R_i whose elements are the 4-tuples $(x, \mathfrak{U}, \mathfrak{W}, y)$, where $x, y \in \Sigma^*$ and $\mathfrak{U}, \mathfrak{W}$ is the pair of morphisms of the category $\mathfrak{C}_i$, in what follows it is called conditional equality.

The diagram of the category includes one arrow for each pair; the arrow is labeled by the pair and the equality itself is written in the form $\mathfrak{U} \doteq \mathfrak{W}$ (the symbol " $\doteq$ " stands for conditional equality, in distinction from the symbol "=" which denotes exact equality, as defined below). Since in most cases we are dealing with conditional equalities, the adjective "conditional" is omitted.

To each morphism of the given categories we can associate a language in the alphabet Σ represented by this morphism: to the morphism $\mathfrak{U}$ we associate the language $\hat{\mathfrak{U}}$, because this language is associated to each variable by definition. Two morphisms $\mathfrak{U}$ and $\mathfrak{W}$ are called equal (exact equality) if the languages $\hat{\mathfrak{U}}$ and $\hat{\mathfrak{W}}$ are identical.

Let us elucidate the meaning of the words x, y entering the 4-tuple $(x, \mathfrak{U}, \mathfrak{W}, y)$. Let $\mathfrak{U} = X_{\beta b}^{\alpha R}$, $\mathfrak{W} = Y_{\psi b}^{\varphi Q}$. Then for the automaton $\mathfrak{A}$ the word x defines the computation $(x, q_0, Z_0) \Rightarrow (\varepsilon, \alpha, Z_0\gamma R)$ and the word y defines the computation $(by, \beta, Z_0\gamma R) \Rightarrow (\varepsilon, F, \varnothing)$ (for the automaton $\mathfrak{B}$ these words respectively define the computations $(x, q_0, Z_0) \Rightarrow (\varepsilon, \varphi, Z_0\gamma Q)$ and $(by, \psi, Z_0\gamma Q) \Rightarrow (\varepsilon, F, \varnothing)$).

The operations of the procedure Π are applied to conditional equalities in R_i and from the category $\mathfrak{G}_i$ they construct the category $\mathfrak{G}_{i+1}$. Each equality to which an operation has been applied is marked used and no additional operations of the procedure Π are applied to this equality.

The morphisms of the categories of the system K are represented as disjunctive polynomials over the finite alphabet $\Sigma \cup \bar{X} \cup \bar{Y} \cup \bar{Z}$, where $\bar{X}$ is the set of variables of the system $\mathfrak{A}(\bar{X})$, $\bar{Y}$ is the set of variables of the system of equations $\mathfrak{B}(\bar{Y})$ defined by the automaton $\mathfrak{B}$, and $\bar{Z}$ is the alphabet of auxiliary variables constructed by the comparison algorithm. Each polynomial can be written in a compact form as the expression $\mathfrak{U}^{\alpha\xi}$, where α is an automaton state and ξ is the characteristic of an expression.

Let us consider the operations of the procedure Π. We define these operations for the construction of the first terms of the system K and then introduce the comparison algorithm.

The first element of the system K is the category $\mathfrak{G}_0$, containing only two objects, E_0 and E_F, and the pair of morphisms $H_{\mathfrak{G}_0}(E_0, E_F) = \{X_F^{q_0Z_0}, Y_F^{q_0Z_0}\}$ that generate a conditional equality to be included in R_0 in the form of the 4-tuple $(\varepsilon, X_F^{q_0Z_0}, Y_F^{q_0Z_0}, \varepsilon)$. The diagram of the category $\mathfrak{G}_0$ is

$$E_0 \xrightarrow{X_F^{q_0Z_0},\ Y_F^{q_0Z_0}} E_F .$$

Unit morphisms are not shown here and in what follows.

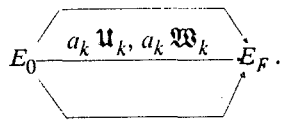
The replacement operation is applied in general to an equality of the form

$$\bigvee_i X_i \mathfrak{U}_i \doteq \bigvee_j Y_j \mathfrak{W}_j \tag{2.1}$$

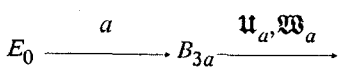
taking this equality to a new equality $\bigvee_i \left(\bigvee_k a_k X_{ik} \mathfrak{U}_{ik} \right) \mathfrak{U}_i \doteq \bigvee_j \left(\bigvee_l a_l Y_{jl} \mathfrak{W}_{jl} \right) \mathfrak{W}_j$, where the exact equality $X_i = \vee_k a_k X_{ik}$ is included in the system $\mathfrak{A}(\bar{X})$, and the exact equality $Y_j = \vee_l a_l Y_l \mathfrak{W}_{jl}$ is respectively included in the system $\mathfrak{B}(\bar{Y})$. The replacement operation thus replaces the first variable in the disjunctive expression in the original equality with the right-hand side of its equation from the system defined by the DPDA.

Let $X_F^{q_0Z_0} = \bigvee_{a \in \Sigma'} a \mathfrak{U}_a$, and $Y_F^{q_0Z_0} = \bigvee_{a \in \Sigma'} a \mathfrak{W}_a$ be the equations defining the variables $X_F^{q_0Z_0}$ and $Y_F^{q_0Z_0}$ in the systems $\mathfrak{A}(\bar{X})$ and $\mathfrak{B}(\bar{X})$, respectively. Applying the replacement operation we pass from the category $\mathfrak{G}_0$ to $\mathfrak{G}_1$, which includes new morphisms from $H_{\mathfrak{G}_1}(E_0, E_F)$: $\bigvee a\mathfrak{U}_a$, $\bigvee a\mathfrak{W}_a$, and $R_1 = R_0 \cup \left\{ \varepsilon, \bigvee_{a \in \Sigma'} a\mathfrak{U}_a, \bigvee_{a \in \Sigma'} a\mathfrak{W}_a, \varepsilon \right\}$. The equality $X_F^{q_0Z_0} \doteq Y_F^{q_0Z_0}$ is marked used. Note that the replacement operation does not affect the words (x, y) entering the corresponding 4-tuple.

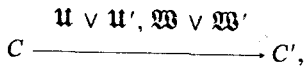
The aggregation operation takes us from the pair of morphisms $\left(\bigvee_{a\in\Sigma'} a\mathfrak{U}_a, \bigvee_{a\in\Sigma'} a\mathfrak{W}_a\right)$ to the set of pairs $\{\mathfrak{U}_a, \mathfrak{W}_a \mid a\in\Sigma'\}$ and includes in R_2 the new 4-tuples $\{(\varepsilon, a\,\mathfrak{U}_a, a\,\mathfrak{W}_a, \varepsilon) \mid a\in\Sigma'\}$ (the equality $\bigvee a\,\mathfrak{U}_a \doteq \bigvee a\,\mathfrak{W}_a$ is marked used). Thus, $R_2 = R_1 \cup \{(\varepsilon, a\,\mathfrak{U}_a, a\,\mathfrak{W}_a, \varepsilon) \mid a\in\Sigma'\}$. The diagram of the category $\mathfrak{G}_2$ (without the arrows included in the diagrams of the categories $\mathfrak{G}_0$ and $\mathfrak{G}_1$) for $\Sigma' = \{a_k \mid 1 \le k \le n\}$ has the form



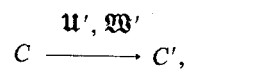
The decomposition operation takes us from the pair of equalities $\mathfrak{U} \doteq \mathfrak{W}$ and $\mathfrak{U}\,\mathfrak{U}' \doteq \mathfrak{W}\,\mathfrak{W}'$ to the pair $\mathfrak{U} \doteq \mathfrak{W}$ and $\mathfrak{U}' \doteq \mathfrak{W}'$ or from the pair of equalities $\mathfrak{U}' \doteq \mathfrak{W}'$ and $\mathfrak{U}\,\mathfrak{U}' \doteq \mathfrak{W}\,\mathfrak{W}'$ to the pair $\mathfrak{U}' \doteq \mathfrak{W}'$, $\mathfrak{U} \doteq \mathfrak{W}$ (the equality $\mathfrak{U}\,\mathfrak{U}' \doteq \mathfrak{W}\,\mathfrak{W}'$ is marked used). Applying the decomposition operation to the equality $a\,\mathfrak{U}_a \doteq a\,\mathfrak{W}_a$ which is a particular case of the equality $\mathfrak{U}\,\mathfrak{U}' \doteq \mathfrak{W}\mathfrak{W}'$ for $\mathfrak{U} \doteq \mathfrak{W} = a$, $\mathfrak{U}' = \mathfrak{U}_a$, $\mathfrak{W}' = \mathfrak{W}_a$, we obtain the equality $\mathfrak{U}_a \doteq \mathfrak{W}_a$. The diagram $R_3 = R_2 \cup \{(a, \mathfrak{U}_a, \mathfrak{W}_a, \varepsilon) \mid a\in\Sigma'\}$, of the category $\mathfrak{G}_3$ includes one new arrow for each $a \in \Sigma'$ and new objects B_{3a} for each pair $(\mathfrak{U}_a, \mathfrak{W}_a)$:



The splitting operation takes us from the pair of equalities $\mathfrak{U} \doteq \mathfrak{W}$ and $\mathfrak{U} \vee \mathfrak{U}' \doteq \mathfrak{W} \vee \mathfrak{W}'$ to the pair $\mathfrak{U} \doteq \mathfrak{W}$ and $\mathfrak{U}' \doteq \mathfrak{W}'$. If the diagram of the category $\mathfrak{G}_i$ includes the arrow



then for the pair of objects C, C' the new arrow is introduced in the category $\mathfrak{G}_{i+1}$,



and the equality $\mathfrak{U} \vee \mathfrak{U}' \doteq \mathfrak{W} \vee \mathfrak{W}'$ is marked used.

The substitution operation is applied to equalities of the form

$$\bigvee_{p=1}^{k} X_p\,\mathfrak{U}_p \doteq \mathfrak{W}^{\psi\xi}, \tag{2.2}$$

where $\mathfrak{W}^{\psi\xi}$ is the expression defined by the state ψ and the characteristic ξ and the expression in the left-hand side is nonlinear. We choose the shortest word from the language $\hat{X}_p$ associated to each variable X_p. We denote this word by $\langle X_p\rangle$ and call it the representative morphism. If there are several shortest words, any of them may be chosen as the representative morphism. Substitute $\langle X_p\rangle$ in equality (2.2). Since $\langle X_p\rangle \in \hat{X}_p$, using Lemma 1 we obtain the expression $\mathfrak{U}_p$. If we substitute the word $\langle X_p\rangle$ in the automaton $\mathfrak{B}$ in state ψ with the string ξ at the top of the pushdown store, then the automaton $\mathfrak{B}$ goes to some state ψ_p, a part of the string ξ is erased, and only the initial segment ξ_p remains, to which a new string η_p is adjoined. In other words, for each $\langle X_p\rangle$ we obtain the new equality

$$\mathfrak{U}_p \doteq \mathfrak{W}^{\psi_p\xi_p\eta_p}, \tag{2.3}$$

and on the whole we obtain a system of inequalities, which is called the direct system of the original quality (2.2). Note that no initial subword of the word $\langle X_p\rangle$ can erase the entire sequence ξ, because in this case the right-hand side would contain the

empty word with norm zero and the left-hand side would contain an expression of positive norm. The expression obtained by substituting the word $\langle X_p \rangle$ in $\mathfrak{W}_i$ will be denoted by the symbol $\int \langle X_p \rangle | \mathfrak{W}$. From (2.3) we thus have $\int \langle X_p \rangle | \mathfrak{W}^{\psi\xi} = \mathfrak{W}^{\psi_p \xi_p \eta_p}$.

