JOURNAL OF COMPUTER AND SYSTEM SCIENCES 4, 339–367 (1970)

Generalized² Sequential Machine Maps*

JAMES W. THATCHER

IBM Watson Research Center, Yorktown Heights, New York 10598

Received May 19, 1969

The introduction of syntax directed translations and transformations into formal language theory presents a very interesting area with considerable promise of application to questions of syntax and semantics of programming languages. The concept of generalized sequential machine (gsm) mapping (already of importance in language theory) is developed here in its natural extension to trees (or expressions). That generalized concept of gsm mapping encompasses most of the previously defined concepts relating to translations and transformations.

1. INTRODUCTION

The objective of this work is to expand the study of generalized finite automata theory by introducing generalized finite state mappings (finite state mappings of trees). In the process, we hope to provide an algebraic framework in which to study formalization of transformations (in the sense that linguists use the term) and translations of natural and artificial languages.

The concepts, "syntax directed translation" (Aho and Ullman [2]; Irons [19]; Lewis and Stearns [20]; and Petrone [23]) and "transformation" (Culik [12]; and Rounds [26]) can be formulated quite neatly within this framework.

One of the most important definitions given here, that of nondeterministic finite state transformation, is very closely related to one introduced by Rounds [26] (cf. Section 10).

Sections 2 and 3 present basic concepts of generalized finite automata theory and its relationship to context-free languages. We provide some general notation for transformations in Section 4 and in the following four sections, two basic kinds of transformations will be introduced and certain fundamental properties proved. There is considerable detail here, but it is necessary in order to have a sound basis on which to develop more interesting applications of transformations to language theory. In Section 9, these definitions will be related to others in the literature and several areas for further investigation will be indicated.

* A version of this paper appeared in the Proceedings of the ACM Symposium on Theory of Computing, May 1969 [29].

2. Generalized Finite Automata Theory Definitions

A *ranked alphabet* is a pair $\langle \Sigma, r \rangle$ when Σ is a finite set of *symbols* and $r \subseteq \Sigma \times \omega$ is a finite relation called the *ranking relation*.[1] If $r(\sigma, n)$, then we say σ has *rank n* and Σ_n denotes the set of symbols of rank n. Unless it is necessary to do otherwise, we will just write Σ instead of $\langle \Sigma, r \rangle$, some fixed ranking relation being understood.

Let Z be a set disjoint from Σ. The set $T_{\Sigma,Z}$ of *Σ-expressions with variables Z* is defined by the following fundamental inductive definition. $T_{\Sigma,Z}$ is a subset of $(\Sigma \cup Z \cup \{(,)\})^*$.[2]

2.1. (0) If $\delta \in Z \cup \Sigma_0$, then $\delta \in T_{\Sigma,Z}$;

(1) If $n > 0$, $\sigma \in \Sigma_n$ and $t_1, \ldots, t_n \in T_{\Sigma,Z}$, then $\sigma(t_1 \cdots t_n) \in T_{\Sigma,Z}$.

We will be interested in several cases for Z; for example, $T_{\Sigma,\emptyset}$ (written T_Σ and called the set of *constant* Σ-expressions), $T_{\Sigma,X}$ and $T_{\Sigma,X\times S}$ where $X = \{x_1, x_2, \ldots\}$ and S is a finite set. There will also be reason to consider the initial segments of X and we will write $X_n = \{x_1, \ldots, x_n\}$.

Example 2.1. Let $\Sigma = \Sigma_0 = \Sigma_2 = \{a, b\}$. Examples of Σ-expressions are a, b, $a(b(aa)\,b(ba))$ and Σ-expressions with variables $X = \{x_1, x_2, \ldots\}$ include x_3, $a(b(x_1x_3)\,a(x_2b(ab))$, $b(a(x_1x_2)\,b(x_1a))$. As is well-known, such expressions can be represented as labeled ordered trees.[3] In this example, all branching is binary and the labeling arbitrary. (See Figures 1 and 2.)

If, instead of $\Sigma_0 = \{a, b\}$ we had taken $\Sigma_0 = \{a\}$, then the labeling would no longer be arbitrary. No b's could label the leaves of the tree.

