INFORMATION AND CONTROL 27, 1–36 (1975)

# Generalizations of Regular Sets and Their Application to a Study of Context-Free Languages*

MASAKO TAKAHASHI

*Department of Information Science, Tokyo Institute of Technology, Oh-Okayama, Meguroku, Tokyo 152, Japan*

We extend the notion of regular sets of strings to those of trees and of forests in a unified mathematical approach, and investigate their properties. Then by taking certain one-dimensional expressions of these objects, we come to an interesting subclass of CF languages defined over paired alphabets. They are shown to form a Boolean algebra with the Dyck set as the universe, and to play an important role in the whole class of CF languages. In particular, using the subclass we prove a refinement of the well-known Chomsky–Schützenberger Theorem, and also prove that the decision procedure for parenthesis grammars can be extended to a broader class of CF grammars.

## INTRODUCTION

We are primarily concerned with the sets of trees and of forests which are related to the structural descriptions of generative grammars like CF (context-free) grammars. We view them as generalizations of regular sets (of strings) and study their properties in a unified mathematical framework. Then applying the results to a study of CF languages, we see some interesting facts about the languages.

The objects that we are studying are certain extensions of the tree sets accepted by tree automata. The reason why we take such extended objects into consideration is the following. On one hand, from a linguistic point of view one may want to express the two distinct types of conjunctions in natural languages (viz., the coordination and the subordination) separately in the structural descriptions. In the extended objects that we study, they are naturally distinguished even in their recursive occurrences (while this is not the case in the derivation tree sets of CF grammars or in the tree sets accepted

* The main part of the paper is based on the author's dissertation work which was carried out in University of Pennsylvania, Philadelphia, PA, partially supported by NSF Grants GS-35125, GS-2509, and GP-5661.

by tree automata). On the other hand, mathematically these are found to be interesting algebraic entities, sometimes making the argument easier and simpler than the case of restricted objects. Furthermore, when we apply the results to a study of CF languages, we find the extension really useful.

In Section 1, we review the regular sets of strings, focusing our attention on the subclass of "local sets of strings." Then by extending the view to sets of trees and of forests, we define "generalized regular sets" (Section 2), and study their mathematical properties (Sections 3 and 4). In Section 5, taking certain one-dimensional expressions of the generalized regular sets we come across an interesting subclass of CF languages called "nest sets," and see their distinctive properties. Using the nest sets as the central notion, we prove a refinement of Chomsky–Schützenberger Theorem, and also prove that there is a subclass of CF grammars which properly contains the parenthesis grammars and for which the equivalence problem is solvable. In Section 6, we discuss extensions of Büchi's canonical systems. They are systems to generate regular sets of forests out of their finite subsets by successive replacement of the subforests, and the ones to generate the nest sets by replacing particular segments of sentences. Finally in Section 7, we exhibit a subclass of pushdown automata that can serve as the "tracers" of the generalized regular sets, and discuss an implication of the tracers with respect to syntax-directed compilation of CF languages.

## 1. Regular Sets of Strings

First we review the class of regular sets of strings. The central notion in our approach to the class is the "local sets of strings" and "projections."

### 1.1. *S-Local Sets and Regular Sets*

The $S$-local set (or local set of strings) is a special type of regular set such that the membership of strings is completely determined by looking at all the pairs of consecutive symbols as well as the initial and the final symbols.

1.1.1. Definition. Let $K$ be an alphabet (i.e., a finite set of symbols). If a set $L \subseteq K^*$ satisfies

$$L - \{\epsilon\} = AK^* \cap K^*B - K^*CK^*$$

for some finite sets $A, B \subseteq K$ and $C \subseteq KK$, then $L$ is called an *S-local set* (over $K$). For example, over an alphabet $K$ with two different symbols $X$ and $Y$, the sets $X^*$, $(XY)^*$, $X^* \cup Y^*$, $\{X\}$, $\{XY\}$ are $S$-local sets, while $X^* \cup (XY)^*$, $X^*(XY)^*$, $\{XX\}$ are not.

1.1.2. Remark that the "standard regular events" by Chomsky *et al.* (1963) are precisely the $S$-local sets without null string, and also that $S$-local sets are "2-testable in the strict sense" according to McNaughton *et al.* (1971).

1.1.3. One may think of the finite sets $K$, $A$, $B$, $C$ in Definition 1.1.1 as specifying a finite graph, in which $K$ represents the set of all nodes and $A$, $B$, $KK - C$ are, respectively, the sets of initial nodes, of terminal nodes, and of the arrows. The $S$-local set $AK^* \cap K^*B - K^*CK^*$ then represents the set of all possible paths in the graph.

Another central notion in our approach to the regular sets is the relabeling operations or the length-preserving homomorphisms, which we call "projections."

1.1.4. DEFINITION. Suppose $A$ and $B$ are alphabets. Then function $h: A^* \to B^*$ is called a *projection*, providing

(1) $h(\epsilon) = \epsilon$,

(2) $h(a) \in B \quad (a \in A)$, and

(3) $h(as) = h(a)\,h(s) \quad (a \in A, s \in A^*)$.

1.1.5. Using the concepts of $S$-local sets and projections we can state a well-known characterization of the regular sets as follows:

THEOREM. (Medvedev, 1959; Chomsky *et al.*, 1963). *A set $S$ of strings is regular if and only if there exist an $S$-local set $S'$ and a projection $h$ such that $S = h(S')$.* ::

1.2. *Basic Properties of S-Local Sets and Relating Results*

1.2.1. THEOREM. *$S \subseteq K^*$ is an $S$-local set if and only if $xYz \in S$ and $x'Yz' \in S$ $(x, x', z, z' \in K^*, Y \in K)$ implies $xYz' \in S$.*

*Proof.* To see "if" clause, given $S \subseteq K^*$ with the property, let $B = \{X \in K \mid K^*X \cap S \neq \phi\}$ and $C = \{XY \mid X, Y \in K, K^*XYK^* \cap S = \phi\}$. Then by induction on the length of strings, we can verify $\{s \in XK^* \mid s's \in S$ for some $s' \in K^*\} = XK^* \cap K^*B - K^*CK^*$ holds for each $X \in K$. Hence by using set $A = \{X \in K \mid XK^* \cap S \neq \phi\}$, we can write

$$S - \{\epsilon\} = \bigcup_{X \in A} \{s \in XK^* \mid s's \in S \text{ for some } s' \in K^*\}$$
$$= \bigcup_{X \in A} (XK^* \cap K^*B - K^*CK^*) = AK^* \cap K^*B - K^*CK^*.$$

The converse is clear from the definition. ::

1.2.2. THEOREM. *The class of S-local sets is closed under intersection, * operation, and + operation* ($S^+ = S \cup SS \cup SSS \cup \cdots$). *(However it is not closed under union nor concatenation. See counterexamples in Section* 1.1.1.) ::

(In the paper we often state theorems without proofs when they are not hard to conceive. Unless otherwise mentioned, see Takahashi (1972b) for more details.)

1.2.3. THEOREM. *The class of S-local sets is closed under inverse projections. (That is, if S is an S-local set then so is the set* $h^{-1}(S) = \{s \mid h(s) \in S\}$ *for any projection h.)* ::

1.2.4. THEOREM. *If* $S_1, S_2, \ldots, S_m$ $(m > 0)$ *are S-local sets and* $\lg(S_1) = \lg(S_2) = \cdots = \lg(S_m)$, *then there is an S-local set S such that* $S_1, S_2, \ldots, S_m$ *are obtained as projective images of S. (The notation* $\lg(s)$ *for string s stands for the length of s. For set S of strings,* $\lg(S) = \{\lg(s) \mid s \in S\}$.) ::

As a corollary to this, we get the following property of the regular sets.

1.2.5. THEOREM. $S_1, S_2, \ldots, S_m$ $(m > 0)$ *are regular sets such that* $\lg(S_1) = \lg(S_2) = \cdots = \lg(S_m)$ *if and only if* $S_1, S_2, \ldots, S_m$ *are projective images of a regular set.* ::

1.2.6. The "length-preserving finite state transduction" by Elgot *et al.* (1965) is defined as a relation between two projective images of a regular set (that is, the relation $\{(h_1(s), h_2(s)) \mid s \in S\}$, where $S$ is a regular set and $h_1$ and $h_2$ are projections). In connection with the transductions, Theorem 1.2.5. can be phrased as follows: $S_1$ and $S_2$ are regular sets such that $\lg(S_1) = \lg(S_2)$ if and only if there is a length-preserving finite state transduction with $S_1$ as the domain and $S_2$ as the range. See Takahashi (1972a) for more on this subject and relating works.

## 2. GENERALIZED REGULAR SETS

Recall the definition of $S$-local sets in Section 1.1.1. Over an alphabet $K$, the set $AK^* \cap K^*B - K^*CK^*$ (where $A, B \subseteq K$ and $C \subseteq KK$) consists of strings $s$ such that $s = X_1X_2 \cdots X_n$ $(X_1, X_2, \ldots, X_n \in K, n > 0)$, $X_1 \in A$, $X_n \in B$ and $X_{i-1}X_i \in KK - C$ $(i = 2, 3, \ldots, n)$. When we interpret the string $X_1X_2 \cdots X_n$ as a derivation tree of a CF grammar which has a starting symbol $X_1$ and production rules $X_{i-1} \to X_i$ $(i = 2, 3, \ldots, n)$, we can observe

a similarity of the $S$-local set to the derivation tree set of a special type of CF grammar (in which the right-hand sides of rules are always single symbols). Removing the restriction we define a type of tree generating systems called "T-grammars." To make the discussion precise, we will start with preliminary definitions on trees.

2.1. *Ranked Alphabets and Trees*

2.1.1. Suppose that $A$ is a finite set associated with a relation $r_A \subseteq A \times \{0, 1, 2,...\}$. Then $A$ is called a *ranked alphabet*, and $r_A$ is the *rank* of $A$. For each $k \geqslant 0$, the subset $\{a \in A \mid (a, k) \in r_A\}$ is denoted by $A^{(k)}$.

Note that unlike the standard definition (e.g., Gorn, 1966; Thatcher *et al.*, 1968) the rank here is not necessarily a function. This is because we are primarily concerned with alphabets with infinite ranks and tree sets over such alphabets. In other words the tree sets we are interested in are those in which the number of direct descendants of each symbol is unbounded. In particular, when $r_A = A \times \{0, 1, 2,...\}$ we call it a *homogeneous rank*.

First we give a recursive definition of trees over a ranked alphabet, and then we will describe functional representations of trees using "tree domains."

2.1.2. For a ranked alphabet $A$, let $\mathscr{T}_A$ be the smallest set such that

(1) if $a \in A^{(0)}$, then $a\langle\ \rangle = a\langle\epsilon\rangle \in \mathscr{T}_A$,

(2) if $k > 0$, $a \in A^{(k)}$, and $t_1, t_2, ..., t_k \in \mathscr{T}_A$ then $a\langle t_1 t_2 \cdots t_k\rangle \in \mathscr{T}_A$.

The members of $\mathscr{T}_A$ are called the *trees* over $A$. For $a \in A^{(0)}$, we identify the tree $a\langle\ \rangle$ with the symbol $a$ itself. Therefore the set $\mathscr{T}_A$ includes $A^{(0)}$ as a subset.

2.1.3. We sometimes illustrate trees in two-dimensional form. For instance, Fig. 1 shows a tree $a\langle b\langle ac\rangle aa\rangle$ over an alphabet $A$ such that $(a, 0), (a, 3), (b, 2), (c, 0) \in r_A$.

2.1.4. In order to specify certain portions of trees, it is convenient to assign an "address" to each node of trees and think of a tree as a mapping from the addresses to an alphabet.

Suppose $J^*$ be the set of strings over positive integers $J = \{1, 2, 3,...\}$ including the null string denoted by 0 (i.e., $0q = q0 = q$ for each $q \in J^*$). We will assign to each tree $t \in \mathscr{T}_A$ a subset $D(t)$ of $J^*$ such that

(1) if $t = a \in A^{(0)}$, then $D(t) = \{0\}$,

(2) if $t = a\langle t_1 t_2 \cdots t_k\rangle (k > 0, a \in A^{(k)}$ and $t_1, t_2, ..., t_k \in \mathscr{T}_A)$, then $D(t) = \{0\} \cup \bigcup_{j=1}^{k} \{jq \mid q \in D(t_j)\}$.

For example, for the tree in Fig. 1 we have

$$D(a\langle b\langle ac\rangle aa\rangle) = \{0, 1, 11, 12, 2, 3\}.$$

The set $D(t)$ is called the *tree-domain* of tree $t$.

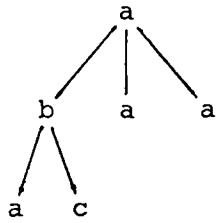


FIG. 1. Tree $a\langle b\langle ac\rangle aa\rangle$.

2.1.5. Corresponding to each $t \in \mathscr{T}_A$, there is a function $\underset{\sim}{t}$ (from the tree domain $D(t)$ to the alphabet $A$) such that

(1) if $t = a \in A^{(0)}$, then $\underset{\sim}{t}(0) = a$,

(2) if $t = a\langle t_1 t_2 \cdots t_k\rangle$ $(k > 0,\ a \in A^{(k)},$ and $t_1, t_2, \ldots, t_k \in \mathscr{T}_A)$, then $\underset{\sim}{t}(0) = a$, $\underset{\sim}{t}(jq) = \underset{=}{t_j}(q)$ for each $j \in \{1, 2, \ldots, k\}$ and $q \in D(t_j)$.

Without fear of ambiguity we drop the double underlines of the functional notation $\underset{\sim}{t}$ and use the same symbol $t$ to represent the function. Thus for instance if $t = a\langle b\langle ac\rangle aa\rangle \in \mathscr{T}_A$ (Fig. 1), we will write $t(0) = t(2) = t(3) = t(11) = a$, $t(1) = b$, and $t(12) = c$.