Let $m = \min_p |\xi_p|$. From each ξ_p extract a string of symbols ζA of length m. Then each equality (2.3) may be written as

$$\mathfrak{U}^{p} \doteq \mathfrak{W}^{\psi_p \xi A \xi_p \eta_p}, \tag{2.4}$$

where $\zeta A \zeta_p = \xi_p$. But

$$\mathfrak{W}^{\psi_p \zeta A \zeta_p \eta_p} = \bigvee_{[A]} \mathfrak{W}^{\psi_p A \zeta_p \eta_p}_{[A]} \mathfrak{W}^{\mu[A]\zeta}. \tag{2.5}$$

Left-multiply each equality (2.4) by X_p and consider the disjunctive union over all p. We have

$$\bigvee_{p=1}^{k} X_p \mathfrak{U}_p \doteq \bigvee_{p=1}^{k} X_p \mathfrak{W}^{\psi_p \zeta A \zeta_p \eta_p} = \bigvee_{p=1}^{k} \bigvee_{[A]} X_p \mathfrak{W}^{\psi_p A \zeta_p \eta_p}_{[A]} \mathfrak{W}^{\mu[A]\zeta}, \tag{2.6}$$

but by (2.2) the left-hand side of (2.6) equals the expression $\mathfrak{W}^{\psi\xi}$, and $\mathfrak{W}^{\psi\xi} = \bigvee_{[A]} \mathfrak{W}^{\psi A \bar{\xi}}_{[A]} \mathfrak{W}^{\mu[A]\zeta}$, because ζA is an initial

substring of the string ξ. Thus,

$$\bigvee_{[A]} \bigvee_{p} X_p \mathfrak{W}^{\psi_p A \xi_p \eta_p}_{[A]} \mathfrak{W}^{\mu[A]\zeta} \doteq \bigvee_{[A]} \mathfrak{W}^{\psi A \bar{\xi}}_{[A]} \mathfrak{W}^{\mu[A]\zeta} \tag{2.7}$$

or, equating the expressions before equal $\mathfrak{W}^{\mu[A]\zeta}$,

$$\bigvee_{p} X_p \mathfrak{W}^{\psi_p A \zeta_p \eta_p}_{[A]} \doteq \mathfrak{W}^{\psi A \bar{\xi}}_{[A]}. \tag{2.8}$$

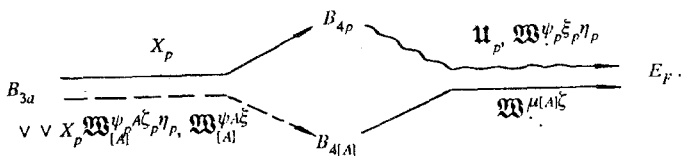
Equality (2.8) is written for each pair $[A]$ and the collection of these equalities for all pairs $[A]$ entering (2.7) is called the inverse system of equality (2.2) obtained by application of the substitution operation to (2.2).

Note that the equalities (2.8) are written assuming that all $\mathfrak{W}^{\mu[A]\zeta}$ are different. In fact, for different pairs $[A]_i \neq [A]_j$ we may have equal states $\mu[A]_i = \mu[A]_j$. Then clearly $\mathfrak{W}^{\mu[A]_i \zeta} = \mathfrak{W}^{\mu[A]_j \zeta}$. In this case, instead of the pair of equalities of the form (2.8) for $[A]_i$ and $[A]_j$, the inverse system will contain only one equality

$$\bigvee_{p} X_p (\mathfrak{W}^{\psi_p A \zeta_p \eta_p}_{[A]_i} \vee \mathfrak{W}^{\psi_p A \xi_p \eta_p}_{[A]_j}) = \mathfrak{W}^{\psi A \bar{\xi}}_{[A]_i} \vee \mathfrak{W}^{\psi A \bar{\xi}}_{[A]_j}. \tag{2.8.1}$$

If the 4-tuple $(x, \bigvee_p X_p \mathfrak{U}_p, \mathfrak{W}^{\psi\xi}, y)$, is linked with the equality (2.2) in R_i, then the passage to the category $\mathfrak{G}_{i+1}$ includes in R_{i+1} 4-tuple $(x <X_p>, \mathfrak{U}_p, \mathfrak{W}^{\psi_p \xi_p \eta_p}, y)$ for each equality of the direct system (2.3) and 4-tuple $(x, \vee X_p \mathfrak{W}^{\psi_p A \zeta_p \eta_p}_{[A]}, \mathfrak{W}^{\psi A \bar{\xi}}_{[A]}, y_{[A]} y)$ for each equality of the inverse system (2.8). Here $y_{[A]}$ is the shortest word in the language $\hat{\mathfrak{W}}^{\mu[A]\zeta}$.

If for these category constructions in the system K the equality (2.2) is identical with the equality $\mathfrak{U}_a \doteq \mathfrak{W}_a$ from R_3, then the element of the diagram of the category $\mathfrak{G}_4$, defined by the substitution operation has the form



We have thus defined all the operations of the procedure Π with the exception of the reduction operation, which will be introduced in the proof of Theorem 4. We now describe the main steps of the comparison algorithm for an arbitrary category $\mathfrak{G}_r$ and the set R_r.

1. Apply each successive operation of the procedure Π to all unmarked equalities in the set R_r.

2. The splitting operation is applied first, if possible, and only the equality subjected to splitting should be unmarked. If this equality corresponds to a wavy (broken) arrow, then the new equality after splitting also corresponds to a wavy (broken) arrow.

3. Apply the substitution operation to each equality defining a wavy arrow. For each equality of the inverse system, include a broken arrow in the diagram of the category $\mathfrak{G}_{r+1}$. For each equality $\mathfrak{U} \doteq \mathfrak{W}$ of the direct system include a wavy arrow if $\mathfrak{U}$ is nonlinear and a broken arrow if $\mathfrak{U} = X$.

4. If the reduction operation is applicable to an equality, then this equality is marked used and the new equality defined by the reduction operation is included in R_{r+1}.

5. To all unmarked equalities from R_r defining broken arrows apply the replacement operation followed by aggregation and then decomposition. To the resulting equalities associate wavy arrows if the left-hand side is nonlinear and broken arrows otherwise.

6. Include all the new equalities in R_{r+1} and augment the diagram of the category $\mathfrak{G}_r$ with the corresponding arrows and objects to the diagram of the category $\mathfrak{G}_{r+1}$.

7. Include additional equality marking rules and corresponding arrow replacement rules in the comparison algorithm. Let $\mathfrak{U} \doteq \mathfrak{W}$ be a given equality.

Rule ⟨r1⟩. The morphisms $\mathfrak{U}, \mathfrak{W}$ are written in the alphabet Σ and are equal as word sets.

Rule ⟨r2⟩. The equality $\mathfrak{U} \doteq \mathfrak{W}$, included in the set R_{r+1} is already contained in the set R_k ($k \leq r$).

In both cases, the equality $\mathfrak{U} \doteq \mathfrak{W}$ is marked used and the corresponding arrow in $\mathfrak{G}_{r+1}$ is replaced with a straight arrow.

The comparison algorithm stops if in some step r_0 all the rules in R_{r_0} have been marked or alternatively one of the negative conditions ⟨m1⟩, ⟨m2⟩ is satisfied:

⟨m1⟩: the morphisms $\mathfrak{U}, \mathfrak{W}$ are written in the alphabet Σ and are not equal as word sets, whereas $\mathfrak{U} \doteq \mathfrak{W} \in R_r$.

⟨m2⟩: $\|\mathfrak{U}\| \neq \|\mathfrak{W}\|$.

If conditions ⟨m1⟩ or ⟨m2⟩ are satisfied, there may exist a word which is contained in one of the languages $\hat{\mathfrak{U}}, \hat{\mathfrak{W}}$ and is not contained in the other language.

Regarding the rule ⟨r2⟩ we should note that, when comparing the automata $\mathfrak{A}$ and $\mathfrak{B}$, we seek the shortest word that proves their inequivalence, and therefore the first inclusion of the equality $\mathfrak{U} \doteq \mathfrak{W}$ in R_k produces a shorter word than the word produced by the second inclusion of the same equality in the set R_r with $r > k$.

This comparison algorithm has been applied in [20] to prove decidability of the equivalence problem for DPDA without ε-rules. In this case, the norm of each variable from the alphabet $\bar{X} \cup \bar{Y}$ is greater than zero, which simplifies all the estimates for the morphisms of the categories $\mathfrak{G}_i$ ($i \geq 0$). The introduction of ε-rules alters the situation, however: the empty word ε may erase from the pushdown store whole sequences of pushdown symbols (the norm of the corresponding variables is zero), and ε-rules therefore require a special analysis.

For the DPDA $\mathfrak{A}$, the pushdown symbol $A \in \Gamma$ is called a special symbol if there exist a state α and a positive number n such that the automaton $\mathfrak{A}$, starting from the state α with the string γA^n in the pushdown store is driven by the empty word to the state α with the string γ in the pushdown store.

The sequence of symbols $A_1A_2\ldots A_k$ is called a special sequence if the computation $(\alpha, \gamma A_1\ldots A_k) \Rightarrow (\alpha, \gamma)$ exists. Each symbol A_i ($1 \leq i \leq k$) is also called special.

The sequence of symbols $B_1\ldots B_m$ in the pushdown alphabet Γ is called weak if (a) it is not a subsequence of any special sequence, does not contain a special sequence as a subsequence, and there exist states $\{\alpha_i, \alpha_j \in S, \alpha_i \neq \alpha_j\}$ of the automaton $\mathfrak{A}$ such that the empty word takes the automaton from state α_i to state α_j erasing the string $B_1\ldots B_m$ from the pushdown store and (b) no weak sequence exists that includes $B_1\ldots B_m$ as a subsequence.

LEMMA 2. All weak and special sequences can be found for any DPDA $\mathfrak{A}$.

Lemma 2 is proved in Sec. 3.

The maximum length of weak sequences is obviously bounded by the number of automaton states. At the same time, arbitrarily long sequences can be constructed from special symbols (we call them ε-sequences in what follows).

For notational simplicity, we first assume that there exist a single special symbol A and a pair of rules erasing this symbol: $(\varepsilon, \psi, A) \to (\psi, \varepsilon)$ and $(a, \theta, A) \to (\omega, \varepsilon)$. We need to establish if the construction of the category system K may produce expressions whose characteristics contain arbitrarily long ε-sequences. To this end, we will analyze all the operations of the procedure Π. First consider the replacement operation. Let the following expression be given:

$$\mathfrak{W}_{\chi a}^{\varphi A_m \ldots A_1} = \mathfrak{W}_{\chi a}^{\varphi A_m \ldots A_{r+1} A^r} = Y_{\psi\varepsilon}^{\varphi A} \mathfrak{W}_{\chi a}^{\psi A_m \ldots A_{r+1} A^{r-1}} \vee Y_{\theta a}^{\varphi A} \mathfrak{W}_{\chi a}^{\omega A_m \ldots A^{r-1}}. \tag{2.9}$$

THEOREM 2. For the special symbol A and the set of rules $W^A = \{(\varepsilon, \psi, A) \to (\psi, \varepsilon), (a, \theta, A) \to (\omega, \varepsilon)\}$ for any $r > 1$ we have the exact equality

$$Y_{\psi\varepsilon}^{\varphi A} \mathfrak{W}_{\chi a}^{\psi\eta A^{r-1}} = Y_{\psi\varepsilon}^{\varphi A} \mathfrak{W}_{\chi a}^{\psi\eta}. \tag{2.10}$$

Theorem 2 is proved in Sec. 3.

