Publ. RIMS, Kyoto Univ.
17 (1981), 267-296

Left Transformation Languages

By

Hidetaka TANAKA*

Abstract

A subfamily LFT of the family CS of context-sensitive languages is investigated. Elements of LFT are called left transformation languages. Any context-sensitive language is represented in the form $h(L_1 \cap L_2^R)$ where L_1, $L_2 \in$ LFT, h is a length-preserving homomorphism, and L_2^R is the mirror image of L_2. This fact implies that the "LBA-problem" is equivalent to the problem whether LFT can be accepted by deterministic linear bounded automata. It is remarkable that LFT is an intersection-closed AFL and the emptiness problem for LFT is effectively solvable. Also LFT is shown to have the quasi-prefix property, and we can see that LFT and the family CF of context-free languages are incomparable.

§ 1. Introduction

The LBA-problem is an important problem yet open in formal language theory. This problem is concerned with the inclusion relation between the family CS of context-sensitive languages and the family DCS of deterministic context-sensitive languages (i.e., languages accepted by deterministic linear bounded automata). To consider the problem this paper introduces and investigates a subfamily LFT of CS, elements of which are called left transformation languages. The importance of the family LFT is assured by the fact that any context-sensitive language can be represented in the form $h(L_1 \cap L_2^R)$ where L_1 and L_2 are left transformation languages, h is a length-preserving homomorphism, and L_2^R is the mirror image of L_2. Because this fact implies that the LBA-problem is equivalent to the problem whether LFT is included in DCS or not.

The equivalence of context-sensitive grammars and one-sided ones was proved by Penttonen [1]. He simulated derivations of context-

Communicated by S. Takasu, April 1, 1980.
* Department of Information Science, Faculty of Science, University of Tokyo, Bunkyo-ku, Tokyo 113, Japan.

sensitive grammars by left context-sensitive productions in two stages. The first stage is executed by applications of length-increasing linear context-free productions, and the second stage is executed by applications of length-preserving left context-sensitive productions. He called the second stage a length-preserving left context-sensitive transformation. We are concerned with this transformation in this paper.

In Section 2 we state the formal definitions of length-preserving left context-sensitive transformation (abbreviation: llcst) and left transformation language (abbreviation: ltl), and show some examples. LLCS is defined to be the family of domains of llcst's, and for each family $\mathcal{L}$ of languages LLCST($\mathcal{L}$) is the family of languages obtained by applying llcst's to elements of $\mathcal{L}$. Then the family LFT of ltl's is defined to be LLCST(REG), where REG is the family of regular languages.

Section 3 provides representation theorems for LLCST($\mathcal{L}$), LFT, and CS. Also we show some closure properties of LLCS and LFT under operations. LFT is an intersection-closed AFL and it equals to the least semi-AFL containing LLCS.

In Section 4 we introduce a new type of automaton called traverse automaton (abbreviation: tra). For each tra M the language $L(M)$ accepted by M and the language $A(M)$ associated with M are defined. $L(M)$ is a simple ltl and $A(M)$ is an ltl, and so in later sections we study LFT by means of tra's. Especially traverses of a tra M, which are finite sequences of words of a special form, play an important role.

In Section 5 we prove elementary lemmas related to traverses of tra's. These lemmas are lately used in proofs of main results of this paper.

One of main results of this paper is that the emptiness problem for ltl's is effectively solvable. This is proved in Section 6 by using the result of Haines [3] which is concerned with the partial order of words by embedding. Since LFT is an intersection-closed AFL incomparable with the family CF of context-free languages, this result gives us another reason to study LFT.

In Section 7 we prove that LFT satisfies the quasi-prefix property and show some examples of languages not in LFT. This implies that LFT and CF are incomparable. Also we can show non-closure properties

of LFT under some operations. These assures us that LFT is a small subfamily of CS.

§ 2. Left Transformation Language

In this section we introduce the definitions of length-preserving left context-sensitive transformation (llcst), left transformation language (ltl), and domain language of llcst. These are objects of this paper.

An *alphabet* is a nonempty finite set Σ, whose elements are called symbols. By Σ^* we denote the free monoid generated by Σ. Elements of Σ^* are called *words*. The identity of Σ^* is called the *empty word* and denoted by λ. Subsets of Σ^* are called *languages*. Λ stands for the language $\{\lambda\}$. For a language L, L^* denotes the submonoid of Σ^* generated by L, which is called the *Kleene closure* of L. The *λ-free Kleene closure* of L, in symbols L^+, is defined by $L^+ = L^* - \Lambda$.

The concept of llcst was first introduced by Penttonen [1]. Our definition differs from the original one, but this alternation was shown not to change the ability (Lemma 1 of [1]).

Definition 2. 1. A *length-preserving left context-sensitive transformation* (abbreviation: *llcst*) is a quadruple $T = (V, \Sigma, Q, P)$ where

(i) V, Σ, and Q are alphabets with $V \supset \Sigma \cup Q$,

(ii) P is a set of productions whose forms are $A \to B$ or $AB \to AC$ where A, B, and $C \in V$.

Σ and Q are called the *input alphabet* and the *output alphabet* of T resp. We usually denote V, Σ, Q, and P by V_T, Σ_T, Q_T, and P_T resp. The binary relation $\Rightarrow_T$ on V^* is defined as the usually way. Formally if $u, v \in V^*$ and $x \to y \in P$, then we write $uxv \Rightarrow_T uyv$. Let $\overset{*}{\Rightarrow_T}$ denote the reflexive and transitive closure of $\Rightarrow_T$. We define a relation from Σ^* into Q^*, also denoted by T, as follows. For each word w in Σ^*,

$$T(w) = \{x \in Q^* \mid w \overset{*}{\Rightarrow_T} x\}.$$

The relation T is called an *llcst relation*. The domain and range of T, in symbols $D(T)$ and $R(T)$ resp., are defined by

$$D(T) = \{w \in \Sigma^* \mid T(w) \neq \emptyset\},$$

$$R(T) = T(\Sigma^*).$$

We note that the inverse T^{-1} of the relation T is also an llcst relation. For T^{-1} is defined by an llcst T':

$$T' = (V, Q, \Sigma, \{y \to x | x \to y \in P\}).$$

Obviously $R(T^{-1}) = D(T)$ and $D(T^{-1}) = R(T)$.

For each language L over Σ, let

$$T(L) = \bigcup_{w \in L} T(w),$$

and for each family $\mathcal{L}$ of languages, let

$$\text{LLCST}(\mathcal{L}) = \{T(L) | L \in \mathcal{L} \text{ and } T \text{ is an llcst}\}.$$

By LLCS we denote the family of domains of llcst relations.

For well-known families of languages we use the following notations.

RE: recursively enumerable languages,

CS: context-sensitive languages,

DCS: deterministic context-sensitive languages,

CF: context-free languages,

LIN: linear context-free languages,

REG: regular languages.

The following proposition proved by Penttonen [1] is the starting point of this paper.

Proposition 2.2 (Penttonen). $\text{CS} = \text{LLCST}(\text{LIN})$. *Especially there exists a linear context-free language L_0 in the least semi-*AFL *containing $\{a^n b^n | n \geqq 1\}$ such that*

$$\text{CS} = \text{LLCST}(L_0) \cup \text{LLCST}(L_0 \cup \Lambda).$$

Now we define left transformation language.

Definition 2.3. Let $\text{LFT} = \text{LLCST}(\text{REG})$. A *left transformation language* (abbreviation: *ltl*) is an element of LFT. Trivially LFT is a subfamily of CS. The close relation between CS and LFT shall be shown in the next section. Also we can see that LFT is the least

semi-AFL containing LLCS.

In the rest of this section we show some examples of languages in LLCS. First we note the following fact.

Lemma 2.4. *For each language L, define*

$$Pref(L) = \{u | uv \in L \text{ for some } v\}.$$

For any language L in LLCS, $L = Pref(L)^*$.

Proof. Assume $L = D(T)$. Since T is left context-sensitive, in a derivation $xy \Rightarrow^* uv$ with $|x| = |u|$ the transformation of x to u is independent upon y. Therefore $xy \in D(T)$ implies $x \in D(T)$, and $L = Pref(L)$. On the other hand, if $x \Rightarrow^* u$ and $y \Rightarrow^* v$, then $xy \Rightarrow^* uv$ by the definition of $\Rightarrow$. Therefore $x, y \in D(T)$ implies $xy \in D(T)$, and $L = L^*$. Hence

$$L = L^* = Pref(L)^*. \qquad \text{Q.E.D.}$$

Example 2.5. For any regular language R, $Pref(R)^* \in$ LLCS. For assume that a regular grammar $G = (V_N, \Sigma, X_0, P)$ generates $R - \Lambda$. Elements of P are of the form $X \to aY$ or $X \to a$ where X, $Y \in V_N$ and $a \in \Sigma$. An llcst T with $D(T) = Pref(R)^*$ can be defined as follows.