2.1.6. For tree $t \in \mathscr{T}_A$ we define $\mathrm{Root}(t) \in A$, $\mathrm{Branch}(t) \subseteq A\langle A^*\rangle$, $\mathrm{Leaf}(t) \subseteq A^{(0)}$, and $\mathrm{Yield}(t) \in (A^{(0)})^*$ as follows;

$$\mathrm{Root}(t) = t(0).$$

$$\mathrm{Branch}(t) = \begin{cases} \{a\} & \text{if } t = a\langle\,\rangle = a \in A^{(0)}, \\ \{a\langle \mathrm{Root}(t_1)\,\mathrm{Root}(t_2) \cdots \mathrm{Root}(t_k)\rangle\} \cup \bigcup_{j=1}^{k} \mathrm{Branch}(t_j) & \\ \qquad \text{if } t = a\langle t_1 t_2 \cdots t_k\rangle\ (k > 0,\ a \in A^{(k)}, \text{ and} & \\ \qquad t_1, t_2, \ldots, t_k \in \mathscr{T}_A), & \end{cases}$$

$$\mathrm{Leaf}(t) = \mathrm{Branch}(t) \cap A^{(0)},$$

$$\mathrm{Yield}(t) = \begin{cases} a & \text{if } t = a \in A^{(0)}, \\ \mathrm{Yield}(t_1)\,\mathrm{Yield}(t_2) \cdots \mathrm{Yield}(t_k) & \\ \qquad \text{if } t = \langle t_1 t_2 \cdots t_k\rangle\ (k > 0, a \in A^{(k)}, \text{ and} & \\ \qquad t_1, t_2, \ldots, t_k \in \mathscr{T}_A). & \end{cases}$$

For example, if $t = a\langle b\langle ac\rangle aa\rangle$ (see Fig. 1) then we have $\mathrm{Root}(t) = a$, $\mathrm{Branch}(t) = \{a\langle baa\rangle, b\langle ac\rangle, a, c\}$, $\mathrm{Leaf}(t) = \{a, c\}$, and $\mathrm{Yield}(t) = acaa$.

## 2.2. *T-Grammars, T-Local Sets, and T-Regular Sets*

2.2.1. DEFINITION. *Suppose $K$ is a ranked alphabet, $P \subseteq \mathrm{Branch}(\mathscr{T}_K)$, and $I \subseteq K$.* Then triple $G = (K, P, I)$ is called a *T-grammar*, provided that the set $P$ satisfies the *regularity condition*;

for each $X \in K$, the set $P(X) = \{x \in K^* \mid X\langle x\rangle \in P\}$ is a regular set $(\subseteq K^*)$.

The set of trees $\{t \in \mathscr{T}_A \mid \mathrm{Root}(t) \in I, \mathrm{Branch}(t) \subseteq P\}$ is called the *T-local set* generated by $G$, and denoted by $\mathscr{T}G$ or $\mathscr{T}(K, P, I)$. For instance, when $K$ is an alphabet of homogeneous rank, $P = \mathrm{Branch}(\mathscr{T}_K) = K\langle K^*\rangle$, and $I = K$, $(K, P, I)$ is a $T$-grammar and we have $\mathscr{T}(K, P, I) = \mathscr{T}_K$.

2.2.2. Suppose $A$ and $B$ are ranked alphabets and $h: A \to B$ is a function such that $h(A^{(k)}) \subseteq B^{(k)}$ for each $k \geqslant 0$. Let us extend the function $h$ to $h': \mathscr{T}_A \to \mathscr{T}_B$ as follows;

(1) if $t = a \in A^{(0)}$, then $h'(t) = h(t) \in B^{(0)}$,

(2) if $t = a\langle t_1 t_2 \cdots t_k\rangle$ $(k > 0$, $a \in A^{(k)}$, and $t_1, t_2, \ldots, t_k \in \mathscr{T}_A)$, then $h'(t) = h(a)\langle h'(t_1)\, h'(t_2) \cdots h'(t_k)\rangle \in \mathscr{T}_B$.

The tree function $h': \mathscr{T}_A \to \mathscr{T}_B$ so obtained is called a *projection* (of trees).

2.2.3. DEFINITION. If $T \subseteq \mathscr{T}_K$ is a $T$-local set and $h: \mathscr{T}_K \to \mathscr{T}_A$ is a projection, then the set $h(T) = \{h(t) \mid t \in T\}$ is called a *T-regular set*. That is, $T$-regular sets are the projective images of $T$-local sets.

2.2.4. Consider a $T$-grammar $G = (K, P, I)$ which has a finite set $P$ and a singleton set I. Such $T$-grammars are similar to the ordinary CF grammars when we see them as tree generating systems. The only difference between the two is that in the former the set of terminal symbols and that of non-terminals are not necessarily disjoint. (The "lexicon rules" of the latter are taken care of in our framework by the projections which transform $T$-local sets to $T$-regular sets.)

2.2.5. In literature, $T$-regular sets which are obtained by $T$-grammars of finite rules are often studied under the name of recognizable sets in connection with tree automata (Thatcher *et al.*, 1968; Brainerd, 1969; Thatcher, 1970).

Though we defined $T$-regular sets by means of grammars which may have

infinitely many rules, because of the regularity condition the sets can be obtained through some finite systems. We will discuss a type of such systems in Section 3.

2.3. *Forests*

While we have defined $T$-regular sets by generalizing the regular sets of strings, we now proceed the course of generalization one step further to define a class of "forest" sets. A forest (Gorn, 1966; Knuth, 1968) is nothing but a sequence of trees arranged sidewise, or equivalently is the structure obtained from a tree by removing the root node.

2.3.1. DEFINITION. For a ranked alphabet $A$, let $\mathscr{F}_A$ be the smallest set satisfying

(1) $\epsilon \in \mathscr{F}_A$,

(2) if $t \in \mathscr{T}_A$ and $w \in \mathscr{F}_A$ then $tw \in \mathscr{F}_A$.

The members of set $\mathscr{F}_A$ are called the *forests* over $A$. (Employing the conventional notation * to mean zero or more iterative juxtapositions, we can write $\mathscr{F}_A = (\mathscr{T}_A)^*$.) As in the case of trees, we illustrate forests by two-dimensional pictures like the one in Fig. 2 (where we assume that $(a, 1)$, $(b, 0)$, $(c, 3)$,..., $(i, 0) \in r_A$ for a ranked alphabet $A$).

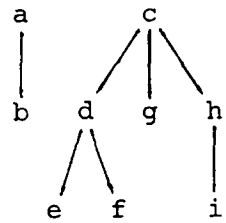


FIG. 2. Forest $a\langle b\rangle c\langle d\langle ef\rangle gh\langle i\rangle\rangle$.

In the paper, we will be only concerned with forests over homogeneously ranked alphabet. As for trees, our primary concern is also the ones over such alphabets. Hence in writing $\mathscr{T}_A$ or $\mathscr{F}_A$ unless otherwise stated we tacitly assume that $A$ is an alphabet with rank $r_A = A \times \{0, 1, 2,...\}$.

2.3.2. We will extend the definitions of Root, Branch, Leaf, and Yield of trees to those of forests:

(1) For the null forest $\epsilon$, we set

$\mathrm{Root}(\epsilon) = \mathrm{Yield}(\epsilon) = \epsilon$ (the null string),
$\mathrm{Branch}(\epsilon) = \mathrm{Leaf}(\epsilon) = \varnothing$ (the empty set).

(2) If $t \in \mathscr{T}_A$ and $w \in \mathscr{F}_A$ then we define

$\mathrm{Root}(tw) = \mathrm{Root}(t)\,\mathrm{Root}(w)$ $(\epsilon A^*)$,
$\mathrm{Branch}(tw) = \mathrm{Branch}(t) \cup \mathrm{Branch}(w)$ $(\subseteq A\langle A^*\rangle)$,
$\mathrm{Leaf}(tw) = \mathrm{Leaf}(t) \cup \mathrm{Leaf}(w)$ $(\subseteq A)$,
$\mathrm{Yield}(tw) = \mathrm{Yield}(t)\,\mathrm{Yield}(w)$ $(\epsilon A^*)$.

For example, if $w = a\langle b\rangle\, c\langle d\langle ef\rangle\, gh\langle i\rangle\rangle$ (see Fig. 2) then we have $\mathrm{Root}(w) = ac$, $\mathrm{Branch}(w) = \{a\langle b\rangle, c\langle dgh\rangle, d\langle ef\rangle, h\langle i\rangle, b, e, f, g, i\}$,

$\mathrm{Leaf}(w) = \{b, e, f, g, i\}$, and $\mathrm{Yield}(w) = befgi$.

2.3.3. The definition of tree-domain (Section 2.1.4) and the functional notation of trees (Section 2.1.5) are also extended to forests;

if $w \in \mathscr{F}_A$ and $t = a\langle w\rangle \in \mathscr{T}_A$ with $a \in A$, then $D(w) = D(t) - \{0\}$ and $w(q) = t(q)$ for each $q \in D(w)$.

(Note $\mathscr{F}_A = \{w \mid a\langle w\rangle \in \mathscr{T}_A,\ a \in A\}$.)

2.3.4. A function $h\colon \mathscr{F}_A \to \mathscr{F}_B$ is called a *projection*, if

(1) the restriction of $h$ to $\mathscr{T}_A$ is a projection of trees,

(2) $h(tw) = h(t)\,h(w)$ $(t \in \mathscr{T}_A,\ w \in \mathscr{F}_A)$,

(3) $h(\epsilon) = \epsilon$.

## 2.4. *F-Grammars, F-Local Sets, and F-Regular Sets*

2.4.1. DEFINITION. Suppose $K$ is an alphabet of homogeneous rank, $P$ is a subset of $K\langle K^*\rangle$ satisfying the regularity condition (Section 2.2.1), and $I$ is a regular set in $K^*$. Then the triple $G = (K, P, I)$ is called an *F-grammar* and the set

$$\mathscr{F}G = \mathscr{F}(K, P, I) = \{w \in \mathscr{F}_K \mid \mathrm{Root}(w) \in I,\ \mathrm{Branch}(w) \subseteq P\}$$

is an *F-local set*. The projective images of F-local sets are called *F-regular sets*.

Note that $\mathscr{F}_A$ is an F-local set generated by F-grammar $(A, A\langle A^*\rangle, A^*)$. Its subset $\mathscr{T}_A$, $A^*$, and $A\langle A\langle\cdots\langle A\rangle\cdots\rangle\rangle = \{a_1\langle a_2\langle\cdots\langle a_n\rangle\cdots\rangle\rangle \mid a_i \in A,\ 1 \leqslant i \leqslant n\}$ are also considered as F-local sets, respectively, generated by F-grammars $(A, A\langle A^*\rangle, A)$, $(A, A, A^*)$, and $(A, A\langle A\rangle \cup A, A)$.

2.4.2. *Remark* (*interrelation of T-regular sets and F-regular sets*). If $W \subseteq \mathscr{F}_A$ is $F$-regular, then set $B\langle W\rangle = \{b\langle w\rangle \mid b \in B, w \in W\}$ is a $T$-regular set for any $B \subseteq A$. Conversely, if $T \subseteq \mathscr{T}_A$ is $T$-regular, then set $\{w \mid a\langle w\rangle \in T, a \in A\}$ is $F$-regular.

2.4.3. The notion of $F$-regular set, which as we shall see later inherits more from the regular sets of strings than the notion of $T$-regular set does, sometimes makes mathematical argument simpler and easier. In fact, in the sequel of the paper we often study $F$-regular sets first and then apply the result to the case of trees. As for the major application of $F$-regular sets to the theory of CF languages, see Section 5. Another instance of application [to the string adjunct languages of Joshi *et al.* (1972)] will be found in Takahashi (1972b).

Pair *et al.* (1968) studied the $F$-regular sets in a different approach under the name of "bilangage régulier." Worth mentioning is the coincidence in spite of the independency of the works.

## 3. Finite Systems for Generalized Regular Sets

In the previous section we defined the regular sets of trees and of forests by means of grammars which possibly have infinitely many rules. Now we search for classes of finite systems that have comparable generating power to those infinite counterparts; that is, the classes of finite systems that can produce all and only $T$- or $F$-regular sets with the help of projection functions.

The essential difference between $T$- or $F$-grammars in Section 2 and the finite systems here is that in the former grammars trees or forests grow only in one direction, namely from top to bottom, while in the latter they can grow in two directions, from top to bottom and left to right. Such finite systems will give us another aspect of $T$- and $F$-regular sets, as well as provide useful tools for the study of these sets.

### 3.1. $\overset{A}{T}$-*Systems*

3.1.1. A $\overset{A}{T}$-system informally speaking is a finite system to generate trees out of three types of basic pieces—triangles, edges, and vertices (cf. Fig. 3). The inside of trees are inlaid with triangle pieces like mosaics, while two other types are used to specify the contour of trees; edge pieces are for the leftmost or the rightmost branching lines, and vertices are for the roots and leaves. For example, the set of trees

$$T = \{X\langle y_1 X\langle y_2 X\langle \cdots\langle y_n X\rangle \cdots\rangle\rangle\rangle \mid y_1, y_2, \ldots, y_n \in Y^*, n \geqslant 0\}$$

over alphabet $K = \{X, Y\}$ (cf. Fig. 4) is generated by a $\overset{\Delta}{T}$-system with triangle pieces $\{XYY, XYX\}$ for the inside of trees, edge pieces $\{XY, XX\}$ for the leftmost branching lines, $\{XX\}$ for the rightmost branching lines, a vertex $\{X\}$ for roots, and $\{X, Y\}$ for leaves.

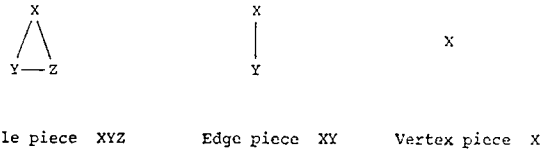


FIG. 3. Three types of basic pieces constituting $\overset{\Delta}{T}$-systems.

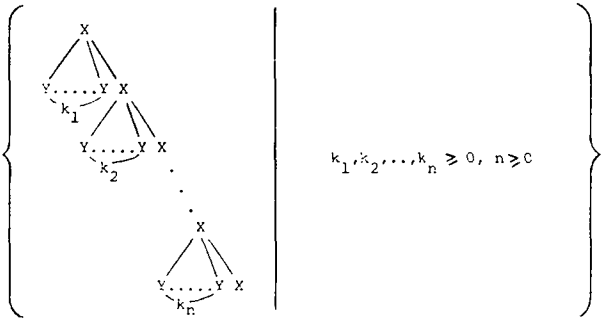


FIG. 4. Tree set $T = \{X\langle y_1 X\langle y_2 X\langle \cdots\langle y_n X\rangle\cdots\rangle\rangle\rangle \mid y_1, y_2, \ldots, y_n \in Y^*, n \geqslant 0\}$.