Example 2.2. Let Σ be a finite alphabet and consider $\Omega = \Sigma \cup \{\Lambda\}$ with ranking determined by $\Omega_1 = \Sigma$, $\Omega_0 = \{\Lambda\}$. Then T_Ω is identified with Σ^* since any Ω-expression is of the form $\sigma_1(\sigma_2(\cdots(\sigma_k(\Lambda))\cdots))$ for $k \geqslant 0$ and this corresponds to the string $\sigma_1 \cdots \sigma_k \in \Sigma^*$.[4] Expressions in $T_{\Omega,X}$ are of the form $\sigma_1 \cdots \sigma_k(x)$ when $x \in X$ or $x = \Lambda$.

We will say that the ranked alphabet $\langle \Sigma, r \rangle$ is *monadic* if $\Sigma_0 = \{\Lambda\}$ and $\Sigma_n = \emptyset$

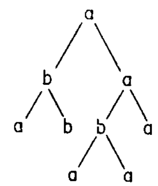
[1] ω is the set of nonnegative integers. Note that in previous work, e.g. [4, 6, 7, 13, 14, 28], the alphabet ranking was assumed to be functional, each symbol having fixed rank. That restriction is relaxed here reflecting common usage both in context-free language theory (derivation trees) and in programming languages where symbols often have several possible ranks. Much of the theory goes through with arbitrary $r \subseteq \Sigma \times \omega$ and all of it does with certain regularity restrictions on infinite r (see [27] where $r = \Sigma \times \omega$).

[2] It is assumed that the left and right parentheses are in neither Σ nor Z. W^* is the free monoid generated by W.

[3] A simple definition of labeled ordered tree and the correspondence with expressions is given in [27].

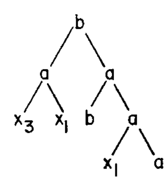
[4] W^* is the free monoid generated by W, λ is the identity and $W^+ = W - \{\lambda\}$.

for $n \geqslant 2$. In what follows, the specialization to the monadic case gives rise to conventional concepts in finite automata theory. Note that under this specialization, Λ plays the role of the empty string—but it is not. Here and in the sequel Λ is a zero-ary symbol; λ is the empty string.



$t_1 = a(b(ab)a(b(aa)a))$

FIGURE 1



$t_2 = b(a(x_3x_1)a(ba(x_1a)))$

FIGURE 2

The set var $(t) \subseteq Z$ is the set of variables occurring in $t \in T_{\Sigma,Z}$. The definition is inductive:

2.2. (0) $\operatorname{var}(\delta) = \begin{cases} \{\delta\} & \text{if } \delta \in Z, \\ \emptyset & \text{otherwise;} \end{cases}$

(1) $\operatorname{var}(\sigma(t_1 \cdots t_n)) = \bigcup_i \operatorname{var}(t_i)$.[5]

For $W \subseteq \Sigma_0 \cup Z$, the *frontier function*, fr_W, is a map from $T_{\Sigma,Z}$ into W^*. $fr_W(t)$ is the string of symbols from W in the order in which they appear as symbols of rank 0 in t.

2.3. (0) $fr_W(\delta) = \begin{cases} \delta & \text{if } \delta \in W, \\ \lambda & \text{otherwise;} \end{cases}$

(1) $fr_W(\sigma(t_1 \cdots t_n)) = fr_W(t_1) \cdots fr_W(t_n)$.

When $W = \Sigma_0 \cup Z$ we will simply write fr for fr_W.

[5] In such definitions the hypotheses "For $\delta \in \Sigma_0 \cup Z \cdots$" in (0) and "For $n > 0$, $\sigma \in \Sigma_n$ and $t_1, \ldots, t_n \in T_{\Sigma,Z} \cdots$" in (1) will be assumed.

The third in this sequence of simple inductive definitions is the map top: $T_{\Sigma,Z} \to \Sigma \cdot \text{tap}\,(t)$ is simply the outermost symbol of t.

2.4. (0) $\text{tap}(\delta) = \delta$;

(1) $\text{top}(\sigma(t_1 \cdots t_n)) = \sigma$.

EXAMPLE 2.3. Let t_i be the example of Fig. 1. $\text{Var}(t_1) = \emptyset$, $\text{var}(t_2) = \{x_1, x_3\}$, $fr(t_1) = abaaa$, $fr_X(t_2) = x_3x_1x_1$, $\text{top}(t_2) = b$. Also, from Example 2.2, $fr(t) = A$ or $fr(t) = x_i \in X$.