If the DPDA contains a rule that adjoins special symbols, e.g., $(a, \varphi, A) \to (\varphi, A^k)$, then the equation for $Y^{\varphi A}$ has the form $Y^{\varphi A} = a\mathfrak{W}^{\varphi A^k} \vee \ldots$, where the ellipsis denotes other disjunctive terms. Then replacement of $Y_{\psi\varepsilon}^{\varphi A^k}$, $Y_{\theta a}^{\varphi A}$ in (2.9) and a-aggregation give the expression $\mathfrak{W}_{\psi\varepsilon}^{\varphi A^k} \cdot \mathfrak{W}_{\chi a}^{\psi A_m \ldots A_{r+1} A^r} \vee \mathfrak{W}_{\theta a}^{\varphi A^k} \cdot \mathfrak{W}_{\chi a}^{\psi A_m \ldots A^{r-1}} = \mathfrak{W}_{\chi a}^{\varphi A_m \ldots A_{r+1} A^{r-1+k}}$, i.e., the ε-sequence becomes longer after the application of the replacement operation. At the same time note that if expressions $Y_{\psi\varepsilon}^{\varphi A} \mathfrak{W}_{\chi a}^{\psi A_m \ldots A^{r-1}}$ and $Y_{\theta a}^{\varphi A} \mathfrak{W}_{\chi a}^{\omega A_m \ldots A^{r-1}}$ are considered separately, then the replacement operation in each of these expressions does not increase the length of the special sequence: in the first case by Theorem 2 and in the second case because the variable $Y_{\theta a}^{\varphi A}$ includes the symbol $a \neq \varepsilon$, which is not eliminated by the replacement operation.

Consider the substitution operation and its possible effect on lengthening the ε-sequence. Let

$$\sigma = \max \{\max_{\bar{X}} |\langle X_{\beta a}^{\alpha R}\rangle|, \max_{\bar{Y}} |\langle Y_{\theta b}^{\psi a}\rangle|\}, \tag{2.11}$$

where $\max_{\bar{X}}$ and $\max_{\bar{Y}}$ are over all the variables entering the systems of equations $\mathfrak{A}(\bar{X})$ and $\mathfrak{B}(\bar{Y})$, respectively. Denote by λ^0 the largest of the numbers $\lambda_{\mathfrak{A}}, \lambda_{\mathfrak{B}}$, where $\lambda_{\mathfrak{A}}$ is expressed in terms of the lengths of the sequences $T_1 \ldots T_m$ determined by the rules of the automaton $\mathfrak{A}$, that are included in the set $U^{\alpha R}$ for all pairs (α, R):

$$\lambda_{\mathfrak{A}} = \max_{a, \alpha, R} \{m + 1 \mid (a, \alpha, R) \to (\beta, RT_m \ldots T_1) \in U^{\alpha R}\}. \tag{2.12}$$

The number $\lambda_{\mathfrak{B}}$ is similarly defined. Then, assuming that rules of the form $(a, \alpha, A) \to (\bar{\alpha}, A^\lambda)$ $(\lambda \leq \lambda^0)$ exist and that each letter of the substituted word x (by assumption, $|x| \leq \sigma$) invokes application of this ε-sequence lengthening rule, we conclude that the overall lengthening after the substitution of the word x does not exceed $\sigma(\lambda^0 - 1)$. We thus have the following theorem.

THEOREM 3. The substitution of a representative morphism increases the length of the ε-sequence by not more than $\sigma(\lambda^0 - 1)$.

The following theorem restricts the length of the ε-sequences that may enter the characteristics of the expressions defining the equalities in the sets R_i, $i > 0$.

THEOREM 4. The length of the ε-sequences entering the characteristic of the expression $\mathfrak{U}, \mathfrak{W}$ for any conditional equality $\mathfrak{U} \doteq \mathfrak{W}$ from the set R_i $(i > 0)$ is bounded by $\mu_0 < \infty$.

In the proof of Theorem 4 (Sec. 3) we introduce the reduction operation, which makes it possible to ignore equalities with very long ε-sequences.

Before stating the main theorems, we introduce the notion of correct operation. An operation of the procedure Π is called correct if its application to any equality $\mathfrak{U} \doteq \mathfrak{W}$ produces equalities $\mathfrak{U}_i \doteq \mathfrak{W}_i$ $(1 \leq i \leq n)$ that satisfy the following condition: if the languages $\hat{\mathfrak{U}}, \hat{\mathfrak{W}}$ satisfy the equality $\hat{\mathfrak{U}} = \hat{\mathfrak{W}}$, then $\hat{\mathfrak{U}}_i = \hat{\mathfrak{W}}_i$ for all $i \in [1, n]$; if $\hat{\mathfrak{U}} \neq \hat{\mathfrak{W}}$ then there exists at least one index $i_0 \in [1, n]$ such that $\hat{\mathfrak{U}}_{i_0} \neq \hat{\mathfrak{W}}_{i_0}$.

The passage from the category $\mathfrak{G}_i$ to $\mathfrak{G}_{i+1}$ is called correct if only correct operations are applied to the equalities during the construction of the set R_{i+1}.

THEOREM 5. The operations of replacement, aggregation, decomposition, splitting, and reduction in the procedure II are correct.

THEOREM 6 (first main theorem) Let $\mathfrak{A}$ and $\mathfrak{B}$ be two deterministic pushdown automata and K a category system constructed by the comparison algorithm from the category $\mathfrak{G}_0$. If the passage from the category $\mathfrak{G}_i$ to $\mathfrak{G}_{i+1}$ during the construction of K is correct for all $i \geq 0$, then the equivalence problem for the automata $\mathfrak{A}$ and $\mathfrak{B}$ is decidable.

By Theorem 5, the substitution operation may be the only incorrect operation in the procedure II. This requires a modification of the comparison algorithm. The modified algorithm constructs a finite sequence of category systems, and the analysis of these systems solves the DPDA equivalence problem, producing the main result of this paper.

THEOREM 7 (second main theorem). The equivalence problem for deterministic pushdown automata is decidable.

The proof of Theorem 7 is given in Sec. 4.

Corollaries of Theorem 7 provide solutions of some hitherto open problems relating to the DPDA equivalence problem.

COROLLARY 1. The functional equivalence problem for unary recursive program schemes (de Bakker–Scott schemes) is decidable.

The result follows from Theorem 7 and [17].

COROLLARY 2. The equivalence problem for recursive program schemes without constant functions is decidable.

This corollary solves the problem posed in [18].

COROLLARY 3. The problem "decide if a deterministic language is an $LR(0)$-language" is decidable.

This solves the problem posed in [19].

COROLLARY 4. The equivalence problem for $LR(0)$-grammars and $LR(1)$-grammars is decidable [15].

COROLLARY 5. The problem of functional equivalence for $LA(1)$-recognizers is decidable.

This follows from Theorem 7 and [16].

A number of problems considered in [14] are reducible to the DPDA equivalence problem, and these problems are therefore also decidable.

3. PROOF OF AUXILIARY PROPOSITIONS

We now proceed to prove some intermediate results that are used in Sec. 4.

Proof of Lemma 2. Let $W_\varepsilon^{\mathfrak{A}}$ be the set of all ε-rules of the ε-reduced automaton $\mathfrak{A}$, $W_\varepsilon^{\mathfrak{A}} \subseteq \bigcup_{R \in \Gamma} W^R$. The set $W_\varepsilon^{\mathfrak{A}}$ is finite. Construct the ε-graph $\mathfrak{G}_{\mathfrak{A}}$ of the automaton $\mathfrak{A}$ in the following way. Mark the graph vertices by the automaton states; join the vertices α and β by arcs directed from α to β and labeled by symbols from the pushdown alphabet $P \subseteq \Gamma$ if and only if the set $W_\varepsilon^{\mathfrak{A}}$ contains rules of the form $(\varepsilon, \alpha, P) \to (\beta, \varepsilon)$. For each state α_0 we write out all the state sequences $\alpha_0, \alpha_1, \ldots, \alpha_k$ such that for each pair (α_p, α_{p+1}) in $\mathfrak{G}_{\mathfrak{A}}$ there are arrows from α_p to α_{p+1}, i.e., we write out all the paths that lead to graph vertices from the vertex α_0 and satisfy the following conditions. First, the length of any path $\alpha_0 \ldots \alpha_k$ does not exceed the number of states of the automaton $\mathfrak{A}$, second, at most one vertex may occur twice in any sequence of vertices forming a path. To each path $\alpha_0 \ldots \alpha_q$ we assign the name $[A_1 \ldots A_k]$, where A_p is the label of the arc joining the vertices α_{p-1} and α_p. The set of named paths constructed in this way is denoted $\bar{S}_{\mathfrak{A}}^{\varepsilon}$. All paths included in $\bar{S}_{\mathfrak{A}}^{\varepsilon}$, are partitioned into two subsets, $\bar{S}_1$ and $\bar{S}_2$. The first subset contains all the paths for which $\alpha_p \neq \alpha_q$, $p \neq q$, $p, q \in [0, k]$; the second subset contains the paths for which $\alpha_0 = \alpha_k$. Remove from $\bar{S}_1$ all the paths that contain as subpaths pieces of paths from $\bar{S}_2$ and then remove all the paths that are subpaths of other paths in $\bar{S}_1$. The result is a finite set S_1. Each path in S_1 defines a weak sequence corresponding to its name, and the set of these names exhausts the set of weak sequences.

Each path included in $\bar{S}_2$ defines a subgraph of the graph $\mathfrak{G}_{\mathfrak{A}}$, that forms an elementary circuit, and the regular event defined by these paths contains all the special sequences generated by the automaton $\mathfrak{A}$. Q.E.D.

Proof of Theorem 2. In the expression $\mathfrak{W}_{\chi a}^{\psi\eta A^{r-1}}$ entering the left-hand side of the exact equality (2.10), separate the first variable $\mathfrak{W}_{\chi a}^{\psi\eta A^{r-1}} = Y_{\psi\varepsilon}^{\psi A} \mathfrak{W}_{\chi a}^{\psi\eta A^{r-2}} \vee Y_{\theta a}^{\psi A} \mathfrak{W}_{\chi a}^{\omega\eta A^{r-2}}$. But $Y_{\psi\varepsilon}^{\psi A} = \varepsilon$ and $Y_{\theta a}^{\psi A} = \varnothing$, because for the pair (ψ, A) there

is a unique rule $(\varepsilon, \psi, A) \to (\psi, \varepsilon)$ and, thus, $\mathfrak{W}_{\chi a}^{\psi\eta A^{r-2}} = \mathfrak{W}_{\chi a}^{\psi\eta A^{r-1}}$. Applying the same argument $(r-2)$ more times, we obtain formula (2.10). Q.E.D.

We give two lemmas from [20] without proof.

LEMMA [20].12. Successive application of replacement and aggregation operations may increase the norm of the morphism by not more than $\sigma\lambda^0 - 1$.

LEMMA [20].13. The substitution operation may increase the norm of the morphism by not more than $\sigma^2\lambda^0$.