3.1.2. DEFINITION. A $\overset{\Delta}{T}$*-system* is a 6-tuple $G = (K, P_0, P_1, P_2, P_3, P_4)$ of finite sets such that $P_0 \subseteq K$ (for roots), $P_1 \subseteq KK$ (for the leftmost branching lines), $P_2 \subseteq KKK$ (for inside triangles), $P_3 \subseteq KK$ (for the rightmost branching lines), and $P_4 \subseteq K$ (for leaves). The $\overset{\Delta}{T}$*-local set* generated by $G$ is

$$\overset{\Delta}{\mathscr{T}}G = \left\{ t \in \mathscr{T}_K \;\middle|\; \begin{array}{l} \mathrm{Root}(t) \in P_0\,;\ \mathrm{Leaf}(t) \subseteq P_4\,; \\ \text{if } X\langle X_1 X_2 \cdots X_k\rangle \in \mathrm{Branch}(t)\ (k > 0,\ X, X_1, \ldots, X_k \in K) \\ \text{then } XX_1 \in P_1,\ XX_i X_{i+1} \in P_2\ (i = 1, \ldots, k-1),\ XX_k \in P_3 \end{array} \right\}.$$

3.1.3. THEOREM. *$T \subseteq \mathscr{T}_A$ is a $T$-regular set if and only if it is a projective image of a $\overset{\Delta}{T}$-local set.*

*Proof.* Notice that $T \subseteq \mathscr{T}_K$ is a $\overset{\Delta}{T}$-local set if and only if it is a $T$-local set such that for each $X \in K$ the set $\{x \in K^* \mid X\langle x\rangle \in \mathrm{Branch}(T)\}$ is $S$-local.

Hence we can derive the theorem as a direct consequence of the following lemma. ::

3.1.4 LEMMA. *For any T-local set $T$, there is a T-grammar $G' = (K', P', I')$ such that*

(1) $P'(Z) = \{z \mid Z\langle z\rangle \in P'\}$ *is an S-local set for each* $Z \in K'$,

(2) $T$ *is a projective image of* $\mathscr{T}G'$.

*Proof.* Suppose $G = (K, P, I)$ is a $T$-grammar such that $T = \mathscr{T}G$. For each $X \in K$, since the set $P(X) = \{x \in K^* \mid X\langle x\rangle \in P\}$ is regular, there exist an $S$-local set $S_X$ over an alphabet $K_X$ and a projection $h_X : (K_X)^* \to K^*$ such that $P(X) = h_X(S_X)$ (Section 1.1.5). Without loss of generality we may assume that $K$ and all $K_X$'s ($X \in K$) are mutually disjoint. Let $K'$ be the union $K \cup \bigcup_{X\in K} K_X$, and define a projection $g$: $(K')^* \to K^*$ by $g(X) = X$ ($X \in K$) and $g(Z) = h_X(Z)$ ($Z \in K_X$, $X \in K$). Now we define a $T$-grammar $G' = (K', P', I')$ with

$$P' = \{Z\langle z\rangle \mid Z \in g^{-1}(X), z \in S_X \text{ for some } X \in K\} \quad (\subseteq K'\langle(K')^*\rangle),$$

$$I' = I \quad (\subseteq K \subseteq K').$$

Then we can verify that $T = \mathscr{T}G = g(\mathscr{T}G')$ and that for each $Z \in K'$ the set $P'(Z)$ equals the $S$-local set $S_{g(Z)}$. ::

3.2. *$\tilde{F}$-Systems*

Forests are defined in Section 2 as the sequences of trees arranged sidewise. By taking another view of the same objects, we will develop a class of forest generating systems which have finite components and can produce precisely the class of $F$-regular sets with the aid of projections.

3.2.1. Informally speaking, we regard forests as the structures which are composed of nodes connected together by means of two types of connectives; one is called the "subordination" and the other is the "coordination." Each node in the structure has at most one successor of each type, and all the nodes in a structure are retrieved back to a node, called the "origin" of the structure.

For example, take a forest $w = a\langle b\rangle\, c\langle d\langle f\rangle\, gh\langle i\rangle\rangle$ over an alphabet $A = \{a, b, \ldots, i\}$ (cf. Figs. 2 and 5). In $w$, the pairs of nodes $(a, b)$, $(c, d)$, $(d, e)$, $(h, i)$ are in the subordinate relation and nodes $b, e, f, g, i$ are terminal nodes with respect to subordination, while the pairs $(a, c)$, $(d, g)$, $(g, h)$, $(e, f)$ are in the coordinate relation and $b, c, f, h, i$ are terminal nodes with respect

to coordination. The node $a$ is the origin of $w$. In order to relate the two aspects of forests, we introduce some additional notions.

3.2.2. A *two-dimensional concatenation* $\varphi_a(a \in A)$ is a binary operation in $\mathscr{F}_A$ such that

$$\varphi_a(w_1, w_2) = a\langle w_1\rangle w_2 \quad (w_1, w_2 \in \mathscr{F}_A).$$

(Recall that $a\langle w_1\rangle w_2$ represents the forest which has tree $a\langle w_1\rangle$ as the leftmost component and forest $w_2$ following the tree to the right.) For example, $\varphi_a(b\langle c\rangle, d\langle ef\rangle) = a\langle b\langle c\rangle\rangle d\langle ef\rangle$. Note that $\varphi_a(\epsilon, \epsilon) = a$, and that any forest can be obtained from $\epsilon$ via two-dimentional concatenations. It means the set $\mathscr{F}_A$ is equal to the smallest set containing $\epsilon$ and closed under $\varphi_a$'s $(a \in A)$.

3.2.3. For each $w \in \mathscr{F}_A$, we define $\mathrm{Origin}(w) \in A \cup \{\epsilon\}$ and $\mathrm{SO}(w)$, $\mathrm{CO}(w) \subseteq AA \cup A$ recursively as follows:

If $w = \varphi_a(w_1, w_2)$ $(w_1, w_2 \in \mathscr{F}_A, a \in A)$, then

$\mathrm{Origin}(w) = a$,
$\mathrm{SO}(w) = \mathrm{SO}(w_1) \cup \mathrm{SO}(w_2) \cup \{aa_1\}$,
$\mathrm{CO}(w) = \mathrm{CO}(w_1) \cup \mathrm{CO}(w_2) \cup \{aa_2\}$, where $a_i = \mathrm{Origin}(w_i)$ $(i = 1, 2)$;

if $w = \epsilon$, then

$\mathrm{Origin}(w) = \epsilon$
$\mathrm{SO}(w) = \mathrm{CO}(w) = \varnothing$ (the empty set).

For example, we have for the forest $w$ in Fig. 5 $\mathrm{Origin}(w) = a$, $\mathrm{SO}(w) = \{ab, cd, de, hi, b, e, f, g, i\}$ and $\mathrm{CO}(w) = \{ac, dg, gh, ef, b, c, f, h, i\}$.

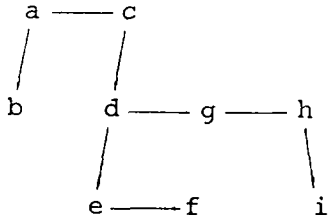


FIG. 5. Forest $a\langle b\rangle\, c\langle d\langle ef\rangle\, gh\langle i\rangle\rangle$.

We are now in a position to define a class of finite systems to generate forest sets.

3.2.4. DEFINITION. An *$\tilde{F}$-system* is a quadruple $G = (K, P_0, P_1, P_2)$ of finite sets such that $P_0 \subseteq K \cup \{\epsilon\}$ and $P_1, P_2 \subseteq KK \cup K$. The *$\tilde{F}$-local set* generated by $G$ is

$$\tilde{\mathscr{F}}G = \{w \in \mathscr{T}_K \mid \mathrm{Origin}(w) \in P_0, \mathrm{SO}(w) \subseteq P_1, \mathrm{CO}(w) \subseteq P_2\}.$$

3.2.5. EXAMPLE. Consider the $F$-local set $W$ generated by $F$-grammar $(K, P, I)$ such that $K = \{X, Y\}$, $P = X\langle Y^*\rangle \cup Y\langle X^*\rangle$, and $I = X^* \cup Y^*$. ($W$ is the set of forests over $K$ such that all the nodes of depth 0, 2, 4,... are labeled by a same symbol (either $X$ or $Y$), whereas the nodes of depth 1, 3, 5,... are by the other symbol ($Y$ or $X$, respectively).) The set $W$ can be generated by $\tilde{F}$-system $G = (K, P_0, P_1, P_2)$ with

$$P_0 = K \cup \{\epsilon\}, \quad P_1 = \{XY, YX, X, Y\}, \quad \text{and} \quad P_2 = \{XX, YY, X, Y\}.$$

3.2.6. *Remark.* Consider an $\tilde{F}$-system $G = (K, P_0, P_1, P_2)$ such that $P_1 = K$. Then, since no symbol pairs are allowed in the subordinate relation, the forests generated by $G$ are degenerated to mere strings, and the $\tilde{F}$-local set $\tilde{\mathscr{F}}G$ becomes an $S$-local set. Conversely any $S$-local set $S$ can be expressed as $\tilde{\mathscr{F}}(K, P_0, P_1, P_2)$ with $P_1 = K$ (where $P_0 = \mathrm{Origin}(S)$ and $P_2 = \mathrm{CO}(S)$). Next, take an $\tilde{\mathscr{F}}$-system $G' = (K, P_0, P_1, P_2)$ with $P_2 = K$. Then every node in the forests generated by $G'$ has at most one direct descendant, and the $\tilde{F}$-local set $\tilde{\mathscr{F}}G'$ (when the angle brackets in the tree notation are ignored) is identified with an $S$-local set. Thus the class of $S$-local sets are embedded in the class of $\tilde{F}$-local sets in two ways; as the sets of "horizontal strings" and as those of "vertical strings."

3.2.7. THEOREM. *$W \subseteq \mathscr{T}_A$ is an $F$-regular set if and only if it is a projective image of an $\tilde{F}$-local set.*

*Proof of if.* Any $\tilde{F}$-local set is $F$-local. Indeed, given $\tilde{F}$-system $G = (K, P_0, P_1, P_2)$, let $I = \tilde{\mathscr{F}}(K, P_0, K, P_2)(\subseteq K^*)$ and $P = \{X\langle x\rangle \mid x \in K, x \in \tilde{\mathscr{F}}(K, \{Y \in K \cup \{\epsilon\} \mid XY \in P_1\}, K, P_2)\}$ $(\subseteq K\langle K^*\rangle)$. Then $(K, P, I)$ is an $F$-grammar satisfying $\mathscr{F}(K, P, I) = \tilde{\mathscr{F}}G$. ::

*Proof of only if.* First, remark that $W \subseteq \mathscr{T}_A$ is an $F$-regular set if and only if $A\langle W\rangle = \{a\langle w\rangle \mid a \in A, w \in W\} \subseteq \mathscr{T}_A$ is a $T$-regular set (Section 2.4.2), and that for any $T$-regular set $T$ there is a $T$-grammar $G = (K, P, I)$ such that (1) the set $P_X = \{x \in K^* \mid X\langle x\rangle \in P\}$ for each $X \in K$ is an $S$-local set over $K_X$ $(\subseteq K)$ and the subalphabets $K_X$'s $(X \in K)$ are mutually disjoint, and (2) $T$ is a projective image of $\mathscr{T}G$. (To substantiate the existence of such a $T$-grammar, it suffices to recollect the proof of Lemma 3.1.4 by

reexamining the property of $T$-grammar $G' = (K', P', I')$ constructed there.) Given $F$-regular set $W \subseteq \mathscr{F}_A$ when the tree set $T = A\langle W \rangle$ is expressed as a projective image of set $\mathscr{T}G \subseteq \mathscr{T}_K$, where $G$ is a $T$-grammar with the property (1) above, the set $W' = \{w \in \mathscr{F}_K \mid X\langle w \rangle \in \mathscr{T}G \text{ for some } X \in K\}$ is shown to be an $\tilde{F}$-local set. (i.e., $W' = \tilde{\mathscr{F}}(K, P_0, P_1, P_2)$ holds for $P_0 = \text{Origin}(W')$, $P_1 = \text{SO}(W')$, and $P_2 = \text{CO}(W')$.) It is obvious that the $F$-regular set $W$ is a projective image of set $W'$. ::

## 4. Generalizations of Myhill–Nerode Theorem

As is well-known, the Boolean closure property of the regular sets of strings is shown by constructing appropriate finite state automata. The same reasoning can be stated in terms of the congruence relations appeared in Myhill–Nerode characterization of the sets. Pursuing the line, we first discuss characterization of the generalized regular sets in terms of certain congruence relations, and then prove that the classes of $T$- and $F$-regular sets are individually closed under Boolean operations.

### 4.1. *Finite Congruences and Characterizable Sets*

We start with a general discussion on finite congruences (viz., congruences of finite indices). Throughout this subsection, we assume that $\mathscr{R}$ is a set representing a universe of discourse like $A^*$, $\mathscr{T}_A$ etc., and $\Lambda$ is a set of operations defined in $\mathscr{R}$. That is, according to the terminology of universal algebra, we take a $\Lambda$-algebra $\mathscr{R}$.

4.1.1. Suppose $\mathbb{E}$ is a partition of $\mathscr{R}$ with the following property; for any $k$-ary operation $\lambda \in \Lambda$ and $k$ blocks $E_1, E_2, \ldots, E_k \in \mathbb{E}$ $(k \geqslant 0)$ there is a block $E \in \mathbb{E}$ such that $\lambda(E_1, E_2, \ldots, E_k) \subseteq E$.

Then we say $\mathbb{E}$ is a *congruence* of $\Lambda$-algebra $\mathscr{R}$ (or $\mathbb{E}$ is a *$\Lambda$-congruence* of set $\mathscr{R}$).

4.1.2. Let $\mathbb{E}$ be a finite congruence of $\Lambda$-algebra $\mathscr{R}$. If a set $R \subseteq \mathscr{R}$ is equal to a union of blocks of $\mathbb{E}$ (i.e., $R = \cup \{E \in \mathbb{E} \mid E \subseteq R\}$), then we say that $\mathbb{E}$ *characterizes* $R$, and $R$ is *$\Lambda$-characterizable.*

As an example let us consider the case of string sets. Let $A$ be an alphabet and $\sigma_a$ $(a \in A)$ represent the concatenation operation of symbol $a$ to the strings over $A$. That is, $\sigma_a(x) = ax$ $(x \in A^*)$. Then concerning $\sigma_A$-algebra $A^*$, where $\sigma_A = \{\sigma_a \mid a \in A\}$, we can state the well-known theorem by Myhill (1959) and Nerode (1958) as follows.