(i) $T = (V_N \cup \Sigma, \Sigma, Q, P')$,

(ii) $Q = \{X \in V_N | X \Rightarrow_G^* w \text{ for some } w \text{ in } \Sigma^*\}$,

(iii) P' consists of the following productions.

(1) If $X_0 \to a \in P$, then $a \to X_0 \in P'$,

(2) if $X_0 \to aX \in P$, then $a \to X \in P'$,

(3) if $X \to a \in P$, then $Xa \to XX_0 \in P'$,

(4) if $X \to aY \in P$, then $Xa \to XY \in P'$.

To show other examples, we introduce flow diagram representing derivations of llcst's. In derivations, when a production $AB \to AC$ is applied, then we write it as in Figure 1(a), and when a production $A \to B$ is applied, it is written as in Figure 1(b). For instance, if $XY \to Xa$, $aZ \to ab$, $Xa \to Xb$, and $X \to c$ are productions, then the derivation:

$$XYZ \Rightarrow XaZ \Rightarrow Xab \Rightarrow Xbb \Rightarrow cbb$$

can be represented as in Figure 1(c).

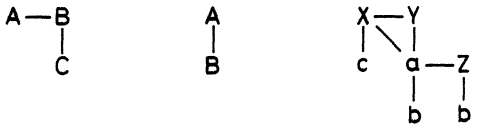

Figure 1 (a) (b) (c)

We note that the length of a derivation is equal to the number of vertical lines in the corresponding flow diagram. From flow diagrams one can construct derivations corresponding to flow diagrams.

Example 2.6. $\{a^n b^m | n \geqq m \geqq 0\}^* \in \mathrm{LLCS}$. We define an llcst $T = (V, \{a, b\}, Q, P)$ by

$$V = \{a, b, A, A', B, B'\},$$

$$Q = \{a, A, A', B, B'\},$$

$$P = \{aa \to aA, ab \to aB, AA \to AA', A'A' \to A'A, AB \to AB',$$

$$A'B' \to A'B, B'b \to B'B, BB \to BB', B'B' \to B'B\}.$$

Then $D(T) = Pref(\{a^n b^n | n \geqq 0\})^* = \{a^n b^m | n \geqq m \geqq 0\}^*$. For instance the flow diagram for $a^5 b^5 \in D(T)$ is shown in Figure 2.

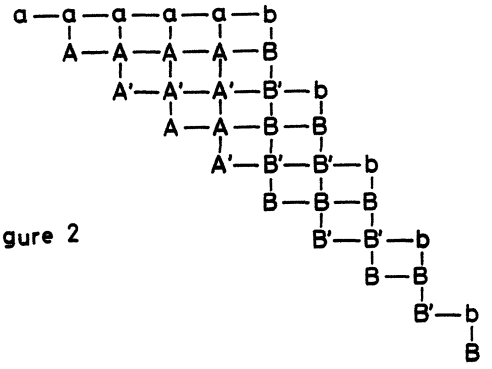

Figure 2

Example 2.7. The notation $N_c(w)$ means the number of occurrences of a symbol c in a word w. Let

$$L=\{w\in\{a,b\}^*|\text{ for any } x \text{ in } Pref(w),\ N_a(x)\geqq N_b(x)\}.$$

We define an llcst $T=(\{a,b,A\},\{a,b\},\{a,A\},P)$ by

$$P=\{aa\to aA,\ ab\to aA,\ AA\to Aa\}.$$

Then we have

$$D(T)=Pref(D_1)^*=Pref(D_1)=L\in\text{LLCS},$$

where D_1 is a 1-Dyck language over $\{a,b\}$. For instance the flow diagram for $a^3ba^2b^2ab^3\in D(T)$ is shown in Figure 3.

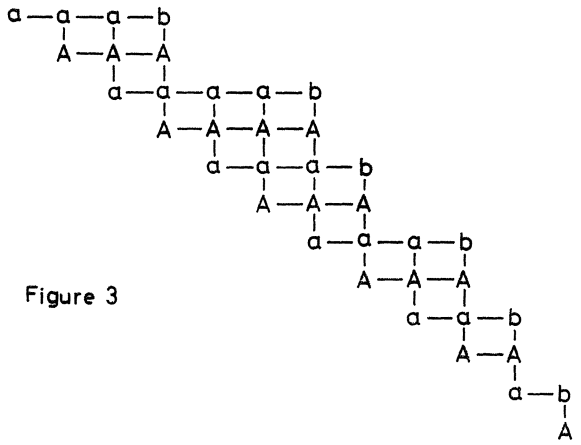

Figure 3

Example 2.8. $L=\{a^nb^m|2^n\geqq m\geqq 0\}^*\in$LLCS. An llcst T with $D(T)=L$ can be defined by

$$T=(\{a,b,A_1,A_2,B_1,B_2,D_1,D_2\},\{a,b\},\{A_1,A_2,D_1,D_2\},P),$$

$$P=\{a\to A_1,\ ab\to aD_1,\ A_1A_1\to A_1B_1,\ A_1D_1\to A_1D_2,$$

$$B_1\to A_2,\ B_1A_2\to B_1B_2,\ A_2A_1\to A_2B_1,\ B_2A_2\to B_2B_2,$$

$$B_2\to A_1,\ B_1D_2\to B_1D_1,\ B_2D_2\to B_2D_1,\ D_2b\to D_2D_1,$$

$$A_1D_1\to A_1D_2,\ A_2D_1\to A_2D_2,\ D_1D_1\to D_1D_2,\ D_2D_2\to D_2D_1\}.$$

For instance the flow diagram for $a^3b^3\in D(T)$ is shown in Figure 4.

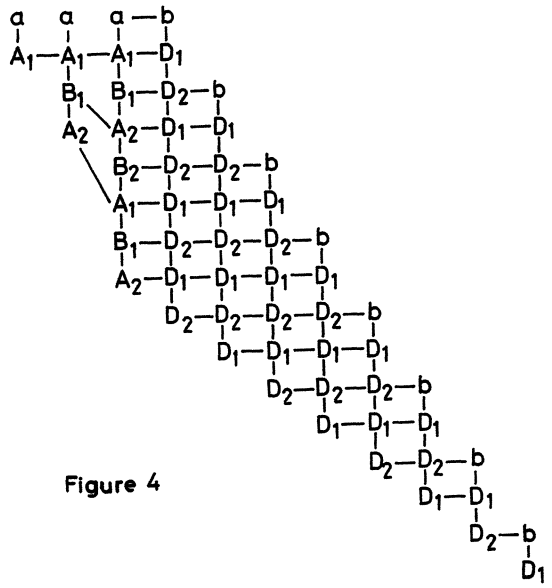

Figure 4

We note that in Examples 2.5–2.8 the length of a derivation from $w \in D(T)$ with $|w| = n$ is at most n^2. In the next example an llcst T makes very wasteful moves and the length of a derivation is exponential.

Example 2.9. Define an llcst $T = (V, \{a, b\}, Q, P)$ by

$$V = \{a, b, A_1, A_2, B_1, B_2, C_1, C_2, D_1, D_2\},$$

$$Q = \{A_1, A_2, D_1, D_2\},$$

$$P = \{a \to A_1,\ A_1 \to B_1,\ B_1 \to A_2,\ A_2 \to B_2,\ B_2 \to A_1,$$

$$A_1 b \to A_1 D_1,\ B_1 D_1 \to B_1 C_1,\ A_2 C_1 \to A_2 D_2,\ B_2 D_2 \to B_2 C_2,$$

$$A_1 C_2 \to A_1 D_1,\ D_2 b \to D_2 D_1,\ D_1 D_1 \to D_1 C_1,\ D_2 C_1 \to D_2 D_2,$$

$$D_1 D_2 \to D_1 C_2,\ D_2 C_2 \to D_2 D_1\}.$$

Then $D(T) = (ab^*)^*$. For instance the flow diagram for $ab^4 \in D(T)$ is shown in Figure 5. We note that for each n the length of the derivation for ab^n is at least $2^{n+1} + 2^n - n - 3$.

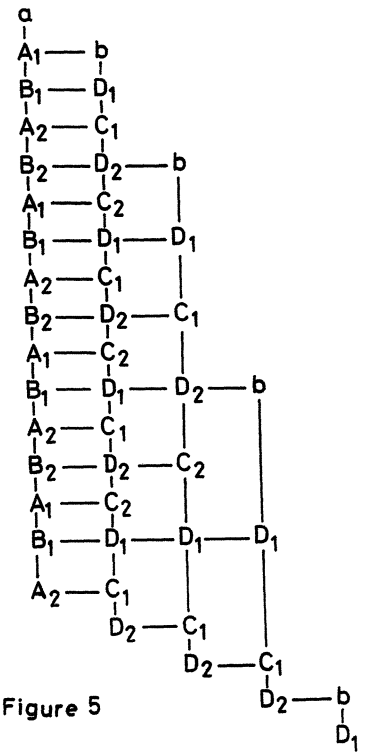

Figure 5

§ 3. Representation Theorems

In this section first we give a representation theorem for LLCST($\mathcal{L}$), and next we show some closure properties of LLCS and LFT under operations. Then we can see that LFT is an intersection-closed AFL and also it is equal to the least semi-AFL containing LLCS. Finally we show two representation theorems for CS. These theorems assure us that LFT is an important subfamily of CS.