Note that for the modified comparison algorithm these results must be sharpened.

LEMMA 3. The length of the characteristic of the expression obtained by substitution of a representative morphism is reduced by not more than $(\mu^0 + 1)(\sigma + 1) - 1$.

Lemma 3 is proved under the assumption that Theorem 4 holds. During substitution, the maximum reduction of the characteristic of an expression can be achieved by erasing a sequence of pushdown symbols of the form $A^{t0}A_1A^{t1}A_2\ldots A_\sigma A^{t\sigma}$, where $A^{t0}, A^{t1}, \ldots, A^{t\sigma}$ are sequences of special symbols that are erased by the empty word. The length of the entire sequence is $t = \Sigma_{i=0}^{\sigma} t_i + \sigma$. Then by Theorem 4, $t \leq \Sigma_0^{\sigma} \mu_0 + \sigma = (\sigma + 1)\mu_0 + \sigma = (\sigma + 1)(\mu_0 + 1) - 1$. Q.E.D.

LEMMA 4. If substitution transforms equality $\bigvee_{l \in J} X_l \mathfrak{U}_l \doteq \mathfrak{W}^{\varphi\eta_1 B\eta_2}$ to direct system of equalities $\mathfrak{U}_l = \mathfrak{W}^{\varphi_l\eta_1 B\xi_l}$,

then the following inequalities hold for any $l \in J$: $\max_{l \in J} |\xi_l| \leq (\mu_0 + \lambda^0)(\sigma + 1) - \lambda^0$.

Indeed, the maximum length of the sequence ξ_l can be obtained when (a) the substitution of a word first erases the largest possible piece of the sequence η_2 (by Lemma 3 the length of this piece does not exceed $(\mu_0 + 1)(\sigma + 1) - 1$) and (b) the substitution of another representative morphism adjoins the longest possible sequence of pushdown symbols to the sequence η_2 (by Lemma [20].13, the length of this sequence does not exceed $\sigma(\lambda^0 - 1)$). Hence we have

$$\max|\xi_l| \leq (\mu_0 + 1)(\sigma + 1) - 1 + \sigma(\lambda^0 - 1) = (\mu_0 + \lambda^0)(\sigma + 1) - \lambda^0.$$

Q.E.D.

To prove Theorem 4, we will need a number of lemmas from [20], which are reproduced below.

LEMMA [20].4. Let χ be the following system of equalities included in R_i: $\chi = (X\mathfrak{U} \doteq Y\mathfrak{W}, X \doteq Y)$. Then the passage from χ to the system $\hat{\chi} = (\mathfrak{U} \doteq \mathfrak{W}, X \doteq Y)$, included in R_{i+1} is a correct operation.

LEMMA [20].5. Let the equalities of the system $\chi = (\mathfrak{U} \vee \mathfrak{U}' \doteq \mathfrak{W} \vee \mathfrak{W}', \mathfrak{U}' \doteq \mathfrak{W}')$ be included in the set R_1 and $\mathfrak{U} \cap \mathfrak{U}' = \varnothing$, $\mathfrak{W} \cap \mathfrak{W}' = \varnothing$. Then inclusion of the system $\hat{\chi} = (\mathfrak{U} \doteq \mathfrak{W}, \mathfrak{U}' \doteq \mathfrak{W}')$ in R_{i+1} is a correct operation.

LEMMA [20].7. $\hat{X}_{[Q]_i}^{\alpha Q} \cap \hat{X}_{[Q]_j}^{\alpha Q} = \varnothing$ for $i \neq j$.

LEMMA [20].9. The equality $\mathfrak{W}^{\psi_\rho\zeta A\zeta_\rho\eta_\rho} = \bigvee_{[A]} \mathfrak{W}_{[A]}^{\psi_\rho A\zeta_\rho\eta_\rho}\mathfrak{W}^{\mu[A]\zeta}$ is an exact equality.

Proof of Theorem 4. This proof is divided into two parts. First assume that all special sequences are generated by a single special symbol A and the set of rules W^A of the automaton $\mathfrak{B}$ contains only three rules: $\{(a, \varphi, A) \to (\varphi, A^\delta), (\varepsilon, \psi, A) \to (\psi, \varepsilon), (a, \theta, A) \to (\omega, \varepsilon)\}$.

Consider the application of the comparison algorithm to the pair of DPDAs $\mathfrak{A}$ and $\mathfrak{B}$, We will try to define these automata so that the comparison algorithm produces an infinitely increasing ε-sequence, in contradiction with the assumption of the theorem.

The length of the ε-sequence may be increased by replacement and aggregation and by substitution. Indeed, if the system $\mathfrak{B}$ $(\bar{Y})$ contains the equation $Y^{\varphi A} = a\mathfrak{W}^{\varphi A^\delta} \vee \tilde{Y}^{\varphi A}(a)$, where $\tilde{Y}^{\varphi A}(a)$ denotes all the terms of the disjunctive polynomial defining the morphism $Y^{\varphi A}$ that start with symbols of the alphabet Σ not equal to a, then replacement of the variable $Y^{\varphi A}$ in the expression $\mathfrak{W}_{\varphi}^{\varphi\xi A^{\tau+1}}$ and a-aggregation produce equality $\mathfrak{W}_{\psi\varepsilon}^{\varphi A^\delta}\mathfrak{W}_{\varphi}^{\psi\xi A^\tau} \vee \mathfrak{W}_{\theta a}^{\varphi A^\tau}\mathfrak{W}_{\varphi}^{\omega\xi A^\tau} = \mathfrak{W}_{\varphi}^{\varphi\xi A^{\tau+\delta}}$. Thus, substitution increases the length of the ε-sequence of the symbols A by $\delta - 1 \leq \lambda^0 - 1$.

Substitution of the word $x = a^n x'$ in the expression $\mathfrak{W}_{\varphi}^{\varphi\,\xi\,A^{\tau+1}}$ leads to the following expression:

$$\int x \mid \mathfrak{W}_{\varphi}^{\varphi\,\xi\,A^{\tau+1}} = \int x' \mid (\int a^n \mid \mathfrak{W}_{\varphi}^{\varphi\,\xi\,A^{\tau+1}}) = \int x' \mid \mathfrak{W}_{\varphi}^{\varphi\,\xi\,A^{n(\delta-1)+\tau+1}}.$$

If we assume that substitution of the word x' does not shorten the resulting ε-sequence, then the overall increase of its length is by $n(\delta - 1) \leq n(\lambda^0 - 1)$.

This argument shows that the application of different operations of the procedure Π in the comparison algorithm may increase the length of the ε-sequences. We are thus faced with the following question: is it possible to construct DPDAs for which this length increase is unbounded, or at least exceeds any prespecified number? We will show that the answer to this question is negative.

Suppose that the comparison algorithm has produced in R_i the equality

$$\mathfrak{U}^{i\eta Q} \doteq \mathfrak{W}^{\varphi\,\xi\,A^{\tau_0}\xi'}, \tag{3.1}$$

which may be represented in the form

$$X_{[Q]_1}^{iQ}\,\mathfrak{U}^{\mu[Q]_1\eta} \vee X_{[Q]_2}^{iQ}\,\mathfrak{U}^{\mu[Q]_2\eta} \doteq \mathfrak{W}^{\varphi\,\xi\,A^{\tau_0}\xi'} \tag{3.2}$$

assuming that only two variables $X_{[Q]_l}^{iQ}$ $(l = 1, 2)$ exist. Applying the substitution operation to (3.2), we obtain the direct system $\mathfrak{U}^{\mu[Q]_l\eta} \doteq \int \langle X_{[Q]_l}^{iQ} \rangle \mid \mathfrak{W}^{\varphi\,\xi\,A^{\tau_0}\xi'}$. If we assume that $\langle X_{[Q]_l}^{iQ} \rangle = x_{l_1} v_l x_{l_2}$, where the word x_{l_1} erases ξ', the word v_l increases the ε-sequence, and the word x_{l_2} adjoins the string ξ_l after the ε-sequence, then

$$\mathfrak{U}^{\mu[Q]_l\eta} \doteq \mathfrak{W}^{\varphi_l\,\xi\,A^{\tau_l}\xi_l} \quad (l = 1, 2;\ \tau_l \geq \tau_0). \tag{3.3}$$

Hence we obtain the inverse system

$$X_{[Q]_1}^{iQ}\,\mathfrak{W}_{[A]}^{\varphi_1 A^{\tau_1 - \tau_0 + 1}\xi_1} \vee X_{[Q]_2}^{iQ}\,\mathfrak{W}_{[A]}^{\varphi_2 A^{\tau_2 - \tau_0 + 1}\xi_2} \doteq \mathfrak{W}_{[A]}^{\varphi A \xi_1}. \tag{3.4}$$

One application of the replacement and aggregation operations to (3.4) may increase the length of the ε-sequence from the right by at most λ_0, after which the previous argument is repeated. The ε-sequences are not lengthened from the left either, because the replacement and aggregation operations are applied to the variables $X_{[Q]_l}^{iQ}$. The ε-sequences can be lengthened only by equalities of the form (3.3), because some substitutions increase the length of the ε-sequence in each step. Applying the substitution operation, we finally obtain from (3.3) the general equality

$$X \doteq \mathfrak{W}^{\varphi\,\xi\,A^{\tau}\xi'B}. \tag{3.5}$$

Then the assumption of unbounded growth of the ε-sequence A^{τ} (recall that there exists a single special symbol A) implies that after replacing the variable X by its expression from the system $\mathfrak{A}(\bar{X})$ and successively executing not more than $\lambda_0 - 1$ substitutions we again obtain equality (3.5), but with the right-hand side containing the ε-sequence A^{τ_1} $(\tau_1 > \tau_0)$. Thus, the operations of the procedure Π may increase without bound the length of the ε-sequence.

Applying substitutions in equality (3.5), we may obtain different variables in the left-hand side. However, since the alphabet $\bar{X}$ is finite, the same variable will eventually recur after finitely many steps, so that without loss of generality we may assume that the equation for X in the system $\mathfrak{A}(X)$ has the form $X = \bigvee_{a \in \Sigma} a\mathfrak{U}(a)\,X(a)$ and for some $a_0 \in \Sigma$ the variable $X_{a_0} = X$. Thus, it is equality (3.5) that is responsible for the construction of unbounded ε-sequences.

Let us construct finite word sets $\hat{\mathfrak{X}} = \{\mathfrak{X}_a \mid a \in \Sigma\}$ and $\mathfrak{X}_a = \langle \mathfrak{U}(a) \rangle$, where the symbol $\langle \mathfrak{U}(a) \rangle$ stands for the shortest word that consists of concatenation of the words $\langle X_i \rangle$ for all variables in $\mathfrak{U}(a)$. Clearly, $\max_{a \in \Sigma} |\mathfrak{X}_a| \leq (\lambda^0 - 1)\sigma$.

Separate from the set $\hat{\mathfrak{x}}$ those words $\mathfrak{x}_a$, that increase the length of the ε-sequence, i.e., the words whose substitution produces the expression $\int a\,\mathfrak{x}_a \mid \mathfrak{W}^{\varphi\,\xi A\,\xi'} = \mathfrak{W}^{\varphi\,\xi\,A^{\tau'}\,\xi'\,B'}$.

