Theoretical Computer Science 51 (1987) 301–308
North-Holland

ON PURELY MORPHIC CHARACTERIZATIONS OF CONTEXT-FREE LANGUAGES

Takashi YOKOMORI

International Institute for Advanced Study of Social Information Science, Fujitsu Limited, Numazu, Shizuoka 410-03, Japan

Communicated by A. Salomaa
Received May 1986
Revised September 1986

Abstract. In this paper we show the following: For any λ-free context-free language L there effectively exist a weak coding g, a homomorphism h such that $L = gh^{-1}(¢D_2)$, where D_2 is the Dyck set over a two-letter alphabet. As an immediate corollary it follows that for any λ-free context-free language L there exist a weak coding g and a mapping F such that $L = gF^{-1}(¢)$.

1. Introduction

There are a number of works which have been devoted to characterization theorems of context-free languages of the form: Let $\mathscr{L}$ be a class of languages; then each L in $\mathscr{L}$ can be expressed as $f(L_0)$, where L_0 is a possibly fixed, simple language in $\mathscr{L}$, and f is a simple combination of language operations. A well-known Chomsky and Stanley characterization in [3] says that L can be expressed as $h(D \cap R)$ for some homomorphism h, a Dyck set D, and a regular set R; in other words, as $f(D)$ where f is a finite transduction. Greibach's theorem [2] asserts that $L = h^{-1}(L_0)$ for some fixed language L_0 and a homorphism h. Further, several works on representation theorems of AFLs show that L can be expressed in the form $L = h_4 h_3^{-1} h_2 h_1^{-1}(D\$)$, for some homomorphisms h_1, h_2, h_3, h_4 (cf. [4, 5, 8]).

In this paper we shall prove the following characterization of context-free languages: For each λ-free context-free language L there exist a weak coding g and a homomorphism h such that $L = gh^{-1}(¢D_2)$, where D_2 is the Dyck set over a two-letter alphabet, and ¢ is a specific letter.

The reader is assumed to be familiar with the rudiments of formal language theory. Here we provide only the following definitions.

A homomorphism h is called a *weak coding* if the image of h is either a symbol or the empty string. A weak coding is called a *coding* if it is λ-free. For a language L over T, the *inverse homomorphism* h^{-1} is defined by $h^{-1}(L) = \{x \mid h(x) \text{ in } L\}$.

Let $G = (N, T, P, S)$ be a context-free grammar in Greibach normal form, i.e., N is the set of nonterminals, T is the set of terminals, S the initial symbol in N, and each rule of P is of one of the forms:

$$A \to aB_1 \dots B_n, \qquad A \to a, \qquad S \to \lambda,$$

where $B_1, \dots, B_n$ in $N - \{S\}$, a in T.

G is called *simple deterministic* if, for all A in N, a in T, and u, v in N^*, $A \to au$ and $A \to av$ in P imply $u = v$. A language L is *simple deterministic* if L is generated by some simple deterministic context-free grammar.

Let $r \geq 1$ and define $G_r = (\{S\}, T_r \cup \tilde{T}_r, P, S)$, where $T_r = \{a_1, \dots, a_r\}$, $\tilde{T}_r = \{\tilde{a}_i \mid a_i$ in $T_r\}$, and $P = \{S \to Sa_iS\tilde{a}_iS\} \cup \{S \to \lambda\}$. A language D_r generated by G_r is called the *Dyck set over* T_r.

Let $A = (Q, T, d, q_0, F)$ be a finite-state automaton; then $T(A)$ denotes the set of strings accepted by A. Similarly, for a pushdown automaton $B = (Q, T, K, d, q_0, Z_0, F)$, $T(B)$ denotes the set of strings accepted by B with final states, while $N(B)$ denotes the set of strings accepted by B with the empty stack, i.e.,

$$T(A) = \{w \text{ in } T^* \mid (q_0, w, Z_0) \vdash^* (q, \lambda, u), q \text{ in } F\}$$

and

$$N(A) = \{w \text{ in } T^* \mid (q_0, w, Z_0) \vdash^* (q, \lambda, \lambda)\}.$$

As a notation, the transition relation $\vdash$ is sometimes associated with A as in $\vdash_A$ if necessary.