A homomorphism h of monoids from Σ^* to Δ^* is called a homomorphism over Σ. It is called *length-preserving* iff for all a in Σ $|h(a)|=1$.

Notation 3. 1. For each family $\mathcal{L}$ of languages, let

$$H^{-1}(\mathcal{L}) = \{h^{-1}(L) \mid L \in \mathcal{L} \text{ and } h \text{ is a homomorphism}\},$$

$$H_p^{-1}(\mathcal{L}) = \{h^{-1}(L) \mid L \in \mathcal{L} \text{ and } h \text{ is length-preserving}\},$$

$$H_p(\mathcal{L}) = \{h(L) \mid L \in \mathcal{L} \text{ and } h \text{ is length-preserving}\},$$

and for families $\mathcal{L}_1$ and $\mathcal{L}_2$ of languages, let

$$\mathcal{L}_1 \wedge \mathcal{L}_2 = \{L_1 \cap L_2 \mid L_1 \in \mathcal{L}_1 \text{ and } L_2 \in \mathcal{L}_2\}.$$

Lemma 3.2. $\mathrm{LLCST}(\mathcal{L}) \subset H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{LLCS})$.

Proof. Assume $L \in \mathcal{L}$ and T is an llcst (V, Σ, Q, P). Define length-preserving homomorphisms $f: (\Sigma \times Q)^* \to \Sigma^*$ and $g: (\Sigma \times Q)^* \to Q^*$ by $f((X,a)) = X$ and $g((X, a)) = a$ for each (X, a) in $\Sigma \times Q$. Then

$$T(L) = g(f^{-1}(L) \cap S(T)),$$

where $S(T)$ is a language over $\Sigma \times Q$ defined by

$$S(T) = \{w \in (\Sigma \times Q)^* \mid f(w) \Rightarrow^* g(w)\}.$$

Thus to prove the lemma, it suffices to show $S(T) \in \mathrm{LLCS}$. Now define a new llcst $T' = (V \times Q, \Sigma \times Q, Q', P')$ by

(i) $Q' = \{(a, a) \mid a \in Q\}$,

(ii) P' consists of the following productions.

(1) If $A \to B \in P$, then $(A, a) \to (B, a) \in P'$ for all a in Q,

(2) if $AB \to AC \in P$, then $(A, a)(B, b) \to (A, a)(C, b)$ is in P' for all a, b in Q.

Clearly $D(T') = S(T)$, and the lemma holds. Q.E.D.

A *renaming* is an injective length-preserving homomorphism. If $\mathcal{L}$ is closed under renaming, we have a representation theorem for $\mathrm{LLCST}(\mathcal{L})$.

Theorem 3.3. *If $\mathcal{L}$ is closed under renaming,*

$$\mathrm{LLCST}(\mathcal{L}) = H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{LLCS}).$$

Proof. By Lemma 3.2 it suffices to show the inclusion:

$$\mathrm{LLCST}(\mathcal{L}) \supset H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{LLCS}).$$

Assume $f: \Sigma^* \to \Delta^*$ and $g: \Sigma^* \to \Gamma^*$ are length-preserving homomorphism, $L \in \mathcal{L}$ with $L \subset \Delta^*$, and $T = (V, \Sigma, Q, P)$ is an llcst. We show

$$g(f^{-1}(L) \cap D(T)) \in \mathrm{LLCST}(\mathcal{L}).$$

Since $\mathcal{L}$ is closed under renaming, without loss of generality we can assume $\Delta \cap \Gamma = \emptyset$. Define a new llcst $T' = (\Delta \cup \Gamma \cup V \times \Gamma, \Delta, \Gamma, P')$ as follows.

(1) If $a \in \Delta$, $X \in \Sigma$, and $f(X) = a$, then $a \to (X, g(X)) \in P'$.

(2) If $A \to B \in P$, then $(A, c) \to (B, c) \in P'$ for all c in Γ.

(3) If $AB \to AC \in P$, then $(A, c)(B, d) \to (A, c)(C, d) \in P'$ for all c, d in Γ.

(4) If $A \in Q$, then $(A, c) \to c \in P'$ for all c in Γ.

Clearly $T'(L) = g(f^{-1}(L) \cap D(T))$ which establishes the theorem.

Remark. If $\mathcal{L}$ is not closed under renaming, then the equation above does not necessarily hold. For example let $\mathcal{L} = \{a^*\}$. Since $H_p^{-1}(\mathcal{L})$ contains $\{a, b\}^*$, by Example 2. 6,

$$H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{LLCS}) \supset \{a^n b^m \mid n \geqq m \geqq 0\}^*.$$

But this language L is not in $\mathrm{LLCST}(\mathcal{L})$. For assume $T(a^*) = L$ for some llcst T. Since T is length-preserving, $a^6 \Rightarrow^* a^3 b^3$ and $a^3 \Rightarrow^* a^2 b$. Then $a^6 \Rightarrow^* a^3 b^3 \Rightarrow^* a^2 b b^3 = a^2 b^4 \in T(a^*)$. This is a contradiction.

To state a corollary of the theorem above, we recall the definitions of semi-AFL and AFL. A homomorphism h over Σ is called *λ-free* iff $h(a) \neq \lambda$ for any a in Σ. A family of languages is called a *semi-AFL* iff it is closed under union, λ-free homomorphism, inverse homomorphism, and intersection with regular languages. An AFL is a semi-AFL closed under λ-free Kleene closure. For instance RE, CS, DCS, CF, and REG are AFL's. LIN is a semi-AFL but it is not an AFL. Later we shall show that LFT is an AFL closed under intersection. We note that LLCS is not a semi-AFL, because it is not closed under λ-free homomorphism or union.

Corollary 3. 4. *If $\mathcal{L}$ is a semi-*AFL, *then*

$$\mathrm{LLCST}(\mathcal{L}) = H_p(\mathcal{L} \wedge \mathrm{LLCS}) = H_p(\mathcal{L} \wedge \mathrm{LFT}).$$

Proof. Since a semi-AFL is closed under inverse homomorphism,

$$\mathrm{LLCST}(\mathcal{L}) = H_p(\mathcal{L} \wedge \mathrm{LLCS}).$$

Also REG is an AFL, $\mathrm{LFT} = H_p(\mathrm{REG} \wedge \mathrm{LLCS})$. Then

$$\begin{aligned}\mathrm{LLCST}(\mathcal{L}) &\subset H_p(\mathcal{L} \wedge \mathrm{LFT}) \\ &= H_p(\mathcal{L} \wedge H_p(\mathrm{REG} \wedge \mathrm{LLCS})) \\ &\subset H_p(H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{REG} \wedge \mathrm{LLCS})) \\ &= H_p(\mathcal{L} \wedge \mathrm{LLCS}) = \mathrm{LLCST}(\mathcal{L}).\end{aligned}$$

This completes the proof. Q.E.D.

Next we consider the closure properties of LLCS under operations. It is clear that LLCS is closed under length-preserving homomorphism, but it is not closed under λ-free homomorphism or union.

Lemma 3.5. $H^{-1}(\mathrm{LLCS}) = \mathrm{LLCS}$.

Proof. Assume $T = (V, \Sigma, Q, P)$ is an llcst and $h: \Delta^* \to \Sigma^*$ is a homomorphism. We show $h^{-1}(D(T)) \in \mathrm{LLCS}$. Since LLCS is closed under renaming, without loss of generality we can assume $\Delta \cap \Sigma = \emptyset$. Let

$$k = \max\{|h(a)| \mid a \text{ in } \Delta\},$$

$$W = \{w \in V^* \mid |w| \leqq k\}.$$

Define a new llcst $T' = (V', \Delta, Q', P')$ as follows.

(i) $V' = \Delta \cup \{(i, w) \mid i = 0, 1 \text{ and } w \in W\}$,

(ii) $Q' = \{(0, w) \mid w \in W \cap Q^*\}$,

(iii) P' consists of the following productions. Let $x, y, z \in W$.

(1) $a \to (0, h(a)) \in P'$ for all a in Δ.

(2) If $x \Rightarrow_T y$, then $(0, x) \to (0, y) \in P'$.

(3) If $xy \Rightarrow_T xz$, then $(i, x)\,(0, y) \to (i, x)\,(0, z)$ is in P' for $i = 0, 1$.

(4) $(i, x)\,(0, \lambda) \to (i, x)\,(1, x) \in P'$ for $i = 0, 1$.

(5) $(1, x) \to (0, \lambda) \in P'$.

Then we have $D(T') = h^{-1}(D(T))$. Q.E.D.