During substitution, the procedure first erases the symbols of the sequence $\xi' B$, then the ε-sequence is increased, and finally, the sequence $\xi'' B'$ is adjoined. Since the length of $\mathfrak{x}_a$ does not exceed $\sigma(\lambda^0 - 1)$, the length of the sequence $\xi'' B'$ is bounded and does not exceed $\sigma(\lambda^0 - 1)^2$. Since by assumption there exists an arbitrarily long sequence of equalities of the form $X \doteq \mathfrak{W}^{\varphi\,\xi\,A^{\tau_p}\eta_p}$, with a monotone increasing sequence of exponents τ_p, there exists a pair of indices p' and p'' for which $\eta_{p'} = \eta_{p''}$. Therefore, without loss of generality we may assume that the substitution of the word $a\,\mathfrak{x}_a$ in expression $\mathfrak{W}^{\varphi\,\xi\,A^{\tau}\,\xi'\,B}$ produces the expression $\mathfrak{W}^{\varphi\,\xi\,A^{\tau'}\,\xi'\,B}$. As a result, we obtain the new equality

$$X \doteq \mathfrak{W}^{\varphi\,\xi\,A^{\tau'}\,\xi'\,B}. \tag{3.6}$$

From equalities (3.5) and (3.6) we obtain

$$\mathfrak{W}^{\varphi\xi A^{\tau}\xi' B} \doteq \mathfrak{W}^{\varphi\xi A^{\tau'}\xi' B}. \tag{3.7}$$

Denote by $[\xi']$ the pair (state, pushdown symbol) associated to the last letter of the string ξ'. Then, twice applying Lemma [20].9 to (3.7), we obtain

$$\begin{aligned} &\vee_{[\xi']}\mathfrak{W}^{\xi' B}_{[\xi']}\,(\,\mathfrak{W}^{\mu[\xi']\,A^{\tau}}_{\psi\,\varepsilon}\,\mathfrak{W}^{\psi\,\xi}_{\cdot} \vee \mathfrak{W}^{\mu[\xi']\,A^{\tau}}_{\theta\,a}\,\mathfrak{W}^{\omega\,\xi}_{\cdot}) \doteq \\ &\doteq \vee_{[\xi']}\mathfrak{W}^{\varphi\xi' B}_{[\xi']}\,(\,\mathfrak{W}^{\mu[\xi']\,A^{\tau'}}_{\psi\,\varepsilon}\,\mathfrak{W}^{\psi\,\xi}_{\cdot} \vee \mathfrak{W}^{\mu[\xi']\,A^{\tau'}}_{\theta\,a}\,\mathfrak{W}^{\omega\,\xi}_{\cdot}). \end{aligned} \tag{3.8}$$

Applying Lemmas 1 and [20].7 to equality (3.8), we obtain for each state $\mu[\xi']$

$$\begin{aligned} &\mathfrak{W}^{\mu[\xi']\,A^{\tau}}_{\psi\,\varepsilon}\,\mathfrak{W}^{\psi\,\xi}_{\cdot} \vee \mathfrak{W}^{\mu[\xi']\,A^{\tau}}_{\theta\,a}\,\mathfrak{W}^{\omega\,\xi}_{\cdot} \doteq \\ &\doteq \mathfrak{W}^{\mu[\xi']\,A^{\tau'}}_{\psi\,\varepsilon}\,\mathfrak{W}^{\psi\,\xi}_{\cdot} \vee \mathfrak{W}^{\mu[\xi']\,A^{\tau'}}_{\theta\,a}\,\mathfrak{W}^{\omega\,\xi} \end{aligned} \tag{3.9}$$

or, separating the first variable from the left and from the right,

$$\begin{aligned} &(\,Y^{\mu[\xi']A}_{\psi\varepsilon}\,\mathfrak{W}^{\psi A^{\tau-1}}_{\psi\varepsilon} \vee Y^{\mu[\xi']A}_{\theta\,a}\,\mathfrak{W}^{\omega A^{\tau-1}}_{\psi\varepsilon})\mathfrak{W}^{\psi\xi}_{\cdot} \vee (\,Y^{\mu[\xi']A}_{\psi\varepsilon}\,\mathfrak{W}^{\psi A^{\tau-1}}_{\theta a} \vee \\ &\vee\, Y^{\mu[\xi']A}_{\theta\,a}\,\mathfrak{W}^{\omega A^{\tau-1}}_{\theta a})\mathfrak{W}^{\omega\,\xi}_{\cdot} \doteq (\,Y^{\mu[\xi']A}_{\psi\varepsilon}\,\mathfrak{W}^{\psi A^{\tau'-1}}_{\psi\varepsilon} \vee Y^{\mu[\xi']A}_{\theta\,a}\,\mathfrak{W}^{\omega A^{\tau'-1}}_{\psi\varepsilon}\,) \times \\ &\times \mathfrak{W}^{\psi\xi}_{\cdot} \vee (\,Y^{\mu[\xi']A}_{\psi\varepsilon}\,\mathfrak{W}^{\psi A^{\tau'-1}}_{\theta a} \vee Y^{\mu[\xi']A}_{\theta\,a}\,\mathfrak{W}^{\omega A^{\tau'-1}}_{\theta a})\mathfrak{W}^{\omega\,\xi}_{\cdot}. \end{aligned} \tag{3.10}$$

But $\hat{\mathfrak{W}}^{\psi A^{\tau}}_{\psi\varepsilon} = \varepsilon$ and $\hat{\mathfrak{W}}^{\psi A^{\tau}}_{\theta a} = \varnothing$. Then from (3.10) we obtain

$$\begin{aligned} &Y^{\mu[\xi']A}_{\theta a}(\,\mathfrak{W}^{\omega A^{\tau-1}}_{\psi\,\varepsilon}\,\mathfrak{W}^{\psi\,\xi}_{\cdot} \vee \mathfrak{W}^{\omega A^{\tau-1}}_{\theta\,a}\,\mathfrak{W}^{\omega\,\xi}_{\cdot}) \doteq \\ &\doteq Y^{\mu[\xi']A}_{\theta a}(\,\mathfrak{W}^{\omega A^{\tau'-1}}_{\psi\,\varepsilon}\,\mathfrak{W}^{\psi\,\xi}_{\cdot} \vee \mathfrak{W}^{\omega A^{\tau'-1}}_{\theta\,a}\,\mathfrak{W}^{\omega\,\xi}_{\cdot}), \end{aligned} \tag{3.11}$$

and since $\tau' > \tau$, Lemma [20].9 leads to the equality

$$\begin{aligned} \mathfrak{W}^{\omega A^{\tau-1}}_{\psi\,\varepsilon}\,\mathfrak{W}^{\psi\,\xi}_{\cdot} \vee \mathfrak{W}^{\omega A^{\tau-1}}_{\theta\,a}\,\mathfrak{W}^{\omega\,\xi}_{\cdot} &= (\,\mathfrak{W}^{\omega A^{\tau-1}}_{\psi\,\varepsilon}\,\mathfrak{W}^{\psi A^{\tau'-\tau}}_{\psi\varepsilon} \vee \mathfrak{W}^{\omega A^{\tau-1}}_{\theta\,a}\,\mathfrak{W}^{\omega A^{\tau'-\tau}}_{\psi\varepsilon}) \times \\ &\times \mathfrak{W}^{\psi\,\xi}_{\cdot} \vee (\,\mathfrak{W}^{\omega A^{\tau-1}}_{\theta a}\,\mathfrak{W}^{\omega A^{\tau'-\tau}}_{\theta a} \vee \mathfrak{W}^{\omega A^{\tau-1}}_{\psi\varepsilon}\,\mathfrak{W}^{\psi A^{\tau'-\tau}}_{\theta a})\mathfrak{W}^{\omega\,\xi}_{\cdot}, \end{aligned}$$

whence

$$\mathfrak{W}_{\psi\varepsilon}^{\omega A^{\tau-1}}\mathfrak{W}_{\cdot}^{\psi\xi} \vee \mathfrak{W}_{\theta a}^{\omega A^{\tau-1}}\mathfrak{W}_{\cdot}^{\omega\xi} \doteq \mathfrak{W}_{\psi\varepsilon}^{\omega A^{\tau-1}}\mathfrak{W}_{\cdot}^{\psi\xi} \vee$$
$$\vee \mathfrak{W}_{\theta a}^{\omega A^{\tau-1}}(\mathfrak{W}_{\psi\varepsilon}^{\omega A^{\tau'-\tau}}\mathfrak{W}_{\cdot}^{\psi\xi} \vee \mathfrak{W}_{\theta a}^{\omega A^{\tau'-\tau}}\mathfrak{W}_{\cdot}^{\omega\xi}).$$

Omitting equal disjunctive terms (Lemma [20].5) and applying Lemma [20].4, we obtain

$$\mathfrak{W}_{\cdot}^{\omega\xi} \doteq \mathfrak{W}_{\psi\varepsilon}^{\omega A^{\tau'-\tau}}\mathfrak{W}_{\cdot}^{\psi\xi} \vee \mathfrak{W}_{\theta a}^{\omega A^{\tau'-\tau}}\mathfrak{W}_{\cdot}^{\omega\xi}. \tag{3.12}$$

Treating the equality (3.12) as an equation for $\mathfrak{W}_{\cdot}^{\omega\xi}$, we obtain

$$\mathfrak{W}_{\cdot}^{\omega\xi} \doteq \{\mathfrak{W}_{\theta a}^{\omega A^{\tau'-\tau}}\}^{*}\mathfrak{W}_{\psi\varepsilon}^{\omega A^{\tau'-\tau}}\mathfrak{W}_{\cdot}^{\psi\xi}, \tag{3.13}$$

but $\mathfrak{W}_{\theta a}^{\omega A^{\tau}} = (Y_{\theta a}^{\omega A})^{\tau}$, $\mathfrak{W}_{\psi\varepsilon}^{\omega A^{\tau}} = \bigvee_{k=0}^{\tau-1}(Y_{\theta a}^{\omega A})^{k} Y_{\psi\varepsilon}^{\omega A}$. Since $\{x^{m}\}^{*} \vee_{k=0}^{m-1} x^{k} = x^{*}$, we obtain from (3.13)

$$\mathfrak{W}_{\cdot}^{\omega\xi} \doteq \{Y_{\theta a}^{\omega A}\}^{*} Y_{\psi\varepsilon}^{\omega A}\mathfrak{W}_{\cdot}^{\psi\xi}. \tag{3.13.1}$$

or

$$\mathfrak{W}_{\cdot}^{\omega\xi} \doteq Y_{\theta a}^{\omega A}\mathfrak{W}_{\cdot}^{\omega\xi} \vee Y_{\psi\varepsilon}^{\omega A}\mathfrak{W}_{\cdot}^{\psi\xi}. \tag{3.14}$$

Equality (3.14) has been obtained from (3.7) by a sequence of correct transformations and it is equivalent to equality (3.7) in the sense that (3.14) is exact if and only if equality (3.7) is exact. The sequence of symbols A in equality (3.14) has been reduced to 1, and therefore the set R_i in the $(i + 1)$-th step of the comparison algorithm is augmented with the new equality (3.14), after which the operations of the procedure Π are no longer applied to it. We thus obtain yet another marking rule, and the actions themselves define a new operation of the procedure Π – the reduction operation. Since this argument is valid for any variable X, and the total number of variables is finite, there exists a minimal number μ_0 that jointly bounds the length of all ε-sequences produced by the comparison algorithm. If the set W^{A} for the special variable A contains more than two rules, then the proof remains the same, although the expressions become longer. This concludes the first part of the proof of Theorem 4.