4.1.3. THEOREM. *A set $S \subseteq A^*$ is regular if and only if there is a finite congruence $\mathbb{E}$ of $\sigma_A$-algebra $A^*$ such that $S = \cup \{E \in \mathbb{E} \mid E \subseteq S\}$ (i.e., $S$ is $\sigma_A$-characterizable).* ::

As another example, consider the class of $T$-regular sets in $\mathscr{T}_A$, where $A$ is a finitely ranked alphabet. Let $\tau_{a,k}$ $((a, k) \in r_A)$ stand for the $k$-ary operation in $\mathscr{T}_A$ such that

$$\tau_{a,k}(t_1, t_2, \ldots, t_k) = a\langle t_1 t_2 \cdots t_k \rangle \qquad (t_i \in \mathscr{T}_A, i = 1, 2, \ldots, k),$$

and let $\tau_A = \{\tau_{a,k} \mid (a, k) \in r_A\}$. Then by using the minimization technique of tree automata (Brainerd, 1968), we can prove the following.

4.1.4. THEOREM. *For an alphabet $A$ of finite rank, a set $T \subseteq \mathscr{T}_A$ is $T$-regular if and only if $T$ is $\tau_A$-characterizable.* ::

The premise that the rank of $A$ be finite is indispensable in the theorem. In fact, though $\mathscr{T}_A$ is always $\tau_A$-characterizable (by congruence $\mathbb{E} = \{\mathscr{T}_A\}$), it may not be $T$-regular (depending on the rank of $A$). In Section 4.2, we will discuss another algebra of trees over a homogeneously ranked alphabet, in which characterizable sets and $T$-regular sets are coextensive.

We now remark two fundamental properties of $\Lambda$-characterizable subsets of $\mathscr{R}$.

4.1.5. THEOREM. *If $R_1, R_2, \ldots, R_n$ $(n > 0)$ are $\Lambda$-characterizable subsets of $\mathscr{R}$, then there is a finite congruence $\mathbb{E}$ of $\mathscr{R}$ which characterizes $R_1, R_2, \ldots, R_n$ simultaneously.*

*Proof.* Given finite congruences $\mathbb{E}_1, \mathbb{E}_2, \ldots, \mathbb{E}_n$ which characterize $R_1, R_2, \ldots, R_n$, respectively, take their refinement $\mathbb{E} = \{E_1 \cap E_2 \cap \cdots \cap E_n \mid E_i \in \mathbb{E}_i (i = 1, 2, \ldots, n)\}$. ::

In the case of strings, the theorem can be interpreted as follows. Given a finite number of regular sets $S_1, S_2, \ldots, S_n$, one can find deterministic connected finite state automata $M_1, M_2, \ldots, M_n$ accepting sets $S_1, S_2, \ldots, S_n$, respectively, and sharing the components in common except the final states.

4.1.6. THEOREM. *The class of $\Lambda$-characterizable subsets of $\mathscr{R}$ is closed under Boolean operations.*

*Proof.* If $R_1$ and $R_2$ are in the class, take a finite congruence $\mathbb{E}$ of $\mathscr{R}$ which

characterizes both $R_1$ and $R_2$, i.e., $R_i = \cup \{E \in \mathbb{E} \mid E \subseteq R_i\}$ $(i = 1, 2)$. Then we get

$$R_1 \cup R_2 = \cup \{E \in \mathbb{E} \mid E \subseteq R_1 \cup R_2\}$$

and

$$\mathscr{R} - R_1 = \cup \{E \in \mathbb{E} \mid E \subseteq \mathscr{R} - R_1\}. \quad ::$$

4.2. *Congruences to Characterize T-Regular Sets*

We first introduce a binary operation "1" on $\mathscr{T}_A$. (Throughout this subsection we assume that $A$ is an alphabet of homogeneous rank $A \times \{0, 1, 2,...\}$.)

4.2.1. DEFINITION. Suppose $t, t' \in \mathscr{T}_A$ and $t' = a\langle t_1 t_2 \cdots t_k \rangle$ $(a \in A, k \geqslant 0, t_1, t_2, ..., t_k \in \mathscr{T}_A)$. Then we define $t1t'$ as the tree $a\langle tt_1 t_2 \cdots t_k \rangle$ in $\mathscr{T}_A$. ($t$ is added to $t'$ as the direct leftmost descendant of the root. For example, $a\langle b \rangle \, 1c\langle de \rangle = c\langle a\langle b \rangle de \rangle$.)

Note that the set $\mathscr{T}_A$ is the smallest set including $A$ and closed under 1 operation.

4.2.2. THEOREM. *A set $T \subseteq \mathscr{T}_A$ is T-regular if and only if $T$ is $\{1\}$-characterizable.*

*Proof.* Given a finite congruence of $\{1\}$-algebra $\mathscr{T}_A$ that characterizes set $T$, one can construct a $\overset{\Delta}{T}$-system $G$ and a projection $h$ such that $T = h(\overset{\Delta}{\mathscr{T}} G)$, and vice versa. (An alternate proof is to take the set of forests $W = \{w \in \mathscr{F}_A \mid a\langle w \rangle \in T \text{ for some } a \in A\}$ (cf. Section 2.4.2) and apply the characterization of $F$-regular sets (Section 4.3) to set $W$.) ::

4.2.3. COROLLARY. *The class of T-regular sets $\subseteq \mathscr{T}_A$ is closed under Boolean operations.* ::

4.3. *Congruences to Characterize F-Regular Sets*

As for the characterization of forest sets in $\mathscr{F}_A$, recall the two-dimensional concatenation $\varphi_a : \mathscr{F}_A \times \mathscr{F}_A \to \mathscr{F}_A$ defined as $\varphi_a(w_1, w_2) = a\langle w_1 \rangle w_2$ $(w_1, w_2 \in \mathscr{F}_A, a \in A)$, and let $\varphi_A = \{\varphi_a \mid a \in A\}$.

4.3.1. THEOREM. *A set $W \subseteq \mathscr{F}_A$ is F-regular if and only if $W$ is $\varphi_A$-characterizable.*

*Proof.* We show the theorem by taking binary tree representation of forests (Knuth, 1968). For alphabet $A$ with rank $r_A = A \times \{0, 1, 2,...\}$, let $A \cup \{\#\}$ be an alphabet with finite rank $\{(a, 2) \mid a \in A\} \cup \{(\#, 0)\}$, where $\#$

is a distinct symbol ($\notin A$). Define a function Bin: $\mathscr{F}_A \to \mathscr{T}_{A\cup\{\#\}}$ to convert forests over $A$ to binary trees over $A \cup \{\#\}$ as follows;

$$\begin{aligned}&\mathrm{Bin}(\epsilon) = \#\langle\ \rangle,\\ &\mathrm{Bin}(a\langle w_1\rangle w_2) = a\langle t_1 t_2\rangle,\\ &\quad \text{where } t_i = \mathrm{Bin}(w_i),\ w_i \in \mathscr{F}_A\ (i = 1, 2), \quad \text{and} \quad a \in A.\end{aligned}$$

Clearly the function is one-to-one and onto. Furthermore we can observe that a set $W \subseteq \mathscr{F}_A$ is $F$-regular if and only if $\mathrm{Bin}(W) \subseteq \mathscr{T}_{A\cup\{\#\}}$ is $T$-regular. (To substantiate the statement, it suffices to note that (1) the class of $F$-regular sets is the class of projective images of $\tilde{F}$-local sets, (2) $W \subseteq \mathscr{F}_K$ $(\# \notin K)$ is $\tilde{F}$-local if and only if $T = \mathrm{Bin}(W)$ is a $T$-local set (where Bin: $\mathscr{F}_K \to \mathscr{F}_{K\cup\{\#\}}$ is the function to yield binary tree representations of forests over $K$ in the same way as above) such that

$$\begin{aligned}&\text{if } X\langle Y_i Z_i\rangle \in \mathrm{Branch}(T)\ (X \subset K,\ Y_i, Z_i \in K \cup \{\#\},\ i = 1, 2) \text{ then}\\ &\qquad \{X\langle Y_1 Z_2\rangle,\ X\langle Y_2 Z_1\rangle\} \subseteq \mathrm{Branch}(T),\end{aligned}$$

and (3) the class of $T$-regular sets in $\mathscr{T}_{A\cup\{\#\}}$ is identical to the class of projective images of the kind of $T$-local sets described in (2). Propositions (1) and (2) are clear (Section 3.2.7 and from definitions). To see (3), suppose $T' \subseteq \mathscr{T}_{K'}$ be a $T$-local set such that a projective image $h'(T')$ is a $T$-regular set in $\mathscr{T}_{A\cup\{\#\}}$. Define a $T$-grammar $G = (K, P, I)$ by setting

$$\begin{aligned}K &= \{Xx \mid X\langle x\rangle \in \mathrm{Branch}(T')\},\\ P &= \{Xx\langle Yy Zz\rangle \mid X\langle x\rangle,\ Y\langle y\rangle,\ Z\langle z\rangle \in \mathrm{Branch}(T'),\ x = YZ\}\\ &\quad \cup \{Xx\langle\ \rangle \mid X\langle x\rangle = X\langle\ \rangle \in \mathrm{Branch}(T')\},\\ I' &= \{Xx \mid X\langle x\rangle \in \mathrm{Branch}(T'),\ X \in \mathrm{Root}(T')\}.\end{aligned}$$

Without loss of generality we assume $\# \notin K'$, and replace all the terminal symbols of trees in $\mathscr{T}G$ by symbol $\#$, obtaining set $T \subseteq \mathscr{T}_{K\cup\{\#\}}$. Then $T$ is shown to be $T$-local, to satisfy the condition in (2), and to yield the $T$-regular set $h'(T')$ as $h(T)$, where $h$: $\mathscr{T}_{K\cup\{\#\}} \to \mathscr{T}_{A\cup\{\#\}}$ is a projection defined by $h(Xx) = h'(X)$ $(Xx \in K,\ X \in K')$ and $h(\#) = \#$. Thus proposition (3) is verified.) On the other hand, it follows from definitions that $W$ is a $\varphi_A$-characterizable set in $\mathscr{F}_A$ if and only if $\mathrm{Bin}(W)$ is $\tau_{A\cup\{\#\}}$-characterizable in $\mathscr{T}_{A\cup\{\#\}}$. Hence we can conclude the theorem in the light of Theorem 4.1.4. ::

4.3.2. The Boolean closure property of $F$-regular sets (Pair *et al.*, 1968) follows from Theorems 4.1.6 and 4.3.1.

COROLLARY. *The class of $F$-regular sets is closed under Boolean operations.* ::

4.3.3. THEOREM. *Given $\tilde{F}$-systems $G_i$ and projections $h_i$ $(i = 1, 2)$, whether $h_1(\tilde{\mathscr{F}}G_1) = h_2(\tilde{\mathscr{F}}G_2)$ or not is decidable.*

*Proof.* Recollect the function Bin that converts forests to their binary tree representations, and note that given $\tilde{F}$-system $G = (K, P_0, P_1, P_2)$ with $\# \notin K$ one can effectively obtain a $T$-grammar $G' = (K \cup \{\#\}, P, I)$ to generate set $\mathrm{Bin}(\tilde{\mathscr{F}}G)$. (Indeed, let $P = \{X\langle YZ\rangle \mid X, Y, Z \in K,\ XY \in P_1,\ XZ \in P_2\} \cup \{X\langle \#Z\rangle \mid X \in K \cap P_1, Z \in K, XZ \in P_2\} \cup \{X\langle Y\# \rangle \mid X \in K \cap P_2,\ Y \in K, XY \in P_1\} \cup \{X\langle \#\# \rangle \mid X \in K \cap P_1 \cap P_2\} \cup \{\#\langle\ \rangle\}$ and $I = \{X \in K \mid X \in P_0\} \cup \{\# \mid \epsilon \in P_0\}$.)

Now that the function Bin is one-to-one, the equivalence problem posed upon $F$-regular sets is transfered to that of $T$-regular sets over a finitely ranked alphabet. The latter however is paraphrased as the equivalence problem of tree automata, and is known to be solvable (Thatcher *et al.*, 1968). ::

## 5. A SUBCLASS OF CF LANGUAGES, NEST SETS

Chomsky and Schützenberger (1963) proved a fundamental theorem of CF languages; the class of CF languages is identical to the class of homomorphic images of sets $L$ such that $L$ is the intersection of a standard regular event and a Dyck set. Taking advantage of our $F$-regular sets in structural study of CF languages, we will prove a refinement of the theorem.

First we define a subclass of CF languages over a paired alphabet, by restricting the form of production rules of CF grammars to $X \to \acute{a}Y\grave{a}Z$ and $X \to \epsilon$ (where $X$, $Y$, $Z$ are nonterminal symbols, and $\acute{a}$ and $\grave{a}$ are paired terminal symbols). Such CF languages, which are extensions of Dyck sets and named "nest sets," are shown to have distinctive properties and play an important role in the whole class of CF languages. Among others, it is shown that the class of nest sets over a paired alphabet forms a Boolean algebra having the Dyck set as the universe, and that nest sets are precisely sets of form $h(S \cap D)$, where $h$ is a "pair-preserving" projection, $S$ is an $S$-local set, and $D$ is a Dyck set. Talking about the relation of nest sets to two classes known in literature, the parenthesis languages of McNaughton (1967) and the bracketed CF languages of Ginsburg *et al.* (1967), we show that an extension of both classes is embedded into the class of nest sets. Then based on a known algorithm (to see the equivalence of tree automata), we prove that the equivalence problem for the "generalized parenthesis grammars" is solvable.

5.1. *Paired Alphabets and Nest Sets*

5.1.1. For an alphabet $A$, let $\acute{A} = \{\acute{a} \mid a \in A\}$ and $\grave{A} = \{\grave{a} \mid a \in A\}$. Assuming that the sets $\acute{A}$ and $\grave{A}$ are disjoint and the mappings $A \to \acute{A}$ and $A \to \grave{A}$ defined by $a \mapsto \acute{a}$ and $a \mapsto \grave{a}$ $(a \in A)$, respectively, are one-to-one, we call the union $\acute{A} \cup \grave{A} = \{\acute{a}, \grave{a} \mid a \in A\}$ the *paired alphabet* with *base* $A$, and the elements of $\acute{A}(\grave{A})$ the *left* (*right*, resp.) *symbols*. Then write $\hat{A} = \acute{A} \cup \grave{A}$.