Lemma 3. 6. LLCS *is closed under intersection.*

Proof. Assume $T_1=(V_1, \Sigma, Q_1, P_1)$ and $T_2=(V_2, \Sigma, Q_2, P_2)$ are llcst's. We show $D(T_1) \cap D(T_2) \in$LLCS. Define a new llcst T as follows. $T=(V_1 \times V_2 \cup \Sigma, \Sigma, Q_1 \times Q_2, P)$ and

(i) if $A \to B \in P_1$, then $(A, X) \to (B, X) \in P$ for all X in V_2,

(ii) if $AB \to AC \in P_1$, then $(A, X)(B, Y) \to (A, X)(C, Y) \in P$ for all X, Y in V_2,

(iii) if $X \to Y \in P_2$, then $(A, X) \to (A, Y) \in P$ for all A in V_1,

(iv) if $XY \to XZ \in P_2$, then $(A, X)(B, Y) \to (A, X)(B, Z) \in P$ for all A, B in V_1,

(v) $a \to (a, a) \in P$ for all a in Σ.

Clearly we have $D(T)=D(T_1) \cap D(T_2)$. Q.E.D.

Theorem 3. 7. LLCS *is closed under length-preserving homomorphism, inverse homomorphism, intersection, and*

$$\text{LLCST(LLCS)} = \text{LLCS}.$$

But LLCS *is not closed under union, λ-free homomorphism, intersection with regular languages, or reversal.*

Proof. Clearly LLCS is closed under length-preserving homomorphism, and by Lemmas 3. 5 and 3. 6, it is closed under inverse homomorphism and intersection. By Theorem 3. 3,

$$\text{LLCS} \subset \text{LLCST(LLCS)} = H_p(H_p^{-1}(\text{LLCS}) \wedge (\text{LLCS}))$$
$$= H_p(\text{LLCS} \wedge \text{LLCS}) = H_p(\text{LLCS}) = \text{LLCS},$$

and hence LLCST(LLCS) = LLCS. $a^* \in$LLCS but $(a^2)^* \notin$LLCS. Therefore LLCS is not closed under λ-free homomorphism. Also since $a^* \cup b^*$ is not in LLCS, LLCS is not closed under union or intersection with regular languages. By Example 2. 6,

$$L=\{a^n b^m \mid n \geqq m \geqq 0\}^* \in \text{LLCS},$$

but by Lemma 2. 4 the reversal L^R of L is not in LLCS. Q.E.D.

Now we consider closure properties of LLCST($\mathcal{L}$) and LFT.

Lemma 3.8. *If $\mathcal{L}$ is closed under renaming and union, then*

(i) $H_p^{-1}(\mathrm{LLCST}(\mathcal{L})) = H_p(\mathrm{LLCST}(\mathcal{L})) = \mathrm{LLCST}(\mathcal{L})$,

(ii) $\mathrm{LLCST}(\mathcal{L})$ *is closed under union,*

(iii) $\mathrm{LLCST}(\mathcal{L}) \wedge \mathrm{LLCS} = \mathrm{LLCST}(\mathcal{L})$,

(iv) $\mathrm{LLCST}(\mathrm{LLCST}(\mathcal{L})) = \mathrm{LLCST}(\mathcal{L})$.

Proof. (i) and (ii) are easily seen.

$$\begin{aligned}\mathrm{LLCST}(\mathcal{L}) &\subset \mathrm{LLCST}(\mathcal{L}) \wedge \mathrm{LLCS} = H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{LLCS}) \wedge \mathrm{LLCS} \\ &\subset H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{LLCS} \wedge H_p^{-1}(\mathrm{LLCS})) \\ &= H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{LLCS} \wedge \mathrm{LLCS}) = H_p(H_p^{-1}(\mathcal{L}) \wedge \mathrm{LLCS}) \\ &= \mathrm{LLCST}(\mathcal{L})\end{aligned}$$

and (iii) holds. Then

$$\begin{aligned}\mathrm{LLCST}(\mathcal{L}) &\subset \mathrm{LLCST}(\mathrm{LLCST}(\mathcal{L})) \\ &= H_p(H_p^{-1}(\mathrm{LLCST}(\mathcal{L})) \wedge \mathrm{LLCS}) \\ &= H_p(\mathrm{LLCST}(\mathcal{L}) \wedge \mathrm{LLCS}) \\ &= H_p(\mathrm{LLCST}(\mathcal{L})) = \mathrm{LLCST}(\mathcal{L})\end{aligned}$$

and (iv) holds. Q.E.D.

Theorem 3.9. *If $\mathcal{L}$ is a semi-*AFL (AFL), *so is* $\mathrm{LLCST}(\mathcal{L})$ *and*

(i) $\mathrm{LLCST}(\mathcal{L}) \wedge \mathrm{LFT} = \mathrm{LLCST}(\mathcal{L})$,

(ii) $\mathrm{LLCST}(\mathrm{LLCST}(\mathcal{L})) = \mathrm{LLCST}(\mathcal{L})$.

Furthermore if $\mathcal{L}$ is closed under intersection, so is $\mathrm{LLCST}(\mathcal{L})$.

Proof. Assume $\mathcal{L}$ is a semi-AFL. Note that $\mathcal{L} \wedge \mathrm{LLCS}$ is closed under inverse homomorphism. For

$$\mathcal{L} \wedge \mathrm{LLCS} \subset H^{-1}(\mathcal{L} \wedge \mathrm{LLCS}) \subset H^{-1}(\mathcal{L}) \wedge H^{-1}(\mathrm{LLCS}) = \mathcal{L} \wedge \mathrm{LLCS}.$$

Then we have

$$\begin{aligned}\mathrm{LLCST}(\mathcal{L}) &= H_p(\mathcal{L} \wedge \mathrm{LLCS}) = H_p(\mathcal{L} \wedge \mathrm{LLCS} \wedge \mathrm{REG}) \\ &= H_p(H^{-1}(\mathcal{L} \wedge \mathrm{LLCS}) \wedge \mathrm{REG}).\end{aligned}$$

Recall the representation theorem for principal semi-AFL proved by Ginsburg et al [2]. This theorem says that for each nonempty language L the least semi-AFL containing L is represented in the form:

$$H_p(H^{-1}(L)\cap\mathrm{REG}).$$

By this theorem $\mathrm{LLCST}(\mathcal{L})$ is the union of principal semi-AFL's generated by languages in $\mathcal{L}\cap\mathrm{LLCS}$. Hence $\mathrm{LLCST}(\mathcal{L})$ is closed under λ-free homomorphism, inverse homomorphism, and intersection with regular languages. By (ii) of Lemma 3.8 $\mathrm{LLCST}(\mathcal{L})$ is closed under union. Consequently $\mathrm{LLCST}(\mathcal{L})$ is a semi-AFL.

It is easily proved that if $\mathcal{L}$ is an AFL, then $\mathrm{LLCST}(\mathcal{L})$ is closed under λ-free Kleene closure, and $\mathrm{LLCST}(\mathcal{L})$ is an AFL.

(ii) is implied by (iv) of Lemma 3.8. And by Corollary 3.4, (ii) implies (i).

Finally assume that $\mathcal{L}$ is a semi-AFL closed under intersection.

$$\begin{aligned}
\mathrm{LLCST}(\mathcal{L}) &\subset \mathrm{LLCST}(\mathcal{L})\cap\mathrm{LLCST}(\mathcal{L}) \\
&= H_p(\mathcal{L}\cap\mathrm{LLCS})\cap\mathrm{LLCST}(\mathcal{L}) \\
&\subset H_p(\mathcal{L}\cap\mathrm{LLCS}\cap H_p^{-1}(\mathrm{LLCST}(\mathcal{L}))) \\
&= H_p(\mathcal{L}\cap\mathrm{LLCS}\cap\mathrm{LLCST}(\mathcal{L})) = H_p(\mathcal{L}\cap\mathrm{LLCST}(\mathcal{L})) \\
&= H_p(\mathcal{L}\cap H_p(\mathcal{L}\cap\mathrm{LLCS})) \\
&\subset H_p(H_p(H_p^{-1}(\mathcal{L})\cap\mathcal{L}\cap\mathrm{LLCS})) \\
&= H_p(H_p^{-1}(\mathcal{L})\cap\mathcal{L}\cap\mathrm{LLCS}) = H_p(\mathcal{L}\cap\mathcal{L}\cap\mathrm{LLCS}) \\
&= H_p(\mathcal{L}\cap\mathrm{LLCS}) = \mathrm{LLCST}(\mathcal{L}),
\end{aligned}$$

and $\mathrm{LLCST}(\mathcal{L})$ is closed under intersection. Q.E.D.

Theorem 3.10. LFT *is an* AFL *closed under intersection and* λ*-free substitution. Also* LFT *is equal to the least semi-*AFL *containing* LLCS.

Proof. $\mathrm{LFT}=\mathrm{LLCST}(\mathrm{REG})$ and REG is an AFL closed under intersection. Hence LFT is an AFL closed under intersection. Since an intersection-closed AFL is closed under λ-free substitution, so is LFT.