Now assume that two special symbols A and B exist. If for each of these symbols there are states ψ_A and ψ_B such that the ε-rules $(\varepsilon, \psi_A, A) \to (\psi_A, \varepsilon)$, $(\varepsilon, \psi_B, B) \to (\psi_B, \varepsilon)$ exist, then the proof of the theorem for these symbols is similar to the reasoning in the proof of the first part of Theorem 4. We accordingly assume that both symbols may be erased by ε-rules in the same state ψ, i.e., the pair of rules $(\varepsilon, \psi, A) \to (\psi, \varepsilon)$, $(a, \theta, A) \to (\omega, \varepsilon)$ is augmented with rules for the special symbol B: $(\varepsilon, \psi, B) \to (\psi, \varepsilon)$, $(a', \theta', B) \to (\omega', \varepsilon)$.

In this case, an arbitrary ε-sequence is a word in the alphabet consisting of the two symbols A, B.

Similarly to the first part of the proof, we assume that the comparison algorithm produces for the variable X the equalities $X \doteq \mathfrak{W}_{\cdot}^{\varphi\xi\eta_1 A\xi'}$ and $X \doteq \mathfrak{W}_{\cdot}^{\varphi\xi\eta_2 A\xi'}$, where η_1 and η_2 are ε-sequences, $\eta_2 = \eta_1\eta_1'$. The special symbol A is placed before ξ' for definiteness and does not detract from the generality of our analysis.

Equating the right-hand sides of the equalities, we obtain

$$\mathfrak{W}_{\cdot}^{\varphi\xi\eta_1 A\xi'} \doteq \mathfrak{W}_{\cdot}^{\varphi\xi\eta_2 A\xi'} \tag{3.15}$$

or, transforming similarly to (3.7)-(3.11), we obtain from (3.15)

$$\mathfrak{W}_{\cdot}^{\omega\xi\eta_1} \doteq \mathfrak{W}_{\cdot}^{\omega\xi\eta_2}, \bigvee_{[A]}\mathfrak{W}_{[A]}^{\varphi A\xi'}\mathfrak{W}_{\cdot}^{\mu[A]\xi\eta_1} \doteq \bigvee_{[A]}\mathfrak{W}_{[A]}^{\varphi A\xi'}\mathfrak{W}_{\cdot}^{\mu[A]\xi\eta_2}.$$

Now let $\eta_1' \doteq C\eta_2$, where C is either A or B. Then

$$\mathfrak{W}_{\cdot}^{\omega\xi\eta_1} \doteq \mathfrak{W}_{[C]}^{\omega C\eta_2}\mathfrak{W}_{\cdot}^{\mu[C]\xi\eta_1} \vee \mathfrak{W}_{\psi\varepsilon}^{\omega C\eta'_2}\mathfrak{W}_{\cdot}^{\psi\xi\eta_1}, \tag{3.16}$$

where $[C]$ equals (a, θ) or (a', θ') depending on the value of C. However, we may assume that $C = A$. Otherwise, we would have to consider the following lengthening of the ε-sequence: $\eta_1\eta_1'\eta_1''$, where $\eta_1'' = C\eta_3'$. If $C = A$, then we can replace (3.15) with the equality $\mathfrak{W}_{\cdot}^{\varphi\xi\eta_1 A\xi'} \doteq \mathfrak{W}_{\cdot}^{\varphi\xi\eta_1\eta'_2 A\eta'_3\xi'}$, if $C = B$, then we adjoin η_1 to ξ and treat η_1'' as a lengthening of the ε-sequence η_1'. Thus, (3.16) can be represented in the form

$$\mathfrak{W}_{\cdot}^{\omega\xi\eta_1} \doteq \mathfrak{W}_{\psi\varepsilon}^{\omega A\eta_2}\mathfrak{W}_{\cdot}^{\psi\xi\eta_1} \vee \mathfrak{W}_{\theta a}^{\omega A\eta'_2}\mathfrak{W}_{\cdot}^{\omega\xi\eta_1}, \tag{3.17}$$

whence

$$\mathfrak{W}_{\cdot}^{\omega\xi\eta_1} \doteq \{\mathfrak{W}_{\theta a}^{\omega A\eta'_2}\}^*\mathfrak{W}_{\psi\varepsilon}^{\omega A\eta'_2}\mathfrak{W}_{\cdot}^{\psi\xi\eta_1}. \tag{3.18}$$

For any ε-sequence $\eta = C_{m+1}\ldots C_1$ we can prove by induction

$$\mathfrak{W}_{\theta a}^{\omega A\eta} = \left(\prod_{k=0}^{m} Y_{[C_{k+1}]}^{\mu[C_k]C_{k+1}}\right)^{>\omega<} Y_{\theta a}^{\mu[C_{m+1}]A},$$

where $(\,)^{>\omega<}$ indicates that $\mu[C_0] = \omega$, and

$$\mathfrak{W}_{\psi\varepsilon}^{\omega A\eta} = \left(\bigvee_{r=0}^{m}\prod_{k=0}^{r} Y_{[C_{k+1}]}^{\mu[C_k]C_{k+1}}\right)_{\psi}^{>\omega<};$$

the notation $(\,)_{\psi}^{>\omega<}$ indicates that $\mu[C_0] = \omega$, $\mu[C_{r+1}] = \psi$. Setting $\eta_2' = C_{m+1}\ldots C_1$, we obtain from (3.18)

$$\mathfrak{W}_{\cdot}^{\omega\xi\eta_1} \doteq \left\{\left(\prod_{k=0}^{m} Y_{[C_{k+1}]}^{\mu[C_k]C_{k+1}}\right)^{>\omega<} Y_{\theta a}^{\mu[C_{m+1}]A}\right\}^*\left(\bigvee_{r=0}^{m}\prod_{k=0}^{r} Y_{[C_{k+1}]}^{\mu[C_k]C_{k+1}}\right)_{\psi}^{>\omega<}\mathfrak{W}_{\cdot}^{\psi\xi\eta_1}$$

or, denoting the expression in braces by $\mathfrak{Q}_{\theta a}^{\omega}$, and the expression in parentheses by $\mathfrak{Q}_{\psi}^{\omega}$, we write

$$\mathfrak{W}_{\cdot}^{\omega\xi\eta_1} \doteq \left\{\mathfrak{Q}_{\theta a}^{\omega}\right\}^*\mathfrak{Q}_{\psi}^{\omega}\mathfrak{W}_{\cdot}^{\psi\xi\eta_1}. \tag{3.19}$$

Consider the sequence of equalities $X \doteq \mathfrak{W}_{\cdot}^{\varphi\xi\eta\xi'}$, $X \doteq \mathfrak{W}_{\cdot}^{\varphi\xi\eta'\xi'}$, ..., $X \doteq \mathfrak{W}_{\cdot}^{\varphi\xi\eta^{(n)}\xi'}$, which can be obtained in the process of lengthening the ε-sequence. Since the replacement of X and the use of substitutions to return to the variable X involves a sequence of the same operations, the ε-sequence is augmented by adjoining the same segment, which we denote by ζ. In other words, first the string ξ' is erased, then the string $\zeta \in (A \cup B)^*$ is adjoined, and then again the string ξ' is added.

This process describes a general scheme for reconstruction and increase of the ε-sequence and it is not necessarily implemented step by step in the actual comparison algorithm, whose execution depends on the automata being compared. Different variants are possible. For instance, after the erasure of ξ', the algorithm first adjoins some sequence ξ''', then erases it, and only after that starts increasing the length of the ε-sequence. However, the lengthening of the ε-sequence always proceeds in accordance with the general scheme described above, and we may therefore take $\eta^{(n)} = \eta\xi^n$. Consider the following equality:

$$\mathfrak{W}_{\cdot}^{\varphi\xi\eta_1\zeta\xi'} \doteq \mathfrak{W}_{\cdot}^{\varphi\xi\eta_1\zeta\zeta\xi'}. \tag{3.20}$$

Arguing in (3.20) along the same lines as in the derivation of equality (3.19) from (3.15), we obtain

$$\mathfrak{W}_{\cdot}^{\omega\xi\eta_1} \doteq \left\{\left(\mathfrak{Q}_{\theta a}^{\omega}\right)^2\right\}^*(\mathfrak{Q}_{\theta a}^{\omega}\mathfrak{Q}_{\psi}^{\omega} \vee \mathfrak{Q}_{\psi}^{\omega})\mathfrak{W}_{\cdot}^{\psi\xi\eta_1}. \tag{3.21}$$

Then from (3.21) and (3.19) we have

$$\left\{\mathfrak{Q}_{\theta a}^{\omega}\right\}^*\mathfrak{Q}_{\psi}^{\omega}\mathfrak{W}_{\cdot}^{\omega\xi\eta_1} \doteq \left\{\left(\mathfrak{Q}_{\theta a}^{\omega}\right)^2\right\}^*(\mathfrak{Q}_{\theta a}^{\omega}\mathfrak{Q}_{\psi}^{\omega} \vee \mathfrak{Q}_{\psi}^{\omega})\mathfrak{W}_{\cdot}^{\psi\xi\eta_1}. \tag{\#}$$

But by the identity $(x^2)^*(xy \vee y) = x^*y$, the conditional equality (#) is an identity. Therefore, the equality (3.17) may

be included in the set R_i and the equality $X \doteq \mathfrak{W}^{\psi\xi\eta\zeta^2\xi'}$ may be marked as previously occurring in the form $X \doteq \mathfrak{W}^{\psi\xi\eta\zeta\xi'}$; there is no need to transform this equality any further by operations of the procedure Π. Here also we say that the reduction operation is applied. The ε-sequence thus does not increase without bound in this case either. We denote its maximum length increase by the same number μ_0 that has been previously introduced for the case with one special symbol.

The proof for an arbitrary but finite number of special symbols is constructed similarly. This completes the proof of Theorem 4. Q.E.D.

The proof of Theorem 5 follows directly from [20] (Lemmas [20].2-[20].6 and Theorem [20].4). It is accordingly omitted.

4. PROOF OF THE MAIN THEOREMS

The proof of the first main theorem (Theorem 6) is identical with the proof of Theorem [20].2 (Sec. 4), replacing the number $\Lambda = (\lambda^0 + 1)(\sigma + 1) + 2$ introduced in the proof of the theorem in [20] with the number $\Lambda = (\mu_0 + \lambda^0)(\sigma + 1) + 2$ and the bound provided by inequality [20].(4.4) with the bound $n \leq \Lambda\sigma(\mu_0 + 1)$.

The proof of the second main theorem (Theorem 7) is modified in comparison with [20] and it is therefore reproduced here in full.

We first prove one lemma, using the value $\Lambda = (\mu_0 + \lambda^0)(\sigma + 1) + 2$ in its proof. Let τ be the number of different variables in the union of the alphabets $\bar{X}$ and $\bar{Y}$, $\tau = |\bar{X}| + |\bar{Y}|$. Moreover, let $\mathfrak{G}_{i_0}$ be the category constructed by the comparison algorithm, and the set R_{i_0} includes only marked equalities. Then we have the following lemma.

LEMMA 5. The category $\mathfrak{G}_{i_0}$ is the direct limit of the category system $K = \{\mathfrak{G}_0, \mathfrak{G}_1, \mathfrak{G}_2, \ldots, \mathfrak{G}_{i_0}\}$.