5.1.2. Suppose $G = (N, \hat{A}, P, I)$ is a CF grammar with a paired alphabet $\hat{A}$ as the set of terminal symbols. ($N$ is the set of nonterminals, $P$ is the set of production rules, and $I \in N$ is the initial symbol. As for the definitions and conventional notations on CF grammars, see Hopcroft *et al.*, 1969.) If every rule in $P$ is either of form $X \to \acute{a}Y\grave{a}Z$ or $X \to \epsilon$ $(X, Y, Z \in N, a \in A)$, then the CF language $L(G)$ generated by $G$ is called a *nest set* over $\hat{A}$.

5.1.3. EXAMPLES. (1) Let $\hat{A} = \{\acute{a}, \grave{a}\}$ and $L = \{\acute{a}^n\grave{a}^n \mid n \geqslant 0\}$. Then the linear CF language $L$ is a nest set, which is generated by $G = (\{I, E\}, \hat{A}, \{I \to \acute{a}I\grave{a}E, E \to \epsilon, I \to \epsilon\}, I)$.

(2) The *Dyck set* over a paired alphabet $\hat{A}$ is the smallest set $L$ such that $L = \{\epsilon\} \cup \bigcup_{a \in A} \acute{a}L\grave{a}L$. The Dyck set is a nest set; indeed it is generated by CF grammar $G = (\{I\}, \hat{A}, P, I)$ such that $P = \{I \to \acute{a}I\grave{a}I \mid a \in A\} \cup \{I \to \epsilon\}$.

(3) Let $X$, $Y$, $Z$ be the smallest sets such that

$$X = \{\epsilon\} \cup \acute{a}Y\grave{a} \cup XX,$$
$$Y = \{\epsilon\} \cup \acute{b}Z\grave{b} \cup YY,$$
$$Z = \{\epsilon\} \cup \acute{c}X\grave{c} \cup ZZ,$$

and let $L = X$. ($L$ is a set of balanced strings of three kinds of parentheses in which the use of parentheses is cyclic.) Then $L$ is a nest set, for it is generated by CF grammar $G = (N, \hat{A}, P, X)$, where $N = \{X, Y, Z\}$, $A = \{a, b, c\}$ and $P = \{X \to \acute{a}Y\grave{a}X,\ Y \to \acute{b}Z\grave{b}Y,\ Z \to \acute{c}X\grave{c}Z,\ X \to \epsilon,\ Y \to \epsilon,\ Z \to \epsilon\}$.

5.2. *Nest Sets and F-Regular Sets*

5.2.1. In this subsection, we will study the nest sets by relating them to the $F$-regular sets. For this end, we first introduce a function Trace to convert forests to certain strings over a paired alphabet;

(1) $\mathrm{Trace}(\epsilon) = \epsilon$,

(2) $\mathrm{Trace}(a\langle w_1 \rangle w_2) = \acute{a} \cdot \mathrm{Trace}(w_1) \cdot \grave{a} \cdot \mathrm{Trace}(w_2)$ $(a \in A, w_1, w_2 \in \mathscr{F}_A)$.

For example, $\mathrm{Trace}(a) = \acute{a}\grave{a}$ and $\mathrm{Trace}(ab\langle c \rangle) = \acute{a}\grave{a}\acute{b}\acute{c}\grave{c}\grave{b}$. Notice that

string Trace($w$) for any $w \in \mathscr{F}_{\hat{A}}$ is a member of the Dyck set over $\hat{A}$, and that the image $\text{Trace}(\mathscr{F}_{\hat{A}}) = \{\text{Trace}(w) \mid w \in \mathscr{F}_{\hat{A}}\}$ equals the Dyck set. Therefore when we write $\mathscr{N}_{\hat{A}}$ for the Dyck set over $\hat{A}$, we have one-to-one onto function $\text{Trace}: \mathscr{F}_{\hat{A}} \to \mathscr{N}_{\hat{A}}$.

5.2.2. THEOREM. *If $W \subseteq \mathscr{F}_{\hat{A}}$ is F-regular, then* Trace($W$) *is a nest set over $\hat{A}$. Conversely for any nest set $L$ over $\hat{A}$, there is an F-regular set $W \subseteq \mathscr{F}_{\hat{A}}$ such that $L =$* Trace($W$).

*Proof of the first part.* Given $\tilde{F}$-system $(K, P_0, P_1, P_2)$ and projection $h: \mathscr{F}_K \to \mathscr{F}_{\hat{A}}$ such that $W = h(\tilde{\mathscr{F}}(K, P_0, P_1, P_2))$, we can define a CF grammar $G = (N, \hat{A}, P, I)$ satisfying the property $P \subseteq \{X \to \hat{a}Y\hat{a}Z, X \to \epsilon \mid X, Y, Z \in N, a \in A\}$ and $L(G) = \text{Trace}(W)$ as follows. First, using a new symbol $E$ not in $K$ (to represent the null string $\epsilon$), let $N' = K \cup \{E\}$ and

$$\begin{aligned} P' = {} & \{X \to \hat{a}Y\hat{a}Z \mid X, Y, Z \in K, XY \in P_1, XZ \in P_2, a = h(X) \in A\} \\ & \cup \{X \to \hat{a}E\hat{a}Z \mid X, Z \in K, X \in P_1, XZ \in P_2, a = h(X) \in A\} \\ & \cup \{X \to \hat{a}Y\hat{a}E \mid X, Y \in K, XY \in P_1, X \in P_2, a = h(X) \in A\} \\ & \cup \{X \to \hat{a}E\hat{a}E \mid X \in K \cap P_1 \cap P_2, a = h(X) \in A\} \\ & \cup \{E \to \epsilon\}. \end{aligned}$$

Then it is verified recursively that, for CF grammar $G_X = (N', \hat{A}, P', X)$ and $F$-regular set $W_X = h(\tilde{\mathscr{F}}(K, \{X\}, P_1, P_2))$ defined for each $X \in K$, $L(G_X) = \text{Trace}(W_X)$ holds. Next, adding one more symbol $I$ to $N'$ ($I \notin N'$, for the initial symbol), we define a CF grammar $G = (N, \hat{A}, P, I)$, where $N = N' \cup \{I\}$ and $P = P' \cup \{I \to x \mid X \to x$ is a rule in $P'$ for some $X \in K \cap P_0\} \cup \{I \to \epsilon \mid \epsilon \in P_0\}$. Then by examining the relations

$$L(G) = \left( \bigcup_{X \in K \cap P_0} L(G_X) \right) \cup (\{\epsilon\} \cap P_0),$$

and

$$W = \left( \bigcup_{X \in K \cap P_0} W_X \right) \cup (\{\epsilon\} \cap P_0),$$

we get $L(G) = \text{Trace}(W)$. The form of the rules in $P$ guarantees that the CF language $L(G)$ is a nest set. ::

*Proof of the second part.* Given a CF grammar $G = (N, \hat{A}, P, I)$ such that $L(G) = L$ and $P \subseteq \{X \to \hat{a}Y\hat{a}Z \mid X, Y, Z \in N, a \in A\} \cup \{X \to \epsilon \mid X \in N\}$, we can define $\tilde{F}$-system $(K, P_0, P_1, P_2)$ and projection $h: \mathscr{F}_K \to \mathscr{F}_{\hat{A}}$ so that

the $F$-regular set $W = h(\tilde{\mathscr{F}}(K, P_0, P_1, P_2))$ satisfies $\mathrm{Trace}(W) = L$, as follows;

$$K = \{(X \to \acute{a}Y\grave{a}Z) \mid X \to \acute{a}Y\grave{a}Z \text{ is a rule in } P,\ X, Y, Z \in N,\ a \in A\},$$
$$P_2 = \{(X \to \acute{a}Y\grave{a}Z)(Z \to z) \in KK \mid X, Y, Z \in N,\ a \in A,\ z \in \hat{A}N\hat{A}N\} \cup \{(X \to \acute{a}Y\grave{a}Z) \in K \mid Z \to \epsilon \text{ is a rule in } P,\ X, Y, Z \in N,\ a \in A\},$$
$$P_1 = \{(X \to \acute{a}Y\grave{a}Z)(Y \to y) \in KK \mid X, Y, Z \in N,\ a \in A,\ y \in \hat{A}N\hat{A}N\} \cup \{(X \to \acute{a}Y\grave{a}Z) \in K \mid Y \to \epsilon \text{ is a rule in } P,\ X, Y, Z \in N,\ a \in A\},$$
$$P_0 = \{(I \to \acute{a}Y\grave{a}Z) \in K \mid Y, Z \in N,\ a \in A\} \cup \{\epsilon \mid I \to \epsilon \text{ is a rule in } P\},$$
and $h((X \to \acute{a}Y\grave{a}Z)) = a\ (X, Y, Z \in N,\ a \in A)$. ::

It follows directly from the theorem that nest sets over a paired alphabet $\hat{A}$ are subsets of $\mathrm{Trace}(\mathscr{F}_A) = \mathscr{N}_{\hat{A}}$ (the Dyck set over $\hat{A}$). Another consequence of the theorem is the closure properties of the nest sets.

5.2.3. THEOREM. *The class of nest sets over paired alphabet $\hat{A}$ forms a Boolean algebra with $\mathscr{N}_{\hat{A}}$ as the universe.*

*Proof.* The one-to-one onto function Trace: $\mathscr{F}_A \to \mathscr{N}_{\hat{A}}$ can be naturally extended to the one that converts $F$-regular sets over $A$ to the nest sets over $\hat{A}$. Now that the class of $F$-regular sets over $A$ is closed under Boolean operations within the universe $\mathscr{F}_A$ (Section 4.3.2), so is the class of nest sets over $\hat{A}$ within the universe $\mathscr{N}_{\hat{A}}$. ::

5.2.4. Let $\hat{A}$ and $\hat{B}$ be two paired alphabets with base alphabets $A$ and $B$, respectively. We say a projection $h: (\hat{A})^* \to (\hat{B})^*$ is *pair-preserving* if $h$ satisfies the conditions;

(1) $h(\acute{A}) \subseteq \acute{B}$, $h(\grave{A}) \subseteq \grave{B}$,

(2) $h(\acute{a}) = \acute{b}$ if and only if $h(\grave{a}) = \grave{b}$ $(a \in A,\ b \in B)$.

5.2.5. THEOREM. *The class of nest sets is closed under pair-preserving projections and their inverses.*

*Proof.* For any pair-preserving projection $h: (\hat{K})^* \to (\hat{A})^*$ there is a projection $g: \mathscr{F}_K \to \mathscr{F}_A$ satisfying $h(\mathrm{Trace}(w)) = \mathrm{Trace}(g(w))$ $(w \in \mathscr{F}_K)$. Therefore the property of nest sets immediately follows from the closure property of $F$-regular sets under projections and their inverses (Pair *et al.*, 1968). ::

5.3. *Refinement of Chomsky–Schützenberger Theorem*

We first characterize the nest sets in terms of $S$-local sets and the Dyck sets.

5.3.1. THEOREM. *$L \subseteq (\hat{A})^*$ is a nest set if and only if there exist an $S$-local set $S \subseteq (\hat{K})^*$ and a pair-preserving projection $h\colon (\hat{K})^* \to (\hat{A})^*$ such that $L = h(S \cap \mathcal{N}_{\hat{K}})$.*

*Proof of only if.* First we note that for any $\tilde{F}$-local set $W$ over $K$ there is an $S$-local set $S$ over $\hat{K}$ such that $\mathrm{Trace}(W) = S \cap \mathcal{N}_{\hat{K}}$. (Indeed, using three sets $I \subseteq \hat{K}, F \subseteq \hat{K}$, and $P \subseteq \hat{K}\hat{K}$ such that $I = \{\hat{X} \mid X \in \mathrm{Origin}(W) \cap K\}$, $F = \{\hat{X} \mid X \in \mathrm{CO}(W) \cap K\}$, and $P = \{\hat{X}\hat{Y} \mid X, Y \in K, XY \in \mathrm{SO}(W)\} \cup \{\hat{X}\hat{X} \mid X \in \mathrm{SO}(W) \cap K\} \cup \{\hat{X}\hat{Y} \mid X, Y \in K, XY \in \mathrm{CO}(W)\} \cup \{\hat{X}\hat{Z} \mid X, Z \in K, X \in \mathrm{CO}(W)\}$, define an $S$-local set $S = I(\hat{K})^* \cap (\hat{K})^* F - (\hat{K})^* (\hat{K}\hat{K} - P)(\hat{K})^* \cup (W \cap \{\epsilon\})$. Then clearly $\mathrm{Trace}(W) = S \cap \mathcal{N}_{\hat{K}}$ holds.) Any nest set $L$ can be written as $\mathrm{Trace}(g(W))$, where $W$ is an $\tilde{F}$-local set and $g$ is a projection of forests (Sections 5.2.2 and 3.2.7). Hence by taking $S$-local set $S$ such that $\mathrm{Trace}(W) = S \cap \mathcal{N}_{\hat{K}}$ and using pair-preserving projection $h$ such that $\mathrm{Trace} \cdot g = h \cdot \mathrm{Trace}$ (Section 5.2.5), we can write $L = \mathrm{Trace}(g(W)) = h(\mathrm{Trace}(W)) = h(S \cap \mathcal{N}_{\hat{K}})$. ::

*Proof of if.* We first show that for any $S$-local set $S \subseteq (\hat{K})^*$ the intersection $S \cap \mathcal{N}_{\hat{K}}$ is equal to set $\mathrm{Trace}(W)$, where $W$ is an $F$-local set. Let $S' = S \cap \mathcal{N}_{\hat{K}}$, $I' = \{X \subset K \mid \hat{X}(\hat{K})^* \cap S' \neq \varnothing\}$, $F' = \{X \in K \mid (\hat{K})^* \hat{X} \cap S' \neq \varnothing\}$, $P' = \{XY \mid X, Y \in K, \hat{X}\hat{Y} \in \mathrm{CO}(S')\}$, and $I = I'K^* \cap K^*F' - K^*(KK - P')K^*$. Also for each $X \in K$ let $I_X = \{Y \in K \mid \hat{X}\hat{Y} \in \mathrm{CO}(S')\}$, $F_X = \{Y \in K \mid \hat{Y}\hat{X} \in \mathrm{CO}(S')\}$, and $P = \{X\langle s \rangle \mid X \in K, s \in I_X K^* \cap K^* F_X - K^*(KK - P')K^*\} \cup \{X \in K \mid \hat{X}\hat{X} \in \mathrm{CO}(S')\}$. Then for the $F$-local set $W$ generated by $F$-grammar $(K, P, I)$, clearly $S \cap \mathcal{N}_{\hat{K}} \subseteq \mathrm{Trace}(W)$ holds. The $S$-locality of set $S$ guarantees the reverse inclusion. Thus $S \cap \mathcal{N}_{\hat{K}} = \mathrm{Trace}(W)$ is verified. Since $\mathrm{Trace}(W)$ is known to be a nest set (Section 5.2.2), so is the image $h(\mathrm{Trace}(W)) = h(S \cap \mathcal{N}_{\hat{K}})$ by pair-preserving projection $h$ (Section 5.2.5). ::

5.3.2. Given CF language $L \subseteq A^*$, let $G$ be a CF grammar generating set $L - \{\epsilon\}$ without $\epsilon$-rules, $T$ be the set of derivation trees of $G$, and let $L' = \mathrm{Trace}(T) \cup (L \cap \{\epsilon\})$. Then $L'$ is a nest set over $\hat{B}$ where $B$ is the vocabulary of $G$, and $L$ can be expressed as a homomorphic image of $L'$. (Indeed, using homomorphism $f\colon (\hat{B})^* \to A^*$ such that $f(\hat{a}) = a$, $f(\dot{a}) = f(\hat{X}) = f(\dot{X}) = \epsilon$ $(a \in A, X \in B - A)$, we can write $L = f(L')$.) This together with last theorem implies; any CF language is a homomorphic image of the intersection of an $S$-local set and a Dyck set (cf. Chomsky *et al.*, 1963).