The latter statement is clear, because

$$\mathrm{LFT}=\mathrm{LLCST}(\mathrm{REG})=H_p(\mathrm{REG}/\backslash\mathrm{LLCS}). \qquad \text{Q.E.D.}$$

Finally in this section we show two representation theorems for CS. The first one is directly implied by Proposition 2. 2 and Theorem 3. 3.

Theorem 3. 11. $\mathrm{CS}=H_p(\mathrm{LIN}/\backslash\mathrm{LLCS})=H_p(\mathrm{LIN}/\backslash\mathrm{LFT})$.

For each word w, w^R denotes the reversal (mirror image) of w. For each language L and family $\mathcal{L}$, let

$$L^R=\{w^R \mid w\in L\},$$
$$\mathcal{L}^R=\{L^R \mid L\in\mathcal{L}\}.$$

Then LLCS^R is equal to the family of domains of length-preserving "right" context-sensitive transformations, and LFT^R is equal to the family of "right" transformation languages.

Theorem 3. 12. $\mathrm{CS}=H_p(\mathrm{REG}/\backslash\mathrm{LLCS}/\backslash\mathrm{LLCS}^R)=H_p(\mathrm{LFT}/\backslash\mathrm{LFT}^R)$.

Proof. Since LFT is an AFL, LFT^R is also an AFL. Then

$$\begin{aligned}
H_p(\mathrm{LFT}/\backslash\mathrm{LFT}^R) &= \mathrm{LLCST}(\mathrm{LFT}^R)=\mathrm{LLCST}((H_p(\mathrm{REG}/\backslash\mathrm{LLCS}))^R)\\
&=\mathrm{LLCST}(H_p(\mathrm{REG}^R/\backslash\mathrm{LLCS}^R))\\
&=\mathrm{LLCST}(H_p(\mathrm{REG}/\backslash\mathrm{LLCS}^R))\\
&=H_p(H_p(\mathrm{REG}/\backslash\mathrm{LLCS}^R)/\backslash\mathrm{LLCS})\\
&\subset H_p(H_p(\mathrm{REG}/\backslash\mathrm{LLCS}^R/\backslash H_p^{-1}(\mathrm{LLCS})))\\
&=H_p(\mathrm{REG}/\backslash\mathrm{LLCS}^R/\backslash\mathrm{LLCS})\subset H_p(\mathrm{LFT}/\backslash\mathrm{LFT}^R).
\end{aligned}$$

Therefore

$$\mathrm{CS}\supset H_p(\mathrm{REG}/\backslash\mathrm{LLCS}/\backslash\mathrm{LLCS}^R)=H_p(\mathrm{LFT}/\backslash\mathrm{LFT}^R)=\mathrm{LLCST}(\mathrm{LFT}^R),$$

and by Theorem 3. 9 $\mathrm{LLCST}(\mathrm{LFT}^R)$ is an AFL. To complete the proof, it suffices to show that the linear language L_0 in Proposition 2. 2

is in $H_p(\mathrm{REG}/\backslash\mathrm{LLCS}/\backslash\mathrm{LLCS}^R)$, because

$$\mathrm{LLCST}(\mathrm{LLCST}(\mathrm{LFT}^R)) = \mathrm{LLCST}(\mathrm{LFT}^R).$$

By Example 2.6,

$$\{a^n b^m | n \geqq m \geqq 0\}^* \in \mathrm{LLCS} \quad \text{and} \quad \{a^n b^m | 0 \leqq n \leqq m\}^* \in \mathrm{LLCS}^R.$$

Therefore $L = \{a^n b^n | n \geqq 0\}^* \in \mathrm{LLCS}/\backslash\mathrm{LLCS}^R$. Since L_0 is in the least semi-AFL containing L, L_0 is in $\mathrm{LLCST}(\mathrm{LFT}^R)$. Since $\mathrm{LLCST}(\mathrm{LFT}^R)$ contains Λ and is closed under union, $L_0 \cup \Lambda$ is also in $\mathrm{LLCST}(\mathrm{LFT}^R)$. Therefore

$$\mathrm{CS} \supset \mathrm{LLCST}(\mathrm{LFT}^R) \supset \mathrm{LLCST}(L_0) \cup \mathrm{LLCST}(L_0 \cup \Lambda) = \mathrm{CS}.$$

Hence the theorem holds. Q.E.D.

Corollary 3.13. *If $\mathcal{L}$ is a semi-*AFL *closed under intersection and reversal, then $\mathcal{L} \supset \mathrm{CS}$ iff $\mathcal{L} \supset \mathrm{LLCS}(\mathrm{LFT})$.*

The complexity class $\mathrm{DSPACE}(f)$ with space bound f such that $f(n) \geqq n$ for all n is an AFL closed under intersection and reversal. Therefore by the corollary above

$$\mathrm{DSPACE}(f) \supset \mathrm{CS} \quad \text{iff} \quad \mathrm{DSPACE}(f) \supset \mathrm{LLCS}(\mathrm{LFT}).$$

Especially $\mathrm{DSPACE}(n)$ is equal to DCS, and hence the LBA-problem is equivalent to the problem whether $\mathrm{DCS} \supset \mathrm{LLCS}(\mathrm{LFT})$ or not.

Also the nondeterministic complexity class $\mathrm{NTIME}(g)$ with step bound g is an AFL closed under intersection and reversal. Therefore

$$\mathrm{NTIME}(g) \supset \mathrm{CS} \quad \text{iff} \quad \mathrm{NTIME}(g) \supset \mathrm{LLCS}(\mathrm{LFT}).$$

These equivalence of inclusion problems assures us that LLCS and LFT are important subfamilies of CS. Then naturally there arises a question whether LFT is much smaller than CS. In later sections we consider this problem.

§ 4. Traverse Automaton

To study LFT we introduce a subfamily SLFT of LFT, whose elements are called simple ltl's. Any ltl is a length-preserving

homomorphic image of some simple ltl. Next we define a new type of automaton called traverse automaton. A traverse automaton can be considered as both a recognizer and a transducer. As a recognizer it accepts a simple ltl, and as a transducer it results an ltl.

Let # be a special symbol only used as an endmarker. In the sequel we assume that any "alphabet" does not contain #.

Definition 4. 1. A language R over Σ is called a *local language* iff there exists $C \subset (\Sigma \cup \{\#\})^2$ such that

$$\#R\# = \#\Sigma^*\# - (\Sigma \cup \{\#\})^*((\Sigma \cup \{\#\})^2 - C)(\Sigma \cup \{\#\})^*,$$

that is, $R \ni w$ iff $\#w\# = a_1 a_2 \cdots a_n$ and for all i $a_i a_{i+1} \in C$. We denote C by C_R. Let LOC denote the family of local languages. It is well-known that $\mathrm{REG} = H_p(\mathrm{LOC})$.

Let SLFT denote the family LOC$\wedge$LLCS. Elements of SLFT are called *simple ltl*'s.

Corollary 4. 2. SLFT *is closed under intersection and inverse homomorphism.*

Proof. For both LOC and LLCS are closed under these operations.

Corollary 4. 3. $\mathrm{LFT} = H_p(\mathrm{SLFT}) = \mathrm{LLCST}(\mathrm{LOC})$.

Proof. Since LOC is closed under renaming,

$$\mathrm{LFT} \supset \mathrm{LLCST}(\mathrm{LOC}) = H_p(\mathrm{LOCL} \wedge \mathrm{LCS}) = H_p(\mathrm{SLFT}).$$

Since $\mathrm{REG} = H_p(\mathrm{LOC})$,

$$\begin{aligned}\mathrm{LFT} &= H_p(\mathrm{REG} \wedge \mathrm{LLCS}) = H_p(H_p(\mathrm{LOC}) \wedge \mathrm{LLCS}) \\ &\subset H_p(H_p(\mathrm{LOC} \wedge H_p^{-1}(\mathrm{LLCS}))) \\ &= H_p(\mathrm{LOC} \wedge \mathrm{LLCS}) = H_p(\mathrm{SLFT}),\end{aligned}$$

and the corollary holds. Q.E.D.

Now we define traverse automaton. We use the term "automaton"

because its states and next state function are given by a local language and its storage operations shall be given by its traverse relation defined later.

Definition 4.4. A *traverse automaton* (abbreviation: *tra*) is a pair $M=(T, R)$ where

(i) $T=(V, \Sigma, Q, P)$ is an llcst with $P \subset V \times (V-\Sigma) \cup V^2 \times V(V-\Sigma)$, that is, productions $A \to B$ and $AC \to AD$ of T satisfy $B, D \in V-\Sigma$,

(ii) R is a local language over Σ.

The language accepted by M, in symbols $L(M)$, is defined by

$$L(M)=D(T) \cap R,$$

and the language associated with M, in symbols $A(M)$, is defined by

$$A(M)=T(R).$$

Trivially $L(M)$ is a simple ltl and $A(M)$ is an ltl.