In what follows, our attention is mainly concentrated on λ-free languages. This is because we would like to discuss only the essence of the subject in question, and the result for the general case can be easily extended from the case of λ-freeness.

2. An inverse homomorphic characterization of context-free languages

Lemma 2.1. *For any λ-free simple deterministic language L, there effectively exist a coding g and a homomorphism h such that $L = g(h^{-1}(¢D_2))$, where D_2 is the Dyck set over $T_2 = \{a_1, a_2\}$, ¢ is a symbol not in $T_2 \cup \tilde{T}_2$.*

Proof. We show the following: for any λ-free simple deterministic grammar G, there exist a simple deterministic grammar G_0, a coding g and a homomorphism h such that $L(G) = g(L(G_0))$ and $L(G_0) = h^{-1}(¢D_2)$. This will immediately complete the proof.

Let $G = (N, T, P, S_0)$ be a λ-free simple deterministic grammar such that $L = L(G)$, where $N = \{A_1(=S_0), \dots, A_n\}$. Construct a simple deterministic grammar $G_0 = (N, T', P', S_0)$ as follows:

$$T' = \{[A, a] \mid A \to ax \text{ in } P\}, \qquad P' = \{A \to [A, a]x \mid A \to ax \text{ in } P\}.$$

Define g by $g([A, a]) = a$ for $[A, a]$ in T'. Then, it is obvious that G_0 is simple deterministic and $L(G) = g(L(G_0))$ holds.

Now, since G is simple deterministic, one can define a homomorphism h from T'^* into $\{a_1, a_2, \tilde{a}_1, \tilde{a}_2, ¢\}^*$ as follows:

$$h([A_i, a]) = \tilde{a}_1\tilde{a}_2^i\tilde{a}_1 a_1 a_2^{jm} a_1 \dots a_1 a_2^{j1} a_1$$

$$\text{if } A_i \to aA_{j1} \dots A_{jm} \text{ in } P \text{ and } i \neq 1,$$

$$h([A_1, a]) = ¢ a_1 a_2^{jm} a_1 \dots a_1 a_2^{j1} a_1$$

$$\text{if } A_1 \to aA_{j1} \dots A_{jm} \text{ in } P,$$

$$h([A_i, a]) = \tilde{a}_1\tilde{a}_2^i\tilde{a}_1 \quad \text{if } A_i \to a \text{ in } P \text{ and } i \neq 1,$$

$$h([A_1, a]) = ¢ \qquad \text{if } A_1 \to a \text{ in } P.$$

It suffices to show that $L(G_0) = h^{-1}(¢D_2)$ holds. We claim the following: For $b_1, \dots, b_k$ in T', $A_{i1}, \dots, A_{ir}$ in $N - \{A_1\}$, we have

$$A_1 \Rightarrow_L^k b_1 \dots b_k A_{i1} \dots A_{ir} \ (r \geq 0) \text{ in } G_0$$

iff (1) $h(b_1 \dots b_k) = ¢y$ is a prefix of a word in $¢D_2$,

(2) $f(¢y) = ¢a_1 a_2^{ir} a_1 \dots a_1 a_2^{i1} a_1$,

where f is a mapping defined by $f(\lambda) = \lambda$ and, for $i = 1, 2$,

$$f(xa_i) = f(x)a_i,$$

$$f(x\tilde{a}_i) = f(x)\tilde{a}_i \quad \text{if } f(x) \text{ not in } \{a_1, a_2, \tilde{a}_1, \tilde{a}_2\}^*\{a_i\},$$

$$f(x\tilde{a}_i) = x' \qquad \text{if } f(x) = x'a_i,$$

$$f(¢) = ¢.$$

(Note that $\Rightarrow_L^k$ indicates the k-step leftmost derivation, i.e., k consecutive rewriting steps in which the leftmost nonterminal is always rewritten. It is well-known that any word generated by a simple deterministic grammar has a unique leftmost derivation. Further, from the property of a simple deterministic grammar, the length of a generated nonempty word exactly equals the number of derivation steps used. A mapping image $f(w)$, the reduced word, is the final resultant obtained by cancelling all pairs $a_i\tilde{a}_i$.)