Indeed, by finiteness of K, the category $\mathfrak{G}_{i_0}$ is an element of K. Moreover, for each category $\mathfrak{G}_i$, by construction of K, there exists a direct morphism $F_{ii_0}: \mathfrak{G}_i \to \mathfrak{G}_{i_0}$, which is an embedding functor of $\mathfrak{G}_i$ in $\mathfrak{G}_{i_0}$. Thus, $\mathfrak{G}_{i_0}$ is the direct limit of the system K, $\mathfrak{G}_{i_0} = \lim_{\to} K$.

LEMMA 6. The number of expressions defining the morphisms of the category $\lim_{\to} K$ representable by disjunctive polynomials in the alphabet $\Sigma \cup \bar{X} \cup \bar{Y}$ is finite and does not exceed $2^{\tau^{\Lambda\sigma(\mu_0+1)+1}}$.

Indeed, in the proof of Theorem [20].2 we have obtained a bound of the form $\Lambda\sigma(\mu_0 + 1)$ on the characteristic of any expression forming a morphism (allowing for the changes specified above). But the number of different expressions in a τ-symbol alphabet whose length does not exceed N is bounded by τ^{N+1} or for $N = \Lambda\sigma(\mu_0 + 1)$ by $\tau^{\Lambda\sigma(\mu_0+1)}$. Then the number of different subsets from this set that may represent morphisms of the direct limit of the system K does not exceed $2^{\tau^{\Lambda\sigma(\mu_0+1)+1}}$ Q.E.D.

Proof of Second Main Theorem (Theorem 7). Apply the comparison algorithm to the DPDAs $\mathfrak{A}$ and $\mathfrak{B}$ and assume that the algorithm stops after a certain number of steps. Two variants are possible. In the first variant, the algorithm has constructed the category $\lim_{\to} K$ always using only the positive marking rules ⟨r1⟩ and ⟨r2⟩. In this case, the automata $\mathfrak{A}$ and $\mathfrak{B}$ are equivalent. Indeed, if all the operations applied by the comparison algorithm are correct, equivalence follows from Theorem 5. If some substitutions are incorrect, then again they cannot transform unequal morphisms into equal morphisms. Therefore, arguing as in the proof of Theorem 6, we establish equivalence of the automata.

In the second variant, the comparison algorithm stops after a certain step because it has found a morphism that satisfies one of the negative conditions ⟨m1⟩ or ⟨m2⟩. In either case, there is a pair of words such that at least one of them does not belong to both automata. Substitute these words in the automata $\mathfrak{A}$ and $\mathfrak{B}$. Two cases are possible. In the first case, we have found a word which is not accepted by one of the automata. The automata are inequivalent, and the equivalence problem is decidable, because we have found a word that is accepted by one automaton and not accepted by the other. This completes the proof of Theorem 7 for this case.

But there is also another case, when the words are accepted by both automata and the inequality in the comparison algorithm is the result of an incorrect substitution operation: its incorrectness is attributable to dependence of the characteristic system of one of the equalities to which this operation is applied. Then the word may be used actually to identify the dependent element of the characteristic system and to construct a new characteristic from which the dependent semicomplex is excluded.

In this case, the comparison algorithm is restarted with the new characteristic system from the point of substitution in the equality.

Formally, this is accomplished in the following way. Assume that characteristic system $\aleph = \{\mathfrak{W}_{\cdot}^{\mu[A]_j\zeta} \mid 0 \leq j \leq J\}$ of equality (2.7) is dependent and the word x_0 effectively establishes this dependence, producing from (2.7) the equality

$$\mathfrak{W}_{\cdot}^{\mu[A]_0\zeta} \doteq \bigvee_{j=0}^{J} \mathfrak{W}_{[A]_j}^{\varphi A\xi_0} \mathfrak{W}_{\cdot}^{\mu[A]_j\zeta}. \tag{4.1}$$

Here we have explicitly written $(J + 1)$ pairs that correspond to states in which the symbol A is erased and to the letters of the alphabet Σ entering the erasure rule, i.e., the pairs $[A]_j$ $(0 \leq j \leq J)$.

Two cases are possible. The first when $\mathfrak{W}_{[A]_0}^{\varphi A\xi_0} = \varnothing$, and the second when $\|\mathfrak{W}_{[A]_0}^{\varphi A\xi_0}\| \geq 1$. In the first case, we can substitute in (2.7) the expression $\mathfrak{W}_{\cdot}^{\mu[A]_0\zeta}$, defined by (4.1) and pass to a new equality with the characteristic system $\aleph' = \{\mathfrak{W}_{\cdot}^{\mu[A]_j\zeta} \mid 1 \leq j \leq J\}$, without the complex $\mathfrak{W}_{\cdot}^{\mu[A]_0\zeta}$, restarting the comparison algorithm.

The second case is more complicated. The equality (4.1) has the form

$$\mathfrak{W}_{\cdot}^{\mu[A]_0\zeta} \doteq \mathfrak{W}_{[A]_0}^{\varphi A\xi_0} \mathfrak{W}_{\cdot}^{\mu[A]_0\zeta} \vee \bigvee_{j=1}^{J} \mathfrak{W}_{[A]_j}^{\varphi A\xi_j} \mathfrak{W}_{\cdot}^{\mu[A]_j\zeta}$$

or

$$\mathfrak{W}_{\cdot}^{\mu[A]_0\zeta} \doteq \left\{\mathfrak{W}_{[A]_0}^{\varphi A\xi_0}\right\}^* \bigvee_{j=0}^{J} \mathfrak{W}_{[A]_j}^{\varphi A\xi_j} \mathfrak{W}_{\cdot}^{\mu[A]_j\zeta}, \tag{4.2}$$

where the symbol $\{\ \}^*$ denotes iteration of the event in braces.

Introduce the variable $Z_{[A]_0}^{\varphi A\xi_0}$ defined by the exact equality

$$Z_{[A]_0}^{\varphi A\xi_0} \doteq \mathfrak{W}_{[A]_0}^{\varphi A\xi_0} Z_{[A]_0}^{\varphi A\xi_0} \vee \varepsilon. \tag{4.3}$$

Then the equality (4.2) may be rewritten as

$$\mathfrak{W}_{\cdot}^{\mu[A]_0\zeta} \doteq Z_{[A]_0}^{\varphi A\xi_0} \bigvee_{j=1}^{J} \mathfrak{W}_{[A]_j}^{\varphi A\xi_0} \mathfrak{W}_{\cdot}^{\mu[A]_j\zeta}. \tag{4.4}$$

Substituting (4.4) in (2.7) and grouping the corresponding terms, we obtain

$$\begin{aligned} \bigvee_{i} \bigvee_{j=1}^{J} X_i \left(\mathfrak{W}_{[A]_j}^{\psi_i A\zeta_i\eta_i} \vee \mathfrak{W}_{[A]_0}^{\psi_i A\zeta_i\eta_i} Z_{[A]_0}^{\varphi A\xi_0} \mathfrak{W}_{[A]_j}^{\varphi A\xi_0} \right) \mathfrak{W}_{\cdot}^{\mu[A]_j\zeta} \doteq \\ \doteq \bigvee_{j=1}^{J} \left(\mathfrak{W}_{[A]_j}^{\psi A\bar{\xi}} \vee \mathfrak{W}_{[A]_0}^{\psi A\bar{\xi}} Z_{[A]_0}^{\varphi A\xi_0} \mathfrak{W}_{[A]_j}^{\varphi A\xi_0} \right) \mathfrak{W}_{\cdot}^{\mu[A]_j\zeta}. \end{aligned} \tag{4.5}$$

Since the variable $Z_{[A]_0}^{\varphi A\xi_0}$ in (4.5) is necessarily followed by one of the expressions $\mathfrak{W}_{[A]_j}^{\varphi A\xi_0}$ $(1 \leq j \leq J)$, we introduce J variables $Z_{0j}^{\varphi A\xi_0} = Z_{[A]_0}^{\varphi A\xi_0} \mathfrak{W}_{[A]_j}^{\varphi A\xi_0}$ with the norm $\|Z_{0j}^{\varphi A\xi_0}\| = \|\mathfrak{W}_{[A]_j}^{\varphi A\xi_0}\| \geq 1$, defined by the exact equalities

$$Z_{0j}^{\varphi A\xi_0} = \mathfrak{W}_{[A]_0}^{\varphi A\xi_0} Z_{0j}^{\varphi A\xi_0} \vee \mathfrak{W}_{[A]_j}^{\varphi A\xi_0} \ (1 \leq j \leq J). \tag{4.5.1}$$

Then (4.5) may be rewritten as

$$\bigvee_i \bigvee_{j=1}^{J} X_i \left(\mathfrak{W}_{[A]_j}^{\psi_i A \zeta_i \eta_i} \vee \mathfrak{W}_{[A]_0}^{\psi_i A \zeta_i \eta_i} Z_{0j}^{\varphi A \xi_0} \right) \mathfrak{W}^{\mu[A]_j \zeta} \doteq$$

$$\doteq \bigvee_{j=1}^{J} \left(\mathfrak{W}_{[A]_j}^{\psi A \bar{\xi}} \vee \mathfrak{W}_{[A]_0}^{\psi A \bar{\xi}} Z_{[A]_{0j}}^{\varphi A \xi_0} \right) \mathfrak{W}^{\mu[A]_j \zeta}. \tag{4.6}$$

The equality (4.6) with the characteristic system $\aleph = \left\{ \mathfrak{W}^{\mu[A]_j \zeta} \mid 1 \leqslant j \leqslant J \right\}$ replaces the equality (2.7) and is subsequently used instead of (2.7) in the comparison algorithm. Assuming that the system $\aleph'$ is independent, we obtain from (4.6) the inverse system

$$\bigvee_i X_i \left(\mathfrak{W}_{[A]_j}^{\psi_i A \zeta_i \eta_i} \vee \mathfrak{W}_{[A]_0}^{\psi_i A \zeta_i \eta_i} Z_{0j}^{\varphi A \xi_0} \right) \doteq$$

$$\doteq \mathfrak{W}_{[A]_j}^{\psi A \bar{\xi}} \vee \mathfrak{W}_{[A]_0}^{\psi A \bar{\xi}} Z_{0j}^{\varphi A \xi_0}, \tag{4.7}$$

that replaces the system (2.8).

If the system $\aleph'$ subsequently becomes dependent and the comparison algorithm identifies a word that establishes this dependence, then we repeat the entire process of introduction of the variables Z and elimination of the dependent component from the system $\aleph'$. However, the number of such dependences is finite because the number of expression elements forming any characteristic system is finite. Variables of the form $Z_{ij}^{\psi \xi}$ are called auxiliary variables.

Let us analyze the effect of the introduction of auxiliary variables on the comparison algorithm.