Corollary 4.5. *A language is a simple ltl iff it is accepted by a tra. A language is an ltl iff it is associated with a tra.*

Definition 4.6. Let $M=(T, R)$ be a tra with $T=(V, \Sigma, Q, P)$. The set of *admissible words* of M, in symbols $Adm(M)$, is defined by

$$Adm(M)=\Sigma(V-\Sigma)^* \cap V^*Q.$$

The *traverse relation* $\prec_M$ on $Adm(M)$ is defined as follows. Let $x=a_1a_2\cdots a_n$ and $y=b_1b_2\cdots b_m$ be in $Adm(M)$. We define $x \prec_M y$ iff $a_1b_1 \in C_R$ and there exists a partial function

$$p: \{1, 2, \cdots, m-1\} \to \{1, 2, \cdots, n\}$$

such that

(i) if $i<j$ and both $p(i)$ and $p(j)$ are defined, then $p(i) \leqq p(j)$,

(ii) if $p(i)$ is defined, then $a_{p(i)}b_i \to a_{p(i)}b_{i+1} \in P$,

(iii) if $p(i)$ is undefined, then $b_i \to b_{i+1} \in P$.

If $m=1$, then p is interpreted as $\emptyset$ and $x \prec_M y$ iff $a_1b_1 \in C_R$. To emphasize p we shall write $x \prec_M y$ via p. Let $\prec_M^*$ denote the reflexive and

transitive closure of $\prec_M$.

We say an admissible word $x=a_1a_2\cdots a_n$ is *initial* and write $\#<\!\cdot x$ iff $\#a_1\in C_R$ and for all i, $a_i\to a_{i+1}\in P$. We write $\#<\!\cdot^* x$ iff $\#<\!\cdot x$ or for some y $\#<\!\cdot y$ and $y\prec^* x$.

We say an admissible word $y=b_1b_2\cdots b_m$ is *final* and write $y\cdot\!>\#$ iff $b_1\#\in C_R$. We write $y\cdot\!>^*\#$ iff $y\cdot\!>\#$ or for some z $y\prec^* z$ and $z\cdot\!>\#$.

The *traverse language* of M, in symbols $Tra(M)$, is defined by

$$Tra(M)=\{x\in Adm(M)\,|\,\#<\!\cdot^* x\cdot\!>^*\#\}.$$

A finite sequence $(x_1, \cdots, x_n)$ of admissible words is called a *traverse* of M iff $x_i\prec x_{i+1}$ for all $i<n$. Furthermore it is called *complete* iff $\#<\!\cdot x_1$ and $x_n\cdot\!>\#$.

Let *top* and *bottom* be functions for words defined by

$$top(\lambda)=bottom(\lambda)=\lambda,$$
$$top(aw)=a, \quad \text{and} \quad bottom(wb)=b,$$

where a and b are symbols. Then top and bottom are extended for finite sequences of words as follows.

$$top(x_1, \cdots, x_n)=top(x_1)\cdots top(x_n),$$
$$bottom(x_1, \cdots, x_n)=bottom(x_1)\cdots bottom(x_n).$$

Corollary 4. 7. *Let M be a tra and $w\neq\lambda$. $L(M)\ni w$ iff $w=top(x_1, \cdots, x_n)$ for some complete traverse $(x_1, \cdots, x_n)$ of M. $A(M)\ni w$ iff $w=bottom(x_1, \cdots, x_n)$ for some complete traverse $(x_1, \cdots, x_n)$ of M.*

Example 4. 8. Let T be the llcst defined in Example 2. 6. Let $M=(T, a^*b^*)$. Then

$$L(M)=\{a^nb^m\,|\,n\geqq m\geqq 0\},$$
$$Tra(M)=a(AA')^*\cup a(AA')^*A\cup b(BB')^+\cup b(BB')^*B.$$

For instance the complete traverse corresponding to the flow diagram for $a^5b^5\in L(M)$ in Figure 2 is:

$$a\prec aA\prec aAA'\prec aAA'A\prec aAA'AA'\prec bBB'BB'B\prec bBB'BB'$$
$$\prec bBB'B\prec bBB'\prec bB.$$

§ 5. Partial Order by Embedding

In this section we introduce a partial order $\lhd$ for words defined by embedding, and next we prove elementary lemmas which play an important role in later sections.

Definition 5. 1. Let x and y be words. We say x *is embedded in* y and write $x \lhd y$ (or $y \rhd x$) iff $x = x_1 \cdots x_r$ and $y = y_0 x_1 y_1 \cdots x_r y_r$ for some words x_i's and y_j's. For each language L, define

$$Emb(L) = \{x \in L \mid \text{for any } y \text{ in } L,\ y \lhd x \text{ implies } y = x\}.$$

Since $\lhd$ is a partial order, $Emb(L)$ is the set of elements of L that are minimal in L with respect to the partial order $\lhd$. Haines [3] proved the following proposition.

Proposition 5. 2 (Haines). *For any language L, $Emb(L)$ is finite.*

Lemma 5. 3. *Let M be a tra and $x, y, z \in Adm(M)$.*

(i) *If $x \prec z$ and $x \lhd y$, then $y \prec z$.*

(ii) *If $x > \#$ and $x \lhd y$, then $y > \#$.*

Proof. (ii) is obvious, and we prove (i) only. Let $x = a_1 a_2 \cdots a_n$, $y = b_1 b_2 \cdots b_m$, and $z = c_1 c_2 \cdots c_s$. Since $x \lhd y$, there exists an injective function f: $\{1, \cdots, n\} \to \{1, \cdots, m\}$ such that $f(1) < \cdots < f(n)$ and $a_i = b_{f(i)}$ for all i. We note $f(1) = 1$ and $a_1 = b_1$, because $a_1, b_1 \in \Sigma$ and $a_i, b_j \in V - \Sigma$ for all $i, j \geqq 2$.

Assume $x \prec z$ via p. If $s = 1$, then $p = \emptyset$ and $a_1 z = b_1 z \in C_R$ and also $y \prec z$ via $\emptyset$. Now assume $s > 1$. Then $a_1 c_1 = b_1 c_1 \in C_R$ and $y \prec z$ via pf. Thus (i) holds. Q.E.D.

Corollary 5. 4. *If $(x_1, \cdots, x_n)$ and $(y_0, y_1, \cdots, y_m)$ are traverses and $x_n \rhd y_0$, then so is $(x_1, \cdots, x_n, y_1, \cdots, y_m)$.*

Corollary 5. 5. *If $\# < x_1 \prec \cdots \prec x_n$, $y_0 \prec y_1 \prec \cdots \prec y_m > \#$, and*

$x_n \triangleright y_0$, *then* $(x_1, \cdots, x_n, y_1, \cdots, y_m)$ *is a complete traverse.*

Next we introduce a partial order $\ll^*$ which is a subset of $\lhd$.

Definition 5. 6. Let x and y be words over Σ. We define $x \ll y$ (or $y \gg x$) iff $x = x_1 a x_2$ and $y = x_1 a y_1 a x_2$ for some words x_1, y_1, x_2 and $a \in \Sigma$. Let $\ll^*$ ($\gg^*$) be the reflexive and transitive closure of $\ll$ ($\gg$). Obviously $x \ll^* y$ implies $x \lhd y$.

Lemma 5. 7. *Let M be a tra and $x, y, z \in Adm(M)$.*

(i) *If $x \prec z$ and $z \gg^* y$, then $x \prec y$.*

(ii) *If $\# <: z$ and $z \gg^* y$, then $\# <: y$.*

Proof. We prove (i) only. The proof for (ii) is easier. To prove (i), it suffices to show that if $x \prec z$ and $z \gg y$, then $x \prec y$. Assume $x \prec z$ via p, $z = a_1 a_2 \cdots a_n$, $i < j$, $a_i = a_j$, and $y = a_1 \cdots a_{i-1} a_j \cdots a_n$. Define a partial function $p' : \{1, 2, \cdots, n-j+i-1\} \to \{1, \cdots, |x|\}$ by

$$p'(t) = \begin{cases} p(t) & \text{if } t < i, \\ p(t+j-i) & \text{if } i \leqq t < n-j+i-1. \end{cases}$$

Then it is clearly seen that $x \prec y$ via p'. Q.E.D.

Lemma 5. 8. *Let $M = (T, R)$ be a tra with $T = (V, \Sigma, Q, P)$. Let k_M be the number of elements of V. Let $u, v \in Adm(M)$.*

(i) *If $u \prec v$, then there exists $z \in Adm(M)$ such that*

(1) *$u \gg^* z$, $|z| \leqq k_M |v|$, and $z \prec v$,*

(2) *for any x with $x \prec u$, $x \prec z$,*

(3) *if $\# <: u$, then $\# <: z$.*

(ii) *If $u :> \#$, then there exists $z \in Adm(M)$ such that*

(1) *$u \gg^* z$, $|z| \leqq k_M$, and $z :> \#$,*

(2) *for any x with $x \prec u$, $x \prec z$,*

(3) *if $\# <: u$, then $\# <: z$.*

Proof. We only prove (i). The proof for (ii) is easier. Define

$$X = \{y \mid u \gg^* y \text{ and } y \prec v\}.$$

Let z be a minimal element of X with respect to $\lhd$. Then $u \gg^* z$, $z \rightarrow v$, and both (2) and (3) hold by Lemma 5.7. Hence to complete the proof it suffices to show $|z| \leqq k_M |v|$. Let $z = a_1 a_2 \cdots a_n$, $v = b_1 b_2 \cdots b_m$, and $z \rightarrow v$ via p.