It should be noted that the claim suffices to prove the lemma. We shall prove the claim by induction on the length of derivation steps.

(*Basic step*, $k = 1$): Suppose that $A_1 \Rightarrow b_1$ or $A_1 \Rightarrow b_1 A_{i1} \dots A_{ir}$. There exists $A_1 \to b_1$ or $A_1 \to b_1 A_{i1} \dots A_{ir}$ in P'. Then, $h(b_1) = ¢$ or $h(b_1) = ¢a_1 a_2^{ir} a_1 \dots a_1 a_2^{i1} a_1$. Clearly, condition (2) holds for either case. Conversely, assuming (1) and (2) for $k = 1$ gives us that $h(b_1) = ¢y_1$ is a prefix of a word in $¢D_2$ and $f(¢y_1) = ¢a_1 a_2^{ir} a_1 \dots a_1 a_2^{i1} a_1$. From the way of constructing h, if $f(¢y_1) = ¢$ $(r = 0)$, i.e., y_1 is in D_2, then we have $A_1 \to b_1$ is in P' leading to $A_1 \Rightarrow b_1$. Otherwise, $h(b_1) = ¢y_1 = ¢a_1 a_2^{ir} a_1 \dots a_1 a_2^{i1} a_1$ implies that $A_1 \to b_1 A_{i1} \dots A_{ir}$ is in P'. This verifies the case $k = 1$.

(*Induction step*): Suppose that $A_1 \Rightarrow_L^k b_1 \ldots b_k A_{i1} \ldots A_{ir}$ $(r \geq 1)$ and $A_{i1} \to b_{k+1} A_{j1} \ldots A_{jm}$ $(m \geq 0)$ is used at the $(k+1)$st step. Let $h(b_{k+1}) = y_{k+1}$. By the induction hypothesis, $f(¢y_1 \ldots y_k) = ¢a_1 a_2^{ir} a_1 \ldots a_1 a_2^{i1} a_1$. Then, we have

$$f(h(b_1 \ldots b_{k+1})) = f(¢y_1 \ldots y_{k+1})$$

$$= ¢a_1 a_2^{ir} a_1 \ldots a_1 a_2^{i2} a_1 a_1 a_2^{jm} a_1 \ldots a_1 a_2^{j1} a_1.$$

(Note that $y_{k+1} = \tilde{a}_1 \tilde{a}_2^{i1} \tilde{a}_1 a_1 a_2^{jm} a_1 \ldots a_1 a_2^{j1} a_1$.)

This also implies that $h(b_1 \ldots b_{k+1})$ is a prefix of a word in $¢D_2$. Since $A_1 \Rightarrow_L^{k+1} b_1 \ldots b_{k+1} A_{j1} \ldots A_{jm} A_{i2} \ldots A_{ir}$, the 'only-if' part of the claim is proved.

Conversely, suppose that we have $h(b_1 \ldots b_{k+1}) = ¢y_1 \ldots y_{k+1}$ is a prefix of a word in $¢D_2$ and $f(¢y_1 \ldots y_{k+1}) = ¢a_1 a_2^{ir} a_1 \ldots a_1 a_2^{i1} a_1$. From the construction of h, we have a partition:

$$f(¢y_1 \ldots y_k) = ¢a_1 a_2^{ir} a_1 \ldots a_1 a_2^{ir} a_1,$$

$$f(y_{k+1}) = h(b_{k+1}) = \tilde{a}_1 \tilde{a}_2^{t} \tilde{a}_1 a_1 a_2^{is} a_1 \ldots a_1 a_2^{i1} a_1,$$

where there exists $A_t \to b_{k+1} A_{i1} \ldots A_{is}$ in P'. However, since $f(¢y_1 \ldots y_{k+1})$ is a word of the form $¢a_1 a_2^{ir} a_1 \ldots a_1 a_2^{i1} a_1$ there must be some cancellation between the two, which implies that $ip = t$. By the induction hypothesis,

$$A_1 \Rightarrow_L^k b_1 \ldots b_k A_t \ldots A_{ir},$$

and applying $A_t \to b_{k+1} A_{i1} \ldots A_{is}$, we have

$$A_1 \Rightarrow_L^{k+1} b_1 \ldots b_{k+1} A_{i1} \ldots A_{is} \ldots A_{ir}.$$