5.3.3. Summarizing the foregoing discussions, we have the following relation between three classes of string sets:

$$\{S \cap D \mid S \text{ is an } S\text{-local set}, D \text{ is a Dyck set}\}$$
$$\downarrow \text{ (via pair-preserving projections)}$$
$$\{\text{nest sets}\} = \{\text{Trace}(W) \mid W \text{ is an } F\text{-regular set}\}$$
$$\downarrow \text{ (via homomorphisms)}$$
$$\{\text{CF languages}\}$$

5.4. *Parenthesis Languages and Nest Sets*

5.4.1. Parenthesis grammars (McNaughton, 1967) are defined to be CF grammars $G = (N, A, P, I)$ such that $A$ contains a pair of brackets, say [and], and the rules in $P$ are restricted to the form $X \to [x]$, where $x$ contains neither [nor].

Now let us consider a type of CF grammars $G = (N, A, P, I)$ such that the alphabet $A$ contains paired symbols $\{\acute{b}, \grave{b} \mid b \in B\} = \hat{B}$ and each rule in $P$ is either one of the following forms;

$$X \to \acute{b}x\grave{b}, \qquad X \to \acute{b}x\grave{b}Y, \qquad X \to c, \qquad X \to cY, \qquad X \to \epsilon,$$

where

$$b \in B, \qquad x \in (N \cup A - \hat{B})^*, \qquad X, Y \in N, \qquad \text{and} \qquad c \in A - \hat{B}.$$

Clearly the parenthesis grammars can be considered as being of this type (by viewing [and] as the paired symbols in $\hat{B}$). Hence we will call the type of CF grammar a *generalized parenthesis grammar*, and the CF language generated thereby a *generalized parenthesis language*. As an example of a generalized parenthesis language which is not a parenthesis language, one can think of a set of arithmetic expressions, in FORTRAN, for instance (cf. Example 5.4.4.).

5.4.2. It is known (McNaughton, 1967; Knuth, 1967) that the equivalence problem for the class of parenthesis grammars is solvable. We claim that the same is true for the class of generalized parenthesis grammars. To prove the claim we will relate the class to that of nest sets (and therefore to that of $F$-regular sets), so that we can apply a result known for the latter.

5.4.3. THEOREM. *The equivalence problem for generalized parenthesis grammars is solvable.*

*Proof.* Given generalized parenthesis grammar $G$, we first convert it to an equivalent CF grammar $(N, A, P, I)$ such that $P \subseteq \{X \to \acute{b}x\grave{b}Y, X \to cY,$

$X \to \epsilon \mid X, Y \in N, b \in B, x \in N^*, c \in C\}$ and $A = \hat{B} \cup C$. (The conversion is straightforward and effective.) Then assume that $f$: $(\hat{B} \cup C)^* \to (\hat{D})^*$ is a one-to-one homomorphism such that $\hat{B} \subseteq \hat{D}, f(a) = a\ (a \in \hat{B})$, and $f(c) = \acute{d}\grave{d}$ $(c \in C,\ d \in D)$. (If $B \cap C = \varnothing$ to begin with, one may set $D = B \cup C$ and $f(c) = \acute{c}\grave{c}$ $(c \in C)$.) We will prove the theorem by showing (1) the image $f(L)$ of the generalized parenthesis language $L = L(G)$ is a nest set over $\hat{D}$, i.e., there is an $F$-regular set $W$ over $D$ such that $f(L) = \mathrm{Trace}(W)$, and (2) the set $W$ is effectively specified in terms of $\tilde{F}$-system and projection. Once the two goals are established, we can solve the equivalence problem by way of Theorem 4.3.3 (by transforming $W$ to a set of binary trees and then applying the minimization technique of tree automata). Remind that the correspondence of $L$ to $W$ specified by $f(L) = \mathrm{Trace}(W)$ is one-to-one.

To see (1), let us first define an alphabet $K = \{(X \to \acute{b}x\grave{b}Y) \in P \mid X, Y \in N, b \in B, x \in N^*\} \cup \{(X \to \acute{d}\grave{d}Y) \mid (X \to cY) \in P, X, Y \in N, c \in C, f(c) = \acute{d}\grave{d}, d \in D\}$. For each $p = (X \to \acute{d}x\grave{d}Y) \in K$ $(X, Y \in N, d \in D, x \in N^*)$, for the sake of convenience we call the $X$, $\acute{d}x\grave{d}$, and $Y$, respectively, by $\mathrm{left}(p)$, $\mathrm{mid}(p)$, and $\mathrm{right}(p)$. Next, for each $X \in N$ we define an $S$-local set

$$S_X = \left\{ p_1 p_2 \cdots p_k \;\middle|\; \begin{array}{l} p_i \in K \quad (1 \leqslant i \leqslant k), \quad \mathrm{left}(p_1) = X, \\ \mathrm{right}(p_i) = \mathrm{left}(p_{i+1}) \quad (1 \leqslant i < k), \\ \mathrm{right}(p_k) \to \epsilon \text{ is a rule in } P \end{array} \right\}$$
$$\cup \{\epsilon \mid X \to \epsilon \text{ is a rule in } P\},$$

and extend the definition by $S_{xy} = S_x S_y$ $(x, y \in N^*)$ and $S_\epsilon = \{\epsilon\}$. (Then for each $x \in N^*$ the set $S_x \subseteq K^*$ is regular since it is a finite concatenation of regular sets.) We are now in a position to define the $F$-regular set $W$ such that $\mathrm{Trace}(W) = f(L)$, by means of an $F$-grammar and a projection. Let $G' = (K, P', I')$ be an $F$-grammar with rules

$$P' = \{p\langle z\rangle \mid p = (X \to \acute{d}x\grave{d}Y) \in K, z \in S_x (X, Y \in N, d \in D, x \in N^*)\}$$

and with initial strings $I' = S_I$; let $h'$: $\mathscr{F}_K \to \mathscr{F}_D$ be a projection such that $h'(p) = d$, where $\mathrm{mid}(p) = \acute{d}x\grave{d}$ $(p \in K,\ d \in D,\ x \in N^*)$; and let $W = h'(\mathscr{F}G')$. (By induction on the "depth" of parenthesis, we can verify the equality $\mathrm{Trace}(W) = f(L)$.)

To see (2), we first assign to each $p \in K$ a distinct alphabet $K_p$, an $S$-local set $O_p \subseteq (K_p)^*$, and a projection $h_p : (K_p)^* \to K^*$ such that $K$ and all $K_p$'s $(p \in K)$ are mutually disjoint, and $h_p(O_p) = \{z \in K^* \mid p\langle z\rangle \in P'\}$. (In effect, when $\mathrm{mid}(p) = \acute{d}X_1X_2 \cdots X_j\grave{d}$ $(X_i \in N\ (1 \leqslant i \leqslant j), d \in D)$, let

$$K_p = \{q_{p,i} \mid q \in K, 1 \leqslant i \leqslant j\} \quad \text{and} \quad O_p = f_{p,1}(S_{X_1}) f_{p,2}(S_{X_2}) \cdots f_{p,j}(S_{X_j}),$$

where $f_{p,i}: K^* \to (K_p)^*$ is a projection defined by $f_{p,i}(q) = q_{p,i}$ $(q \in K, 1 \leqslant i \leqslant j)$. Then using a projection $h_p$ defined by $h_p(q_{p,i}) = q$ $(q_{p,i} \in K_p)$, we can get $h_p(O_p) = S_{X_1} S_{X_2} \cdots S_{X_j} = \{z \in K^* \mid p\langle z\rangle \in P'\}$. In case of $\mathrm{mid}(p) = d\hat{d}$ $(d \in D)$ we set $K_p = \varnothing$ and $O_p = \{\epsilon\}$.) Secondly, let $K'' = \bigcup_{p \in K} K_p \cup K$ and define a projection $h''$: $\mathscr{F}_{K''} \to \mathscr{F}_K$ by setting $h''(Z) = h_p(Z)$ $(Z \in K_p, p \in K)$ and $h''(Z) = Z$ $(Z \in K)$. Third, define $F$-grammar $G'' = (K'', P'', I'')$ by $P'' = \{Z\langle z\rangle \mid Z \in K'', h(Z) = q \in K, z \in O_q\}$ and $I'' = I'$. Then clearly $h''(\mathscr{F}G'') = \mathscr{F}G'$ holds. Moreover from the construction, we can observe that the set $\mathscr{F}G''$ is $\tilde{F}$-local (cf. Sections 3.2.7 and 3.1.4). Finally by taking $\tilde{F}$-system $(K'', P_0, P_1, P_2)$ to generate set $\mathscr{F}G''$ (i.e., by setting $P_0 = \mathrm{Origin}(\mathscr{F}G'')$, $P_1 = \mathrm{SO}(\mathscr{F}G'')$ and $P_2 = \mathrm{CO}(\mathscr{F}G'')$), and taking the projection $h'h''$, we can express the set $W$ as

$$h'h''(\tilde{\mathscr{F}}(K'', P_0, P_1, P_2)) = h'h''(\mathscr{F}G'') = h'(\mathscr{F}G') = W.$$

Note that the course of conversion from generalized parenthesis grammar $G$ to $\tilde{\mathscr{F}}$-system $(K'', P_0, P_1, P_2)$ and projection $h'h''$ is effective. ::

5.4.4. Example. Consider a set $L$ of arithmetic expressions generated by CF grammar $(N, A, P, I)$ with $N = \{I, T\}$, $A = \{u, v, +, \times, [,]\}$, and $P = \{T \to u,\ T \to v,\ T \to T \times T,\ T \to [I + T],\ I \to T,\ I \to I + T\}$. The same set $L$ can be also generated by CF grammar $(\{I, Z\}, A, P', I)$ with $P' = \{I \to uZ,\ I \to vZ,\ I \to [I + I]Z,\ Z \to \epsilon,\ Z \to +I,\ Z \to \times I\}$, and hence it is a generalized parenthesis language. Suppose $f$: $A^* \to (\hat{D})^*$ is a homomorphism such that $f(c) = \hat{c}\hat{c}$ $(c \in A - \{[\ ,\ ]\})$, $f([) = \hat{b}$, $f(]) = \hat{b}$ and $D = \{u, v, +, \times, b\}$. Then $f$ is one-to-one and the image $f(L)$ is a nest set over $\hat{D}$. Taking an $F$-regular set $W = \mathscr{F}(D, P'', I'')$, where

$$I'' = \{u, v, b\}\{+u, +v, +b, \times u, \times v, \times b\}^*$$

and

$$P'' = \{b\langle z\rangle \mid z \in I'' + I''\} \cup \{u\langle\ \rangle, v\langle\ \rangle, +\langle\ \rangle, \times\langle\ \rangle\},$$

we get $f(L) = \mathrm{Trace}(W)$. For example, string $s = [u \times [u + v] + v] \times [u + v + v]$ in $L$ corresponds to $w = b\langle u \times b\langle u + v\rangle + v\rangle \times b\langle u + v + v\rangle$ in $W$ as

$$f(s) = \hat{b}\hat{u}\hat{u}\hat{\times}\hat{\times}\hat{b}\hat{u}\hat{u}\hat{+}\hat{+}\hat{v}\hat{v}\hat{b}\hat{+}\hat{+}\hat{v}\hat{v}\hat{b}\hat{\times}\hat{\times}\hat{b}\hat{u}\hat{u}\hat{+}\hat{+}\hat{v}\hat{v}\hat{+}\hat{+}\hat{v}\hat{v}\hat{b} = \mathrm{Trace}(w).$$

5.4.5. The class of generalized parenthesis grammars by definition includes the bracketed CF grammars of Ginsburg *et al.* (1967), the CF grammars for nest sets specified in Section 5.1.2, the right linear grammars

for regular sets, as well as the (conventional) parenthesis grammars. Consequently the equivalence question for any combination of these grammars may be answered by way of the procedure described.

## 6. Büchi's Canonical Systems and Their Extensions

Büchi (1964) defined a type of string generating systems named "regular canonical systems," and showed that the class of sets generated by the systems is identical to the class of regular sets. The systems, having a finite number of "axioms" and "replacement rules," derives a new string from what is previously obtained or from one of the axioms by replacing a tail segment. We find that in showing the equivalence of the two classes the notion of local sets is quite helpful, and also that taking advantage of our unified framework the argument can be readily extended to the sets of trees and forests. (Brainerd (1969) has the extension to tree sets over a finitely ranked alphabet.) In this section after exhibiting the general idea of the systems in case of strings, we will discuss the extension to forests. Then we revisit the universe of strings, adopting the idea to the nest sets.