First let us estimate the norm of the auxiliary variable Z. In the derivation of equality (4.6) we have shown that $\| Z_{0j}^{\varphi A \xi_0} \| = \| \mathfrak{W}_{[A]_j}^{\varphi A \xi_0} \|$, and equality (4.4) may be written in the form

$$\mathfrak{W}^{\mu[A]_0 \zeta} \doteq \bigvee_{j=1}^{J} Z_{0j}^{\varphi A \xi_0} \mathfrak{W}^{\mu[A]_j \zeta}. \tag{4.8}$$

Let $\zeta = P_m \ldots P_1$. From (4.8) $\| \mathfrak{W}^{\mu[A]_0 P_m \ldots P_1} \| = \min_j \left\{ \| Z_{0j}^{\varphi A \xi_0} \| + \| \mathfrak{W}^{\mu[A]_j P_m \ldots P_1} \| \right\} = \| Z_{0j_s}^{\varphi A \xi_0} \| + \| \mathfrak{W}^{\mu[A]_{j_s} P_m \ldots P_1} \|$ for some j with index s, and hence for the norm

$$\| Z_{0j_s}^{A \xi_0} \| = \| \mathfrak{W}^{\mu[A]_0 P_m \ldots P_1} \| - \| \mathfrak{W}^{\mu[A]_{j_s} P_m \ldots P_1} \| =$$

$$= \| \vee_\kappa Y_{[P_1]_k}^{\mu[A]_0 P_1} \mathfrak{W}^{\mu[P_1]_k P_m \ldots P_2} \| - \| \vee_\kappa Y_{[P_1]_k}^{\mu[A]_{j_s} P_1} \mathfrak{W}^{\mu[P_1]_k P_m \ldots P_2} \| =$$

$$= \min_k \left\{ \| Y_{[P_1]_k}^{\mu[A]_0 P_1} \| + \| \mathfrak{W}^{\mu[P_1]_k P_m \ldots P_2} \| \right\} - \min_k \left\{ \| Y_{[P_1]_k}^{\mu[A]_{j_s} P_1} \| + \| \mathfrak{W}^{\mu[P_1]_k P_m \ldots P_2} \| \right\} =$$

$$= \| Y_{[P_1]_{k_0}}^{\mu[A]_0 P_1} \| + \| \mathfrak{W}^{\mu[P_1]_{k_0} P_m \ldots P_2} \| - \| Y_{[P_1]_{k_1}}^{\mu[A]_{j_s} P_1} \| - \| \mathfrak{W}^{\mu[P_1]_{k_1} P_m \ldots P_2} \|.$$

If $k_0 \neq k_1$, then

$$\| Y_{[P_1]_{k_0}}^{\mu[A]_0 P_1} \| + \| \mathfrak{W}^{\mu[P_1]_{k_0} P_m \ldots P_2} \| = \min_k \left\{ \| Y_{[P_1]_k}^{\mu[A]_0 P_1} \| + \| \mathfrak{W}^{\mu[P_1]_k P_m \ldots P_2} \| \leqslant Y_{[P_1]_{k_1}}^{\mu[A]_0 P_1} + \| \mathfrak{W}^{\mu[P_1]_k P_m \ldots P_2} \| \right.$$

Therefore, from the equality for $\| Z_{0j_s}^{\varphi A \xi_0} \|$ we obtain

$$\| Z_{0j_s}^{\varphi A \xi_0} \| \leqslant Y_{[P_1]_{k_1}}^{\mu[A]_0 P_1} - Y_{[P_1]_{k_1}}^{\mu[A]_{j_s}} \leqslant \sigma - 1 < \sigma.$$

But $\| Z_{0j_s}^{\varphi A \xi_0} \| = \| \mathfrak{W}_{[A]_{j_s}}^{\varphi A \xi_0} \|$. Therefore, if $\| \mathfrak{W}_{[A]_{j_s}}^{\varphi A \xi_0} \| < \sigma$, then the length of the characteristic ξ_0 is a priori less than $\sigma \mu_0$.

Then for any $\|Z_{0j}^{\varphi A\xi_0}\|$ the condition $\|Z_{0j}^{\varphi A\xi_0}\| = \|\mathfrak{W}_{[A]_j}^{\varphi A\xi_0}\|$ gives

$$\|Z_{0j}^{\varphi A\xi_0}\| < \sigma^2\mu_0. \tag{4.9}$$

Since the choice of the variable $\|Z_{0j}^{\varphi A\xi_0}\|$ is arbitrary, we have the following lemma.

LEMMA 7. The characteristic and the norm of any auxiliary variable are upper bounded by $\sigma\mu_0$ and $\sigma^2\mu_0$, respectively.

LEMMA 8. The total number of auxiliary variables that may appear during the application of the comparison algorithm is bounded and finite.

Indeed, any auxiliary variable $Z_{kj}^{\varphi A\xi_0}$ is determined by five parameters φ, A, ξ_0, k, j. The number of possible values of the parameters φ, k, j is finite because the state sets of the automata $\mathfrak{A}$ and $\mathfrak{B}$ are finite, $A \in \Gamma$, and the characteristic length is bounded by Lemma 7. Thus, the total number of possible auxiliary variables that may be constructed for a pair of DPDAs in the modified comparison algorithm is also bounded and finite. Q.E.D.

To prove Theorem 7 for the modified algorithm, we need to repeat the previous proof introducing a system of equations for the auxiliary variables in addition to the original systems of equations $\mathfrak{A}(\bar{X})$ and $\mathfrak{B}(\bar{Y})$ defined by the automata being compared. This additional system is a priori finite by Lemma 8, although the total number of equations in it is not known in advance and is determined during the execution of the comparison algorithm.

The argument must be repeated because the norm of the auxiliary variables is upper bounded by $\sigma^2\mu_0$, while the norm of the variables X, Y is upper bounded by $\sigma\lambda_0$. However, although the bounds change, the general reasoning remains the same, because the number $\sigma^2\mu_0$ is also finite and thus the total number of possible conditional inequalities arising during the execution of the comparison algorithm is finite.

LEMMA 9. The total number of possible dependences that cause backtracking in the modified comparison algorithm is finite.

Indeed, each dependence producing backtracking either reduces by 1 the number of elements in the characteristic system or proves inequivalence of the automata. Since the length of each characteristic system and the total number of possible equalities are both bounded, the possible number of backtrackings is finite. Q.E.D.

Let us return to the proof of Theorem 7. Denote $\pi_0 = \max_{T \in \Gamma\mathfrak{A} \cup \Gamma\mathfrak{B}} |W^T|$, i.e., π_0 is the maximum number of erasing rules for a variable from the set of symbols in the union of the internal alphabets of the two automata. Equality (4.7) shows that the introduction of an auxiliary variable increases the characteristic of the right-hand side of the inverse system by an amount not exceeding $\sigma\mu_0$ (Lemma 7). Since the number of such introductions for one equality is at most $(\pi_0 - 1)$, the maximum possible increase of the characteristic is less than $\pi_0\mu_0\sigma$.

Therefore, the value Λ entering the definition of the interval [20]. Equation (4.2) is taken equal to

$$\Lambda_M = (\pi_0\sigma + 1)\mu_0 + (\sigma + 1)\lambda^0 + 2. \tag{4.10}$$

It bounds the possible characteristic length of the left-hand side of any equality that may occur in the modified comparison algorithm. The substitution operation must be applied to an equality when the characteristic of its left-hand side falls in the interval $[\Lambda_M - (\lambda^0 + \sigma), \Lambda_M]$.

Indeed, reasoning as in the proof of Theorem [20].2 we can show that substitution reduces the characteristic of the left-hand side of any equality included in the direct system compared with the original equality, while for any equality in the inverse system the characteristic length of the left-hand side does not fall in the interval $[\Lambda_M - (\lambda^0 + \sigma), \Lambda_M]$ even after the removal of the maximum number of dependences that may be produced by the substitution operation.

The modified comparison algorithm thus may produce a finite total number of possible equality conditions. The number of elements of the category system K is therefore also finite, which as in the proof of Theorem [20].2 determines finiteness of the total number of steps in the modified algorithm.

The modified comparison algorithm performs the following general comparison procedure for the automata $\mathfrak{A}$ and $\mathfrak{B}$.

As in the proof of Theorem [20].2, the equivalence check of the automata starts with the application of the comparison algorithm. The construction of the category system K continues as long as the comparison algorithm does not stop. Only positive marking rules are used, and in the last step either all the equalities are marked, which means that the automata are equivalent, or a negative rule is applied. In the latter case, we can find a morphism that identifies the word $\hat{w}$ on which the automata differ.

For each automaton we then check if it accepts the word $\hat{w}$ or not. If only one of the automata accepts this word, the automata are inequivalent and the problem is solved. If both automata accept $\hat{w}$, we use this word to find a dependence in the characteristic system of one of the equalities constructed during the derivation of the equality on which the difference is observed. Then the modified comparison algorithm is again applied starting at this point, until the next stop.

Since the total number of possible dependences is finite, the modified comparison algorithm will eventually produce a category system K_n that establishes equivalence of the automata or alternatively it will find a word that is accepted by one automaton and is not accepted by the other, i.e., the algorithm will establish inequivalence of the two automata. Since this result will be obtained after a finite number of steps, the equivalence problem for deterministic pushdown automata is decidable.

This completes the proof of the second main theorem and concludes our analysis of the equivalence problem for deterministic pushdown automata.

REFERENCES

1. S. Ginsburg, The Mathematical Theory of Context Free Languages McGraw-Hill, New York (1966).
2. S. Ginsburg and S. Greibach, "Deterministic context-free languages," IEEE Conf. Record on Switching Circuit Theory and Logical Design (Oct. 1965), pp. 203-220.
3. S. Ginsburg and S. Greibach, "Deterministic context-free languages," Inform. Contr., **9**, No. 6, 620-648 (1966).
4. D. Knuth, "On the translation of languages from left to right," Inform. Contr., **8**, No. 6, 607-639 (1965).
5. D. Rosenkrantz and R. Stearns, "Properties of deterministic top-down grammars," Inform. Contr., **17**, No. 3, 226-256 (1970).
6. A. Korenjak and J. Hopcroft, "Simple deterministic languages," IEEE 7th Annual Symp. on Switching and Automata Theory Design, Berkeley (1966), pp. 36-46.
7. Z. Valiant, "The decidability of equivalence for deterministic finite-turn pushdown automata," Inform. Contr., **25**, No. 2, 123-133 (1974).
8. K. Taniguchi and T. Kasami, "A result on the equivalence problem for deterministic pushdown automata," JCSS, **13**, No. 1, 38-50 (1976).
9. Z. Valiant and M. Paterson, "Deterministic one-counter automata," JCSS, **13**, No. 1, 340-350 (1975).
10. M. Zinna, "Two decidability results for deterministic pushdown automata," Lect. Notes Comput. Sci., **53**, 365-373 (1977).
11. V. V. Zubenkov, "On finite backtracking automata," Dokl. Akad. Nauk UkrSSR, Ser. A, No. 3, 838-841 (1978).
12. V. Yu. Romanovskii, "Equivalence problem of real-time deterministic pushdown automata," Kibernetika, No. 2, 13-23 (1985).
13. A. A. Letichevskii, "On relations representable in push-down automata," Kibernetika, No. 1, 1-9 (1969).
14. A. Aho and D. Ullman, Theory of Parsing, Translation, and Compiling [Russian translation], Vols. 1, 2, Mir, Moscow (1978).
15. D. Lehman, "*LR(k)*-grammars and deterministic languages," in: Languages and Automata [Russian translation], Mir, Moscow (1975), pp. 43-46.
16. A. V. Anisimov, "Automaton characterization of parsing methods with regular control," Kibernetika, No. 1, 1-8 (1976).
17. E. Fridman, "Equivalence problems for deterministic context-free languages and monadic recursion schemes," JCSS, **14**, No. 3, 344-359 (1977).
18. S. Garland and D. Zuckham, "Program schemes, recursion schemes, and formal languages," JCSS, **7**, No. 2, 119-160 (1973).
19. M. Geller and M. Harrison, "Characterization of *LR*(0)-languages," Proc. 14th Annual Symp. on Switching and Automata Theory, Iowa City (1973), pp. 103-108.
20. V. Yu. Meitus, "The equivalence problem for real-time strict deterministic pushdown automata," Kibernetika, No. 5, 14-27 (1989).