This completes the proof. □

Lemma 2.2. *For an arbitrarily given regular set R over $T_r \cup \tilde{T}_r$, there exists a deterministic pushdown automaton such that*

(i) *it accepts $(D_r \cap R)\#$ with the empty stack, where $\#$ is a specific symbol not in $T_r \cup \tilde{T}_r$, and*

(ii) *it has a special state which appears only once as a final state and no transition from this state is defined.*

Proof. Let $A = (Q_0, T_r \cup \tilde{T}_r, d_A, p_0, F_A)$ be a deterministic finite-state automaton such that $T(A) = R$. Then, by constructing $A' = (Q_0 \cup \{p_f\}, T_r \cup \tilde{T}_r \cup \{\#\}, d_{A'}, p_0, \{p_f\})$, where

$$d_{A'}(q, \#) = p_f \quad \text{for all } q \text{ in } F_A$$

$$d_{A'}(p, a) = d_A(p, a) \quad \text{for all } p \text{ in } Q_0,\ a \text{ in } T_r \cup \tilde{T}_r,$$

we obtain a deterministic finite-state automaton A' which accepts $R\#$.

Now, construct a deterministic pushdown automaton $B = (\{q_0, q_1, q_f\}, T_r \cup \tilde{T}_r \cup \{\#\}, T_r \cup \tilde{T}_r \cup \{Z_0\}, d_B, q_0, Z_0, \{q_f\})$, where

$$d_B(q_0, a_i, Z_0) = (q_1, Z_0 a_i) \quad (1 \leq i \leq r),$$

$$d_B(q_1, a_i, a_j) = (q_1, a_j a_i) \quad (1 \leq i, j \leq r),$$

$$d_B(q_1, \tilde{a}_i, a_i) = (q_1, \lambda) \quad (1 \leq i \leq r),$$

$$d_B(q_1, \lambda, Z_0) = (q_0, Z_0), \qquad d_B(q_0, \#, Z_0) = (q_f, \lambda).$$

It is easily seen that $N(B)$ $(= T(B)) = D_r\#$ holds.

Finally, let a deterministic pushdown automaton C be defined as follows; $C = (\{q_0, q_1, q_f\} \times Q_0, T, K, d_C, (q_0, p_0), Z_0, \{(q_f, p_f)\})$, where

$$T = T_r \cup \tilde{T}_r \cup \{\#\}, \qquad K = T_r \cup \tilde{T}_r \cup \{Z_0\},$$

$$d_C((q,p), a, Z) = ((q', d_{A'}(p, a)), u) \quad \text{if } d_B(q, a, Z) = (q', u).$$

Then it can be checked that, for each (q, p), (q', p') in $\{q_0, q_1, q_f\} \times Q_0$, w in T^*, u, v in K^*,

$$((q,p), w, u) \vdash_C^* ((q', p'), \lambda, v) \text{ iff } (q, w, u) \vdash_B^* (q', \lambda, v) \text{ and } d_{A'}(p, w) = p'.$$

Letting $(q, p) = (q_0, p_0)$, $u = Z_0$, and $v = \lambda$, we have that w is in $N(C)$ iff w is in both $N(B)$ $(= D_r\#)$ and $T(A')$ $(= R\#)$. Hence, $N(C) = (D_r \cap R)\#$ holds and C satisfies the desired condition. □

Lemma 2.3. *Let C be the deterministic pushdown automaton obtained in Lemma 2.2. Then there exist a coding f and a λ-free simple deterministic grammar G such that $N(C) = f(L(G))$.*

Proof. Note that, in a given $C = (Q, T, K, d_C, s_0, Z_0, \{s_f\})$, the final state s_f satisfies condition (ii) in Lemma 2.2, and that the length of u in a-transition $d_C(q, a, A) = (p, u)$ is at most 2. We may assume that

$$(s_0, x, Z_0) \vdash^* (q, \lambda, \lambda) \text{ iff } q = s_f.$$

Now, define a context-free grammar $G = (N, T', P, S_0)$ as follows:

(i) $N = (Q \times K \times Q)$;

(ii) $T' = \{[\lambda, a], [q_1, a], [q_1 q_2, a] \mid q_1, q_2 \text{ in } Q, a \text{ in } T\}$;

(iii) $S_0 = (s_0, Z_0, s_f)$;

(iv) for a in $T \cup \{\lambda\}$, q, p in Q, A, B_1, and B_2 in K,

(1) if $d_C(q, a, A) = (p, B_1 B_2)$, then, for each q_1, q_2 in Q,

$$(q, A, q_2) \rightarrow [q_1 q_2, a](p, B_2, q_1)(q_1, B_1, q_2) \text{ is in } P,$$

(2) if $d_C(q, a, A) = (p, B)$, then, for each q' in Q,

$$(q, A, q') \rightarrow [q', a](p, B, q') \text{ is in } P,$$

(3) if $d_C(q, a, A) = (p, \lambda)$, then $(q, A, p) \rightarrow [\lambda, a]$ is in P.

From the way of constructing P of G and the property (determinism) of C, it can be checked that

(i) G is λ-free, and

(ii) for A' in N, u', v' in N^*, a' in T', $A' \to a'u'$ and $A' \to a'v'$ in P imply $u' = v'$, i.e., G is simple deterministic.

Further define a homomorphism f by

$$f([u, a]) = a \quad \text{for each } [u, a] \text{ in } T'.$$

Then we claim that, for each w in T'^*, x, y in T^*, p, q, q' in Q, and $z = (p_1, B_1, q_1)\dots(p_k, B_k, q_k)$ in N^*,

$$(q, A, q') \Rightarrow_L^n wz \quad \text{and} \quad f(w) = x$$

$$\text{iff } (q, xy, A) \vdash_C^n (p, y, B_k \dots B_1)$$
$$\text{and either } z \text{ is in } H \text{ or } (z = \lambda \text{ and } p = q'),$$

where $H = \{z \text{ in } N^* \mid z = z_1(p, A_1, q)(q', A_2, q'')z_2 \text{ implies } q = q'\}$, i.e., H is the set of strings of triples where the triples are all internally linked between the last component and the next first component. (The claim can be shown by induction on n. Refer to [3, pp. 154–157] for the details.)

Now, let $q = s_0$, $A = Z_0$, $p = q' = s_f$, and $z = y = \lambda$. Then, we have that

$$(s_0, Z_0, s_f) \Rightarrow^n w \quad \text{and} \quad f(w) = x$$

$$\text{iff } (s_0, x, Z_0) \vdash_C^n (s_f, \lambda, \lambda).$$

Thus, $N(C) = f(L(G))$ is obtained. □

Further, we have the following well-known result.

Lemma 2.4. *Any context-free language L can be expressed in the form $t(D_r \cap R)$ for some weak identity t and a regular set R (see, e.g., [3]).*

The series of lemmas above leads to the following main result.

Theorem 2.5. *For any λ-free context-free language L, there effectively exist a weak coding g, a homomorphism h such that $L = gh^{-1}(\cent D_2)$.*

Proof. By Lemma 2.4, we have $L = t(D_r \cap R)$. Further, Lemmas 2.2 and 2.3 tell us that $(D_r \cap R)\# = f(L_0)$, for some coding f and a λ-free simple deterministic language L_0. Now, extend t and define t' as $t'(\#) = \lambda$, $t'(a) = t(a)$ (otherwise); then we have

$$L = t'((D_r \cap R)\#) = t'(f(L_0)).$$

Since it holds true by Lemma 2.1 that $L_0 = k(h^{-1}(¢D_2))$ for some coding k and a homomorphism h, we eventually have

$$L = t'(f(k(h^{-1}(¢D_2)))) = g(h^{-1}(¢D_2)), \quad \text{where } g = t'fk.$$

Thus, $L = gh^{-1}(¢D_2)$, for some weak coding g and homomorphism h, is obtained. □

Corollary 2.6. *For any λ-free context-free language L, there effectively exist a weak coding g, a mapping F such that* $L = gF^{-1}(¢)$, *where F is a composition fh, f is a mapping defined in the proof of Lemma* 2.1, *h is a homomorphism.*

The next result immediately follows from Theorem 2.5 and from the fact that for each recursively enumerable language K there exist an alphabet T, a λ-free simple deterministic (linear) language L on $T^+\bar{T}^+$, and a weak identity g such that $K = g(D \cap L)$, where D is the Dyck set over T (cf. [1]).