First assume $p = \emptyset$ and $|z| > k_M$. There exist $1 \leqq i < j \leqq n$ such that $a_i = a_j$. Set $z' = a_1 \cdots a_{i-1} a_j \cdots a_n$. Since $z \rightarrow v$ *via* $\emptyset$, $a_1 b_1 \in C_R$ and thus $z' \rightarrow v$ via $\emptyset$. Since $z \gg z'$, $u \gg^* z'$ and $z' \rightarrow v$. Therefore $z' \in X$ which contradicts to the choice of z. Hence $p = \emptyset$ implies $|z| \leqq k_M$.

Next assume $p \neq \emptyset$ and $|z| > k_M |v|$. Let

$$\{t_1 < t_2 < \cdots < t_s\} = \{p(i) \mid p(i) \text{ is defined}\}.$$

Since p is a partial function from $\{1, \cdots, m-1\}$ into $\{1, \cdots, n\}$, we have $1 \leqq s \leqq m-1$. Let $t_0 = 0$ and $t_{s+1} = n$. Since

$$|z| > k_M m \geqq k_M (s+1),$$

for some q $t_{q+1} - t_q > k_M$, and hence there exist $t_q < i < j \leqq t_{q+1}$ such that $a_i = a_j$. Set $z' = a_1 \cdots a_{i-1} a_j \cdots a_n$. Define a partial function $p' : \{1, \cdots, m-1\} \rightarrow \{1, \cdots, n-j+i\}$ by

$$p'(t) = \begin{cases} p(t) & \text{if } p(t) \leqq t_q, \\ p(t) - j + i & \text{if } p(t) \geqq t_{q+1}, \\ \text{undefined} & \text{if } p(t) \text{ is undefined}. \end{cases}$$

Clearly $z' \rightarrow v$ via p' and $u \gg^* z \gg z'$. This is a contradiction. Therefore $p \neq \emptyset$ implies $|z| \leqq k_M |v|$. Q.E.D.

Definition 5.9. The partial order $\lhd$ is extended for finite sequences of words. We define $(x_1, \cdots, x_n) \lhd (y_1, \cdots, y_m)$ iff $m \geqq n$ and there exist $1 \leqq k_1 < k_2 < \cdots < k_n \leqq m$ such that $x_i \lhd y_{k_i}$ for all i.

Let M be a tra and $z \in Adm(M)$. A traverse $(x_1, \cdots, x_n)$ is called *thrifty for* z iff $x_n \rightarrow z$ and $(x_1, \cdots, x_n)$ is minimal with respect to $\lhd$ in the set of traverses $(y_1, \cdots, y_m)$ with $y_1 \lhd x_1$ and $y_m \rightarrow z$. A traverse $(x_1, \cdots, x_n)$ is called *thrifty for* # iff $x_n \rhd \#$ and $(x_1, \cdots, x_n)$ is minimal with respect to $\lhd$ in the set of traverses $(y_1, \cdots, y_m)$ with $y_1 \lhd x_1$ and $y_m \rhd \#$.

Lemma 5.10. *Let* $\rightarrow^+$ *be the transitive closure of* $\rightarrow$. *If*

$x \prec^+ y$ ($x \succ^* \#$ resp.), then there exists a thrifty traverse $(z_1, \cdots, z_n)$ for y ($\#$ resp.) such that $x \rhd z_1$.

Proof. A minimal element with respect to $\lhd$ in the set of traverses $(y_1, \cdots, y_m)$ with $y_1 \lhd x$ and $y_m \prec y$ ($y_m \succ \#$ resp.) is a desired one. Q.E.D.

Lemma 5.11. *Let M be a tra and $z \in Adm(M)$.*

(i) *If $(x_1, \cdots, x_n)$ is a thrifty traverse for z, then*
  (1) *$x_i \lhd x_j$ implies $i \leqq j$,*
  (2) *for each $i < n$, $|x_i| \leqq k_M |x_{i+1}|$,*
  (3) *$|x_n| \leqq k_M |z|$.*

(ii) *If $(x_1, \cdots, x_n)$ is a thrifty traverse for $\#$, then*
  (1) *$x_i \lhd x_j$ implies $i \leqq j$,*
  (2) *for each $i < n$, $|x_i| \leqq k_M |x_{i+1}|$,*
  (3) *$|x_n| \leqq k_M$.*

Proof. We only prove (i). First assume $x_i \lhd x_j$ and $i > j$. By Lemma 5.3 $(x_1, \cdots, x_j, x_{i+1}, \cdots, x_n)$ is also a traverse and $x_n \prec z$ ($x_j \prec z$ if $i = n$). This is a contradiction. Thus (1) holds. (2) and (3) are easily established by Lemma 5.8. Q.E.D.

§ 6. Emptiness Problem

This section proves that the emptiness problem for left transformation language is effectively solvable. We note that given x_1 and x_2 it is effectively determined whether $x_1 \prec x_2$ or not. Also given z and $(x_1, \cdots, x_n)$ it is effectively determined whether $(x_1, \cdots, x_n)$ is a thrifty traverse for z or not.

Lemma 6.1. *Let M be a tra and $z \in Adm(M) \cup \{\#\}$. Define*

$$Thr(z) = \{(x_1, \cdots, x_n) \mid (x_1, \cdots, x_n) \text{ is a thrifty traverse for } z\}.$$

$Thr(z)$ is finite and it is effectively determined.

Proof. Assume $z \in Adm(M)$. The proof for the case of $z = \#$ is

similar. For each $n \geqq 1$, define N_n as follows.

$$N_1 = \{(x_1) \mid x_1 \prec z \text{ and } |x_1| \leqq k_M|z|\},$$

$$N_{n+1} = \{(x_{n+1}, x_n, \cdots, x_1) \mid (x_n, \cdots, x_1) \in N_n,\ x_{n+1} \prec x_n,$$

$$|x_{n+1}| \leqq k_M|x_n|, \text{ and } x_{n+1} \not\rhd x_i \text{ for any } i \leqq n\}.$$

N_n is finite and we note that N_n is effectively determined. By Lemma 5. 11,

$$Thr(z) \subset \bigcup_{n=1}^{\infty} N_n .$$

Next we show $N_n = \emptyset$ for some n. Assume $N_n \neq \emptyset$ for any n. Then there exists an infinite sequence $x_1, x_2, \cdots, x_n, \cdots$ such that

(1) $\cdots \prec x_n \prec \cdots \prec x_2 \prec x_1 \prec z$,

(2) for all i, $|x_{i+1}| \leqq k_M|x_i|$, and $|x_1| \leqq k_M|z|$,

(3) for any $i < j$, $x_i \not\lhd x_j$.

But (3) contradicts to Proposition 5. 2. Therefore

$$n_0 = \min\{n \mid N_n = \emptyset\}$$

is effectively determined. We note that $N_n = \emptyset$ implies $N_{n+1} = \emptyset$. Thus $Thr(z)$ is a subset of $\bigcup_{n<n_0} N_n$. Since it is effectively determined whether a traverse $(x_n, \cdots, x_1)$ is thrifty for z or not, $Thr(z)$ is effectively determined. Q.E.D.

Lemma 6. 2. *Let $M = (T, R)$ be a tra and $z \in Adm(M)$. The following conditions are equivalent.*

(1) $\# <:^* z$.

(2) *Either $\# <: z$, or there exists a thrifty traverse $(x_1, \cdots, x_n)$ for z such that $x_1 = a_1 a_2 \cdots a_m$, $\# a_1 \in C_R$, and $a_i \Rightarrow^+ a_{i+1}$ for all $i < m$.*

Proof. Assume $\# <:^* z$ and $\# \not<: z$. There exists y such that $\# <: y$ and $y \prec^+ z$. By Lemma 5. 10 there exists a thrifty traverse $(x_1, \cdots, x_n)$ for z with $x_1 \lhd y$. Let $x_1 = a_1 a_2 \cdots a_m$. Since $x_1 \lhd y$ and $\# <: y$, $\# a_1 \in C_R$ and $a_i \Rightarrow^+ a_{i+1}$ for all $i < m$. Therefore (1) implies (2). The converse implication is clear. Q.E.D.

Lemma 6. 3. *Given $z \in Adm(M)$, it is effectively determined whether $\# <\!\cdot^* z$ or not.*

Proof. By Lemma 6. 1 $Thr(z)$ is effectively determined. Thus it can be determined whether the condition (2) of Lemma 6. 2 holds or not. Q.E.D.