### 6.1. *Canonical Systems for Regular Sets*

6.1.1. Definition. An *S-canonical system* is a triple $(A, Q, R)$ of finite sets such that $Q \subseteq A^*$ (set of axioms) and $R \subseteq A^* \times A^*$ (set of replacement rules). The set $\mathscr{S}^c(A, Q, R)$ of strings generated by the system is defined as the smallest set $S \subseteq A^*$ such that

(1) $Q \subseteq S$,

(2) if $(s', s'') \in R$ and $ss' \in S$ then $ss'' \in S$ $(s, s', s'' \in A^*)$.

In particular, if $R$ has the property that for each $(s', s'') \in R$ the $s'$ is a tail segment of $s''$ (i.e., $s'' \in A^*s'$), then the system $(A, Q, R)$ is called a *periodic S-canonical system.*

6.1.2. Example. Let $A = \{a, b\}$. The set $L$ of all strings over $A$ which have five or less consecutive $a$'s (i.e., set $A^* - A^*aaaaaaA^*$) is obtained as $\mathscr{S}^c(A, Q, R)$, where $Q = \{aaaaab\}$, $R = \{(b, baaaaab), (ab, b), (b, \epsilon)\}$. For instance, string *aaaba* in $L$ is generated by the system via *aaaaab* $(\in Q)$, *aaaab*, *aaab*, *aaabaaaaab*, *aaabaaaab*,..., *aaabab*, *aaaba*. (By definition, the intermediate strings also belong to $L$.)

6.1.3. THEOREM (Büchi). *The followings are equivalent*;

(1) $S$ *is a regular set in* $A^*$.

(2) $S = \mathscr{S}^c(A, Q, R)$ *for an S-canonical system* $(A, Q, R)$.

(3) $S = \mathscr{S}^c(A, Q, R)$ *for a periodic S-canonical system* $(A, Q, R)$.

(Apart from the original proof, we can derive the theorem from Theorem 6.2.3, by considering strings as a special type of forests (namely, as the forests $w$ such that $w = \mathrm{Root}(w)$).) ::

6.2. *Canonical Systems for F-Regular Sets*

6.2.1. DEFINITION. For $w, w', w'' \in \mathscr{F}_A$, Replace$(w, w', w'')$ means the set of forests in $\mathscr{F}_A$ which are obtained from $w$ by replacing any number of occurrences of subforest $w'$ by $w''$. More precisely,

(1) when $w = w'$, Replace$(w, w', w'') = \{w, w''\}$,

(2) when $w = \epsilon \neq w'$, Replace$(w, w', w'') = \{w\}$,

(3) when $w = a\langle w_1\rangle\, w_2 \neq w'$ $(w_1, w_2 \in \mathscr{F}_A, a \in A)$,
Replace$(w, w', w'') = \{a\langle x_1\rangle\, x_2 \mid x_i \in \mathrm{Replace}(w_i, w', w''), i = 1, 2\}$.

6.2.2. DEFINITION. Suppose $A, Q$, and $R$ are finite sets such that $Q \subseteq \mathscr{F}_A$ and $R \subseteq \mathscr{F}_A \times \mathscr{F}_A$. Then the triple $(A, Q, R)$ is called an *F-canonical system.* The set of forests generated by the system, $\mathscr{F}^c(A, Q, R)$, is defined as the smallest set $W \subseteq \mathscr{F}_A$ such that

(1) $Q \subseteq W$,

(2) if $w \in W$ and $(w', w'') \in R$, then Replace$(w, w', w'') \subseteq W$.

In particular, if $w'$ is a subforest of $w''$ in the rules $(w', w'') \in R$, then the system $(A, Q, R)$ is said to be *periodic.* (For each $w \in \mathscr{F}_A$, we define Subforest$(w) \subseteq \mathscr{F}_A$ by

(1) Subforest$(\epsilon) = \{\epsilon\}$

(2) Subforest$(w) = \{w\} \cup$ Subforest$(w_1) \cup$ Subforest$(w_2)$
if $w = a\langle w_1\rangle\, w_2$, $a \in A$, and $w_1, w_2 \in \mathscr{F}_A$;

and call the members of Subforest$(w)$ the *subforests* of $w$.)

6.2.3. THEOREM. *The followings are equivalent*;

(1) $W \subseteq \mathscr{F}_A$ *is an F-regular set.*

(2) $W = \mathscr{F}^c(A, Q, R)$ *for an F-canonical system* $(A, Q, R)$.

(3) $W = \mathscr{F}^c(A, Q, R)$ *for a periodic F-canonical system* $(A, Q, R)$.

*Proof of* (2) $\Rightarrow$ (1). Given $F$-canonical system $(A, Q, R)$, let $Q'' = \{w'' \mid (w', w'') \in R \text{ for some } w'\}$ and define a new alphabet $K = \{X_q^w \mid w \in Q \cup Q'', q \in D(w)\}$, where $X_q^w \neq X_{q'}^{w'}$ unless $w = w'$ and $q = q'$. For $w \in Q \cup Q''$, let $\bar{w}$ represent the forest obtained from $w$ by replacing each symbol $w(q) \in A$ by symbol $X_q^w \in K$ $(q \in D(w))$. (In other words, $\bar{w}$ is the forest over $K$ such that $D(\bar{w}) = D(w)$ and $\bar{w}(q) = X_q^w$ for each $q \in D(\bar{w})$.) Also define a projection $h: \mathscr{F}_K \to \mathscr{F}_A$ by $h(X_q^w) = w(q)$ $(X_q^w \in K)$, so that $h(w) = w$ holds for each $w \in Q \cup Q''$. Finally, define a sequence $\bar{Q}_0, \bar{Q}_1, \bar{Q}_2, \ldots$ of forest sets in $\mathscr{F}_K$ by

$$\bar{Q}_0 = \{\bar{w} \mid w \in Q\},$$
$$\bar{Q}_m = \cup \{\text{Replace}(w, x, \bar{y}) \mid w \in \bar{Q}_{m-1}, (h(x), y) \in R\} \quad (m > 0),$$

and let $\bar{Q}_\infty = \bigcup_{m=0}^{\infty} \bar{Q}_m$. Then from the construction clearly $h(\bar{Q}_\infty) = \mathscr{F}^c(A, Q, R)$ holds. Hence the proof now amounts to show the $\tilde{F}$-locality of set $\bar{Q}_\infty$. For this end, we remark the following characterization of $\tilde{F}$-local sets.

6.2.4. LEMMA. *$W \subseteq \mathscr{F}_K$ is an $\tilde{F}$-local set if and only if for any $X\langle y_i\rangle z_i \in \text{Subforest}(W)$ $(X \in K, y_i, z_i \in \mathscr{F}_K, i = 1, 2)$, and any $w \in W$, the inclusions $\text{Replace}(w, X\langle y_1\rangle z_1, X\langle y_1\rangle z_2) \subseteq W$ and $\text{Replace}(w, X\langle y_1\rangle z_1, X\langle y_2\rangle z_1) \subseteq W$ hold.*

*Proof.* To see "if" part, let $U_X = \{x \in \text{Subforest}(W) \mid \text{Origin}(x) = X\}$ for each $X \in K \cup \{\epsilon\}$. Then it follows from the property of set $W$ that $U_X = \tilde{\mathscr{F}}(K, \{X\}, \text{SO}(W), \text{CO}(W))$ $(X \in K \cup \{\epsilon\})$. (The proof is by induction on the number of nodes in forests.) Hence, by examining

$$W = \cup \{U_X \mid X \in \text{Origin}(W)\} = \tilde{\mathscr{F}}(K, \text{Origin}(W), SO(W), CO(W)),$$

we can conclude that $W$ is $\tilde{F}$-local. The converse is clear from the definition. :: (Remark; this is an extension of Theorem 1.2.1.)

6.2.5. *Continuation of Section* 6.2.3. *Proof of* (1) $\Rightarrow$ (3). Suppose $W \subseteq \mathscr{F}_A$ is an $F$-regulae set, $\mathbb{E}$ is a finite congruence of $\varphi_A$-algebra $\mathscr{F}_A$ which characterizes set $W$ (cf. Section 4.3.1), and $m$ is the cardinality of $\mathbb{E}$. For $w \in \mathscr{F}_A$ let $\text{size}(w)$ be the integer such that $\text{size}(\epsilon) = 0$, $\text{size}(a\langle w_1\rangle w_2) = 1 + \max\{\text{size}(w_i) \mid i = 1, 2\}$ $(w_1, w_2 \in \mathscr{F}_A, a \in A)$. Define a periodic $F$-canonical system $(A, Q, R)$ by

$$Q = \{w \in W \mid \text{size}(w) \leqslant m\}$$
$$R = \left\{(w', w'') \;\middle|\; \begin{array}{l} w'' \in \text{Subforest}(W), w' \in \text{Subforest}(w'') \\ \text{size}(w'') \leqslant 2m, \text{size}(w') \leqslant m, w' \text{ and } w'' \text{ are in a same block of } \mathbb{E} \end{array}\right\}.$$

Then we can show that $W = \mathscr{F}^c(A, Q, R)$ holds. ::

6.3. *Canonical Systems for Nest Sets*

Recall function Trace to convert forests over $A$ to strings over $\hat{A} = \{\acute{a}, \grave{a} \mid a \in A\}$ and to convert $F$-regular sets over $A$ to nest sets over $\hat{A}$ (cf. Sections 5.2.1 and 5.2.2). Now applying the function to the components of $F$-canonical systems, we readily come to the notion of canonical systems for nest sets.

6.3.1. DEFINITION. An *N-canonical system* $G = (\hat{A}, Q, R)$ consists of a paired alphabet $\hat{A}$, a finite set $Q \subseteq \mathcal{N}_{\hat{A}}$ (of axioms), and a finite set $R \subseteq \mathcal{N}_{\hat{A}} \times \mathcal{N}_{\hat{A}}$ (of replacement rules), where $\mathcal{N}_{\hat{A}}$ is the Dyck set over $\hat{A}$. The set $\mathcal{N}^c G$ of strings generated by $G$ is defined as the smallest set $L$ such that

(1) $Q \subseteq L$,

(2) if $xyz \in L$, $(y, y') \in R$, and $z \in \{\epsilon\} \cup \grave{A}(\hat{A})^*$, then $xy'z \in L$.

In other words, the replacement rules are applicable to a segment $y \in \mathcal{N}_{\hat{A}}$ of sentence $xyz$, only if $y$ is not followed by a left symbol. (The restriction means that the segment $y$ should corresponds to a subforest of forest $w$ such that $\mathrm{Trace}(w) = xyz$.)

6.3.2. THEOREM. *$L \subseteq (\hat{A})^*$ is a nest set if and only if there is an N-canonical system $G = (\hat{A}, Q, R)$ such that $L = \mathcal{N}^c G$.*

*Proof.* For any $F$-canonical system $G = (A, Q, R)$, $G' = (\hat{A}, \mathrm{Trace}(Q), \{(\mathrm{Trace}(w'), \mathrm{Trace}(w'')) \mid (w', w'') \in R\})$ is an $N$-canonical system and satisfies $\mathcal{N}^c G' = \mathrm{Trace}(\mathcal{F}^c G)$. Likewise any $N$-canonical system is converted to an $F$-canonical system by applying $\mathrm{Trace}^{-1}$ to the components. ::

6.3.3. EXAMPLES. (1) Consider an $N$-canonical system $G = (\hat{A}, Q, R)$, where $Q = \{\epsilon\}$ and $R = \{(\epsilon, \acute{a}\grave{a}) \mid a \in A\}$. Then the set $\mathcal{N}^c G$ is equal to the Dyck set $\mathcal{N}_A$. (For instance, string $\acute{a}\acute{b}\grave{b}\grave{a}\acute{c}\grave{c}\acute{d}\grave{d}$ $(a, b, c, d \in A)$ can be generated by the system via strings $\epsilon(\in Q)$, $\acute{a}\grave{a}$, $\acute{a}\acute{b}\grave{b}\grave{a}$, $\acute{a}\acute{b}\grave{b}\grave{a}\acute{c}\grave{c}$, $\acute{a}\acute{b}\grave{b}\grave{a}\acute{c}\grave{c}\acute{d}\grave{d}$.)

(2) Let $\hat{A} = \{\acute{a}, \grave{a}\}$ and $L$ be the smallest set such that $L = \{\epsilon\} \cup \acute{a}L\grave{a} \cup \acute{a}L\grave{a}\acute{a}L\grave{a}$. Then $L$ is a nest set and can be generated by $N$-canonical system $G = (\hat{A}, Q, R)$, where

$$Q = \{\acute{a}\grave{a}\acute{a}\grave{a}\}, \qquad R = \{(\acute{a}\grave{a}, \epsilon), (\acute{a}\grave{a}\acute{a}, \acute{a}\grave{a}\acute{a}\grave{a}\acute{a}\grave{a}\acute{a}\grave{a})\}.$$

## 7. Pushdown Tracers

As tracing devices of forests, we will study a subclass of pushdown automata called "pushdown tracers." First we define the subclass and describe how they trace forests. Then we show an exact correspondence between $F$-regular sets and the sets traced by the devices. Finally we discuss an implication of the tracers with respect to syntax-directed compilation of programming languages.

### 7.1. *A Subclass of Pushdown Automata, Pushdown Tracers*

A pushdown tracer is a pushdown automaton such that at each transition a single input symbol is read in and a single pushdown symbol is either pushed-down or popped-up.

7.1.1. Definition. A *pushdown tracer* (or *P-tracer*) is a 6-tuple $M = (K, B, C, \delta, I, F)$ of finite sets such that $\delta \subseteq K \times K \times B \times C \times \{\downarrow, \uparrow\}$ and $I, F \subseteq K$. In $M$, $K$ represents the set of states, $B$ is the input alphabet, $C$ is the pushdown alphabet, $I$ is the set of initial states, and $F$ is the set of final states. The transition rules of $M$ are specified by elements of $\delta$; rule $(X, Y, b, c, \downarrow)$ in $\delta$ means that $M$ at current state $X$ seeing input symbol $b$ can move to state $Y$ while pushing-down a pushdown symbol $c$, whereas rule $(X, Y, b, c, \uparrow)$ in $\delta$ means that $M$ at state $X$ with input symbol $b$ can move to state $Y$ while popping-up symbol $c$ from the pushdown tape.