Corollary 2.7. *For each recursively enumerable language K there exist an alphabet T, a weak identity g, a coding f and a homomorphism h such that* $K = g(D \cap f(h^{-1}(¢D_2)))$, *where D is the Dyck set over T.*

Corollary 2.8. *For each recursively enumerable language K there exist an alphabet T, a weak identity g, a coding f and a homomorphism h such that* $K = g(h_0^{-1}(D_2) \cap f(h^{-1}(¢D_2)))$, *where* h_0 *is a homomorphism depending on only the size of T.*

This is also obtained from Theorem 2.5 and the fact that, for D over T, there is a homomorphism h_0 such that $D = h_0^{-1}(D_2)$, and h_0 depends on only the size of T.

3. Concluding remarks

We have shown that any λ-free context-free language L can be expressed in the form $gh^{-1}(¢D_2)$, for some weak coding g and homomorphism h, where D_2 is the Dyck set over a two-letter alphabet. Rather recently, Yokomori and Wood [9] showed the following result that is closely related to the above characterization: Let $\mathscr{L}$ be a full principal AFL closed under context-free substitution. Then there is a fixed language L_0 in $\mathscr{L}$ such that for each L in $\mathscr{L}$ there exist a weak coding g and a homomorphism h such that $L = gh^{-1}(L_0)$. Since the proof for the result is constructive, there effectively exists such a fixed language L_0. However, it turns out that L_0 is much more complicated than $¢D_2$. (In fact, L_0 would be an extended nondeterministic version of $¢D_2$.) Thus, our characterization of context-free languages presented here provides a refinement of the result mentioned above.

It is also interesting to compare the main result here with the following, shown (in principle) in [6] or [7], that for any λ-free context-free language L, there exists a finite substitution f such that $L = f^{-1}(¢D_2)$. That is, it may be said that as far as the representation of context-free languages is concerned, the inverse of a finite substitution can be replaced by a composition of an inverse homomorphism and a weak coding.

Acknowledgment

The author wishes to thank Dr. Tosio Kitagawa, the president of IIAS-SIS, Dr. Hajime Enomoto, the director of IIAS-SIS, for their useful suggestions and warm encouragement.

He is also grateful to the referee who not only worked through an earlier draft of this paper but also suggested the proofs of Lemmas 2.2 and 2.3.

References

[1] J. Engelfriet and G. Rozenberg, Fixed point languages, equality languages, and representation of recursively enumerable languages, *J. Assoc. Comput. Mach.* 27 (1980) 499–518.
[2] S.A. Greibach, The hardest context-free languages, *SIAM J. Comput.* 2 (1973) 304–310.
[3] M.A. Harrison, *Introduction to Formal Language Theory* (Addison-Wesley, Reading, MA, 1978).
[4] J. Karhumäki and M. Linna, A note on morphic characterization of languages, *Discrete Appl. Math.* 5 (1983) 243–246.
[5] M. Latteux and J. Leguy, On the composition of morphisms and inverse morphisms, in: Lecture Notes in Computer Science 154 (Springer, Berlin, 1983) 420–432.
[6] A. Salomaa, *Jewels of Formal Language Theory* (Computer Science Press, Rockville, MD, 1981).
[7] E. Shamir, A representation theorem for algebraic and context-free power series in noncommuting variables, *Inform. Control* 11 (1967) 239–254.
[8] P. Turakainen, A homomorphic characterization of principal semiAFLs without using intersections with regular sets, *Inform. Sci.* 27 (1982) 141–149.
[9] T. Yokomori and D. Wood, An inverse homomorphic characterization of full principal AFL, *Inform. Sci.* 33 (1984) 209–215.