Lemma 6. 4. *Let M be a tra. The following conditions are equivalent.*

(1) *$A(M)$ is not empty.*

(2) *$L(M)$ is not empty.*

(3) *Either $C_R \ni \#\#$, or there exists $z \in Adm(M)$ such that $\# <\!\cdot^* z$ and $z \cdot\!> \#$ and $|z| \leqq k_M$.*

(4) *Either $C_R \ni \#\#$, or there exists a thrifty traverse $(x_1, \cdots, x_n)$ for $\#$ such that $x_1 = a_1 a_2 \cdots a_m$, $\# a_1 \in C_R$, and $a_i \Rightarrow^+ a_{i+1}$ for all $i < m$.*

Proof. The equivalence of (1) and (2) is trivial. Note that $A(M)$ contains λ iff $C_R \ni \#\#$. Hence by Lemma 5. 8 (1) is equivalent to (3). The equivalence of (3) and (4) can be easily proved by the way similar to the proof of Lemma 6. 2. Q.E.D.

Now we prove the main theorem of this section.

Theorem 6. 5. *The emptiness problem for* LFT *is effectively solvable.*

Proof. Suppose $T(R) \in$ LFT, where T is an llcst and $R \in$ REG. If a system generating or accepting R such as a regular grammar, a finite automaton etc., is given, then one can easily construct a length-preserving homomorphism h and a local language R' such that $h(R') = R$. Hence by combining T with h an llcst T' satisfying

$$T'(R') = T(h(R')) = T(R)$$

is obtained. Let $M = (T', R')$. Then M is a tra with $A(M) = T(R)$.

By Lemma 6.3 and (3) of Lemma 6.4, it is effectively determined whether $A(M)$ is empty or not. Q.E.D.

§ 7. Quasi-Prefix Property

Let P be a property possessed by languages. We say a family $\mathcal{L}$ of languages has the property P iff all elements of $\mathcal{L}$ have P.

A language L has the prefix property iff $L=Pref(L)$. By Lemma 2.4 LLCS has the prefix property. But LFT does not, because REG does not have the prefix property. In this section we introduce the quasi-prefix property, which is proved to be possessed by LFT. As a corollary we can see that LFT and CF(LIN) are incomparable.

First we note that LFT is closed under the operation *Pref*. In general we can prove that LFT is closed under right quotient by any languages.

Definition 7.1. The right quotient of L_1 by L_2, in symbols L_1/L_2, is defined by

$$L_1/L_2=\{u|\text{ for some } v\in L_2,\ uv\in L_1\}.$$

If $L\subset\Sigma^*$, then $Pref(L)=L/\Sigma^*$.

Notation 7.2. Let M be a tra. For a language L, $Rgh(M;L)$ denotes the set of traverses $(z, y_1, \cdots, y_m)$ of M such that $y_m>\#$ and $top(y_1, \cdots, y_m)\in L$. Then define

$$Min(M;L)=Emb(\{z|(z, y_1, \cdots, y_m)\in Rgh(M;L)\}).$$

By Proposition 5.2 $Min(M;L)$ is finite.

Lemma 7.3. *Let M be a tra and a language $L\not\ni\lambda$. If $w\neq\lambda$, then the following conditions are equivalent.*

(1) $L(M)/L\ni w$.

(2) *There exists a traverse $(x_1, \cdots, x_n)$ of M such that $\#<x_1$, $w=top(x_1, \cdots, x_n)$, and $x_n\rhd z$ for some z in $Min(M;L)$.*

Proof. First assume (2). Since $z\in Min(M;L)$, there exists

$(z, y_1, \cdots, y_m) \in Rgh(M; L)$. Then $(x_1, \cdots, x_n, y_1, \cdots, y_m)$ is a complete traverse of M because $x_n \triangleright z$. Since $top(y_1, \cdots, y_m)$ is in L, $L(M)/L$ contains w. Therefore (2) implies (1).

Next assume $L(M)/L \ni w$. There exists a complete traverse $(x_1, \cdots, x_n, y_1, \cdots, y_m)$ of M such that $w = top(x_1, \cdots, x_n)$ and L contains $top(y_1, \cdots, y_m)$. Then $(x_n, y_1, \cdots, y_m) \in Rgh(M; L)$ and there is some z in $Min(M; L)$ with $x_n \triangleright z$. Hence (1) implies (2). Q.E.D.

Theorem 7.4. LFT *is closed under right quotient by any languages.*

Proof. We sketch the proof. If h is a length-preserving homomorphism, then

$$h(L_1)/L_2 = h(L_1/h^{-1}(L_2)).$$

Since LFT $= H_p$(SLFT), to prove the theorem it suffices to show that $L(M)/L \in$ LFT for any tra M and language L. If L contains λ,

$$L(M)/L = L(M) \cup L(M)/(L - \Lambda).$$

Since LFT is closed under union, without loss of generality we can assume $L \not\ni \lambda$. Also LFT contains Λ, and hence it suffices to prove that $(L(M)/L) - \Lambda \in$ LFT.

Let $M = (T, R)$ and let \$ be a new symbol. $Pref(R)\$$ is also a local language. Note that if z is an element of $Min(M; L)$, then z has no subwords of the form aa where a is a symbol. For if $Rgh(M; L)$ contains a traverse $(z_1aaz_2, y_1, \cdots, y_m)$, then $(z_1az_2, y_1, \cdots, y_m)$ is also in $Rgh(M; L)$. Then, since $Min(M; L)$ is finite, one can construct a tra $M' = (T', Pref(R)\$)$ such that $(x_1, \cdots, x_n, y)$ is a complete traverse of M' for some y iff $(x_1, \cdots, x_n)$ is a traverse of M, $\# <_M x_1$, and $x_n \triangleright z$ for some z in $Min(M; L)$. Then by Lemma 7.3,

$$L(M') = ((L(M)/L) - \Lambda)\$ \in \text{LFT}.$$

Since LFT is an AFL, $(L(M)/L) - \Lambda$ is also in LFT. Q.E.D.

Corollary 7.5. LFT *is closed under the operation Pref.*

Definition 7.6. A language L has the quasi-prefix property iff there exists a nonnegative integer k such that $uv \in L$ implies $ux \in L$ for some x with $|x| \leqq k$. We call k the prefix constant of L. If $k=0$, then L has the prefix property.

Theorem 7.7. LFT *has the quasi-prefix property.*

Proof. Suppose $L \in$ LFT. There is a tra M with $L=A(M)$. Since $Thr(\#)$ is finite, define

$$k=\max\{n \mid (x_1, \cdots, x_n) \in Thr(\#)\}$$

We prove that k is the prefix constant of L. Assume $uv \in L$ with $|v|>k$. There exists a complete traverse $(x_1, \cdots, x_n)$ of M such that $bottom(x_1, \cdots, x_n)=uv$ and $bottom(x_1, \cdots, x_i)=u$. Then there is a thrifty traverse $(z_1, \cdots, z_m)$ for $\#$ such that $x_{i+1} \rhd z_1$. Then also $(x_1, \cdots, x_{i+1}, z_2, \cdots, z_m)$ is a complete traverse and

$$L \ni bottom(x_1, \cdots, x_{i+1}, z_2, \cdots, z_m)=uz\,,$$

where $z=bottom(x_{i+1}, z_2, \cdots, z_m)$ and $|z|=m \leqq k$. Q.E.D.

Example 7.8. Let $L=\{a^n b^m \mid 0 \leqq n \leqq m\}$. L is in LIN, but it is not in LFT. Because it does not have the quasi-prefix property.

Example 7.9. $L=\{a^m \mid m=2^n$ for some $n\}$ is not in LFT.

By the theorem above we can see that LFT is a very small subfamily of CS.

Corollary 7.10. LFT *and* CF(LIN) *are incomparable.*

Proof. The language L in Example 7.8 is in CF(LIN), but it is not in LFT. On the other hand the language L in Example 2.8 is in LFT, but it is not in CF. Q.E.D.

Corollary 7.11. LFT *is not closed under reversal nor complement.*

Proof. Let $L=\{a^nb^m|n\geqq m\geqq 0\}$. By applying semi-AFL operations to the language $D(T)$ in Example 2.6 L can be obtained, and thus $L\in$LFT. Since L^R does not have the quasi-prefix property, L^R is not in LFT, and LFT is not closed under reversal. Also by applying semi-AFL operations and complement to L, L^R can be obtained. Hence LFT is not closed under complement. Q.E.D.

References

[1] Penttonen, M., One-sided and two-sided context in formal grammars, *Information and Control*, **25** (1974), 371-392.

[2] Ginsburg, S. and Greibach, S. A., Principal AFL, *J. Comput. System Sci.*, **4** (1970), 308-338.

[3] Haines, L. H., On free monoids partially ordered by embedding, *J. Combinatorial Theory*, **6** (1969), 94-98.