7.1.2. Suppose $M = (K, B, C, \delta, I, F)$ is a pushdown tracer, $\hat{A}$ is a paired alphabet, and $h: (\hat{A})^* \to B^*$ is a projection (i.e., a homomorphism such that $h(\hat{A}) \subseteq B$). A forest $w$ in $\mathscr{F}_A$ is then said to be *traced by $M$ with respect to $h$*, if there exist non-null sequence $X_0, X_1, \ldots, X_n$ of states $(\epsilon K)$, sequence $c_1, c_2, \ldots, c_n$ of pushdown symbols $(\epsilon C)$, sequence $s_0, s_1, \ldots, s_n$ of pushdown words $(\epsilon C^*)$ such that

(1) $X_0 \in I$, $s_0 = \epsilon$,

(2) For $i = 1, 2, \ldots, n$,

either $(X_{i-1}, X_i, b_i, c_i, \downarrow) \in \delta$ and $s_{i-1}c_i = s_i$ (push-down $c_i$)
or $(X_{i-1}, X_i, b_i, c_i, \uparrow) \in \delta$ and $s_{i-1} = s_i c_i$ (pop-up $c_i$),
where $b_1, b_2, \ldots, b_n \in B$ and $b_1 b_2 \cdots b_n = h(\mathrm{Trace}(w))$,

(3) $X_n \in F$, $s_n = \epsilon$.

We will write $\mathscr{P}(M, h)$ for the set of all forests traced by $M$ with respect to projection $h$. In other words, $\mathscr{P}(M, h)$ is the set of forests $w$ such that the

string $h(\mathrm{Trace}(w))$ is accepted by pushdown automoton $M$. Note that the length of any string accepted by the pushdown automaton $M$ is an even integer.

7.1.3. The movement of $P$-tracer $M$ over a forest $w$, when $M$ traces $w$, is exactly to follow the path lead by function Trace. Namely, first $M$ starts at the top left corner of $w$, and (1) $M$ goes downward through the leftmost branching lines as far as it can. Then on arriving at a terminal node, (2) if there is any node to the right in the same branch, $M$ moves to the nearest one and continues the process (1). If there is none (that is, $M$ is at the rightmost node of a branch), then $M$ moves upward through the rightmost branching line and come back to the node dominating the branch. Then $M$ continues the process (2), until at last $M$ reaches the top right corner of $w$.

For example, when a $P$-tracer $M$ traces the forest $w$ of Fig. 6a, $M$ traverses $w$ by taking arrowed lines 1–12 of Fig. 6b. As explained below in detail, the upright moves (in the example, arrows 1, 2, 4, 5, 7, 10–12) are controled by pushdown storage, whereas the level moves (arrows 3, 6, 8, and 9) are under the finite state control.

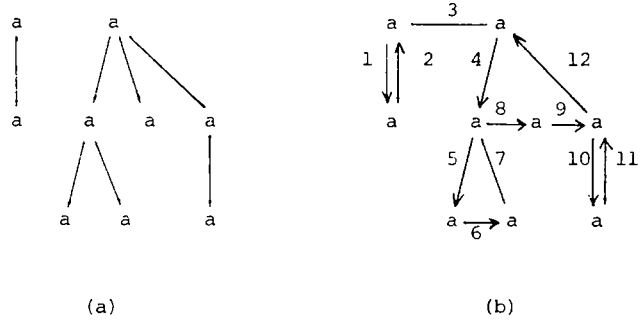


FIG. 6. Forest $w = a\langle a\rangle\, a\langle a\langle aa\rangle\, aa\langle a\rangle\rangle$, and the traversing route of $w$ by pushdown tracers.

## 7.2. *Synthesis and Analysis of Pushdown Tracers*

7.2.1. THEOREM. *For any F-regular set $W \subseteq \mathscr{F}_A$ and any projection $h$: $(\hat{A})^* \to B^*$, there is a P-tracer $M$ such that $\mathscr{P}(M, h) = W$.*

*Proof.* Suppose $G = (K, P_0, P_1, P_2)$ is an $\tilde{F}$-system and $g$: $\mathscr{F}_K \to \mathscr{F}_A$ is a projection such that $W = g(\tilde{\mathscr{F}}G)$. Then using the alphabet $K$ as the set of pushdown symbols and using the paired alphabet $\hat{K}$ together with a new

symbol $\$ (\notin \hat{K})$ as the set of states, we define a $P$-tracer $M = (\hat{K} \cup \{\$\}, B, K, \delta, I, \{\$\})$ where

$$\begin{aligned}\delta = {} & \{(\acute{X}, \grave{Y}, h(a), X, \downarrow) \mid X, Y \in K, XY \in P_1, a = g(X)\} \\ & \cup \{(\acute{X}, \grave{X}, h(a), X, \downarrow) \mid X \in P_1 \cap K, a = g(X)\} \\ & \cup \{(\grave{X}, \grave{Y}, h(a), X, \uparrow) \mid X, Y \in K, XY \in P_2, a = g(X)\} \\ & \cup \{(\grave{X}, \grave{Y}, h(a), X, \uparrow) \mid X, Y \in K, X \in P_2, a = g(X)\} \\ & \cup \{(\grave{X}, \$, h(a), X, \uparrow) \mid X \in P_2 \cap K, a = g(X)\}, \\ I = {} & \{\acute{X} \mid X \in P_0 \cap K\} \cup \{\$ \mid \epsilon \in W\}.\end{aligned}$$

Then it can be verified that $\mathscr{P}(M, h) = W$. ::

7.2.2. THEOREM. *Given $P$-tracer $M$, one can find an $F$-regular set $W$ and a projection $h$ such that $W = \mathscr{P}(M, h)$.*

*Proof.* For $P$-tracer $M = (K, B, C, \delta, I, F)$, define a new alphabet

$$H = \{(X, a, c, b, Y) \mid X, Y \in K, a, b \in B, c \in C\},$$

and a projection $h$: $(\hat{H})^* \to B^*$ by setting $h(\acute{Z}) = a$ and $h(\grave{Z}) = b$ for each $Z = (X, a, c, b, Y) \in H$. Let

$$\begin{aligned}I' = {} & \{\acute{Z} \mid Z = (X, a, c, b, Y) \in H, X \in I\}, \\ F' = {} & \{\grave{Z} \mid Z = (X, a, c, b, Y) \in H, X \in F\}, \\ P' = {} & \{\acute{Z}_1\acute{Z}_2 \mid Z_i = (X_i, a_i, c_i, b_i, Y_i) \in H \ (i = 1, 2), \\ & \quad (X_1, X_2, a_1, c_1, \downarrow) \in \delta\} \\ & \cup \{\acute{Z}\grave{Z} \mid Z = (X, a, c, b, Y) \in H, (X, Y, a, c, \downarrow) \in \delta\} \\ & \cup \{\grave{Z}_1\acute{Z}_2 \mid Z_i = (X_i, a_i, c_i, b_i, Y_i) \in H \ (i = 1, 2), \\ & \quad (Y_1, X_2, b_1, c_1, \uparrow) \in \delta\} \\ & \cup \{\grave{Z}_1\grave{Z}_2 \mid Z_i = (X_i, a_i, c_i, b_i, Y_i) \in H \ (i = 1, 2), \\ & \quad (Y_1, Y_2, b_1, c_1, \uparrow) \in \delta\}\end{aligned}$$

and define $S$-local set $S = \{s \in (\hat{H})^* \mid s\grave{Z} \in I'(\hat{H})^* - (\hat{H})^* (\hat{H}\hat{H} - P')(\hat{H})^*, \grave{Z} \in F'\}$. From Theorem 5.3.1 we know that there is an $F$-regular set $W$ such that $\mathrm{Trace}(W) = S \cap \mathscr{N}_{\hat{H}}$. Then from the construction, we can see that if $w \in W$ then the string $h(\mathrm{Trace}(w))$ in $B^*$ is accepted by pushdown automaton $M$ (that is, $w \in \mathscr{P}(M, h)$) and conversely. Thus $\mathrm{Trace}(W) = \mathscr{P}(M, h)$ holds. ::

7.3. *An Interpretation of Pushdown Tracers*

As for an interpretation of pushdown tracers concerned with programming languages, one may think of syntax-directed compilers of CF languages.

Suppose $G$ is a CF grammar with vocabulary $A$ which involves no $\epsilon$-rules (i.e., the right-hand sides of rules are not null string) and the CF language $L(G)$ generated by $G$ represents a (part of) programming language. Inasmuch as the set $T$ of derivation trees of $G$ is $F$-regular, one can construct a pushdown tracer $M$ which traces precisely the trees in $T$ (with respect to given mapping $h$). Provided that certain semantics are assigned by mapping $h$ to the nodes of derivation trees, by operating the tracer $M$ as an acceptor and generator (that is, as a translator), one can make it functioning as follows; while reading the input tape, $M$ accepts string $s$ if it is in $L(G)$, gives a derivation tree $t$ to $s$, and prints out the whole semantics associated with tree $t$. The order in which the semantics of each node appears in the output list is then of particular interest—one might think it models the build of object programs.

In more practical terms, in order to achieve the goal one has to provide the range of mapping $h$: $\hat{A} \to B$ (that is, the "input alphabet" of tracer $M$) with appropriate read and write commands. First of all, to let it work as an acceptor of $L(G)$, one has to include read command "read $a$" in $h(\acute{a})$ (or in $h(\grave{a})$) when and only when $a \in A$ is a terminal symbol of $G$. Secondly, to emit an "object program" $M$ should have write commands in $B$ that print out operation codes, macroinstructions, etc. (the components, of which the "object program" is made). Now recollecting the way pushdown tracers traverse forests, it should be noted that each node of the derivation tree is visited by $M$ exactly twice; once before $M$ goes down to the subordinate structure of the node, and once again right after it comes back from the substructure. The write command specified in $h(\acute{a})$ then takes places at the first visit of node $a$, whereas the one in $h(\grave{a})$ occurs at the second visit. Thus in the output list produced by $M$ the part corresponding to the subtree dominated by node $a$ is sandwiched between $h(\acute{a})$ and $h(\grave{a})$. In some cases, as the write command in $h(\acute{a})$ one might want to output a transfer vector to the procedure corresponding to the substructure, as well as storage allocation, loading of data from auxiliary memory, etc.; and by $h(\grave{a})$ operation codes to deal with returning routine, unloading data, etc., may be printed. Needless to say, if $h$ is designed so that the symbols $\acute{a}$ and $\grave{a}$ themselves are printed out by $h(\acute{a})$ and $h(\grave{a})$, respectively (maybe as the comments or the labels), then one can see an explicit picture of the derivation tree $t$ (in the form of Trace($t$)).

In short, the feature of our device $M$ is; while accepting a string, $M$ prints out the semantics associated with its syntactic structure in a distinctive order. The output list then may be considered as a model of object program corresponding to the input source program.

## ACKNOWLEDGMENTS

I would like to thank my dissertation supervisor Professor Joshi at University of Pennsylvania for his valuable advice and encouragement. Thanks are also due to Professor Oshiba at Shizuoka University and the anonymous reviewer of the journal for their critical reading and helpful comments.

RECEIVED: April 19, 1973; REVISED: May 1, 1974

## REFERENCES

BRAINERD, W. S. (1968), Minimization of tree automata, *Inform. Contr.* **13**, 484–491.

BRAINERD, W. S. (1969), Tree generating regular systems, *Inform. Contr.* **14**, 217–231.

BÜCHI, J. R. (1964), Regular canonical systems, *Archiv. Math. Logik Grundlagenforsch.* **6**, 91–111.

CHOMSKY, N. AND SCHÜTZENBERGER, M. P. (1963), The algebraic theory of context-free languages, *in* "Computer Programming and Formal Systems" (P. Braffort and D. Hirschburg, Eds.), North Holland Co., Amsterdam.

ELGOT, C. C. AND MEZEI, J. (1965), On relations defined by generalized finite automata, *IBM J. Res. Develop.* **9**, 47–68.

GINSBURG, S. AND HARRISON, M. (1967), Bracketed context-free languages, *J. Comput. System Sci.* **1**, 1–23.

GORN, S. (1966), "Explicit Definitions and Linguistic Dominoes," Proceedings of a conference at Univ. of Western Ontario (J. Hart and T. Takasu, Eds.).

HOPCROFT, J. E. AND ULLMAN, J. D. (1969), "Formal Languages and their Relation to Automata," Addison Wesley, Reading, MA.

JOSHI, A. K., KOSARAJU, S. R., AND YAMADA, H. M. (1972), String adjunct grammars. I and II, *Inform. Contr.* **21**, 93–116, 235–260.

KNUTH, D. E. (1967), A characterization of parenthesis languages, *Inform. Contr.* **11**, 209–289.

KNUTH, D. E. (1968), "The Art of Computer Programming," Addison Wesley, Reading, MA.

MCNAUGHTON, R. (1967), Parenthesis grammars, *J. Assoc. Comput. Mach.* **14**, 490–500.

MCNAUGHTON, R. AND PAPERT, S. (1971), "Counter-Free Automata," MIT Press, Boston, MA.

MEDVEDEV, Y. T. (1964), On the class of events representable in a finite automaton, *in* "Sequential Machines—Selected Papers" (E. F. Moore, Ed.), Addison Wesley, Reading, MA.

MYHILL, J. (1957), Finite automata and the representation of events, *Wright Air Develop. Command Tech. Rept.* **57-624**.

NERODE, A. (1958), Linear automaton transformations, *Proc. Amer. Math. Soc.* **9**, 541–544.

PAIR, C. AND QUERE, A. (1968), Définition et étude des bilangages réguliers, *Inform. Contr.* **13**, 565–593.

TAKAHASHI, M. (1972a), "Primitive Transformations of Regular Sets and Recognizable Sets," Proceeding of IRIA Symposium on Theory of Automata, Languages, and Programming.

TAKAHASHI, M. (1972b), "Regular Sets of Strings, Trees and $W$-Structures," Ph.D. dissertation, Univ. of Pennsylvania.

THATCHER, J. W. AND WRIGHT, J. B. (1968), Generalized finite automata theory with an application to a decision problem of second-order logic, *Math. Systems Theory* **2**, 57–81.

THATCHER, J. W. (1970), Generalized$^2$ sequential machines, *J. Comput. System Sci.* **4**, 339–367.