A *Σ-algebra* is a pair $\mathscr{A} = \langle A, \alpha \rangle$ where A is a set called the *carrier* of $\mathscr{A}$ and α assigns to each $\sigma \in \Sigma$, a function $\alpha_\sigma : A_\sigma \to A$ where $A_\sigma = \bigcup_{r(\sigma,n)} A^n$.[6] For $\sigma \in \Sigma_0$, we will write α_σ for $\alpha_\sigma(\lambda)$.

The *totally free Σ-algebra generated by Z* is $\mathscr{T}_{\Sigma,Z} = \langle T_{\Sigma,Z}, \iota \rangle$ where

2.5. (0) $\iota_\sigma = \sigma\ (\sigma \in \Sigma_0)$;

(1) $\iota_\sigma(t_1 \cdots t_n) = \sigma(t_1 \cdots t_n)$.

The totally free Σ-algebra generated by $\emptyset$ is called the *generic* Σ-algebra, denoted $\mathscr{T}_\Sigma$.

Let $\mathscr{A}$ be a Σ-algebra and let $\mathfrak{A} : Z \to A$ be an *assignment* of values in A to the variables in Z. The assignment $\mathfrak{A}$ determines a unique homomorphism from $\mathscr{T}_{\Sigma,Z}$ into $\mathscr{A}$:

2.6. (0) $h_{\mathfrak{A}}(\delta) = \begin{cases} \mathfrak{A}(\delta) & \text{if } \delta \in Z, \\ \alpha_\delta & \text{if } \delta \in \Sigma_0; \end{cases}$

(1) $h_{\mathfrak{A}}(\sigma(t_1 \cdots t_n)) = \alpha_\sigma(h_{\mathfrak{A}}(t_1) \cdots h_{\mathfrak{A}}(t_n))$.

It should be clear that if $\mathfrak{A}_1 = \mathfrak{A}_2$ on var(t), then $h_{\mathfrak{A}_1}(t) = h_{\mathfrak{A}_2}(t)$; in particular, if $t \in T_\Sigma$, then the image of t under $h_{\mathfrak{A}}$ is independent of $\mathfrak{A}$. We will write $h_{\mathscr{A}}$ to denote this unique homomorphism of $\mathscr{T}_\Sigma$ into $\mathscr{A}$.

A finite Σ-automaton, $\mathfrak{A}$, is a pair $\langle \mathscr{A}, \bar{A} \rangle$ where $\mathscr{A} = \langle A, \alpha \rangle$ is a finite Σ-algebra (the *structure* of $\mathfrak{A}$) and $\bar{A} \subseteq A$ is the set of *final states* of $\mathfrak{A}$.[7] A constant Σ-expression t is *recognized* by $\mathfrak{A}$ if $h_{\mathscr{A}}(t) \in \bar{A}$. $T(\mathfrak{A})$ is the set of Σ-expressions recognized by $\mathfrak{A}$:

2.7. $T(\mathfrak{A}) = h_{\mathscr{A}}^{-1}(\bar{A}) = \{t \mid h_{\mathscr{A}}(t) \in \bar{A}\}$.

$U \subseteq T_\Sigma$ is *recognizable* if there exists a finite Σ-automaton $\mathfrak{A}$ such that $U = T(\mathfrak{A})$.

EXAMPLE 2.4. Using again the ranked alphabet of Example 2.1, any Σ-algebra can be viewed as a set A together with two diadic operations $\alpha_\sigma : A^2 \to A$ ($\sigma \in \Sigma$) and two

[6] For $U, V \subseteq A^*$, $U \cdot V = \{xy \mid x \in U \text{ and } y \in V\}$; $U^0 = \{\lambda\}$ and $U^{n+1} = U^n \cdot U$. Here, we look at A_σ as a subset of A^*.

[7] If we were concerned with infinite ranking relations (see footnote 1), it would be necessary to require that each $\alpha_\sigma^{-1}(a)$ be a regular subset of A^* in order to insure that the generalized theory matches the conventional case with respect to decision problems.

zero-ary operations or constants, $\alpha_\sigma \in A$ ($\sigma \in \Sigma$). With $A = \{0, 1\}$, $\alpha_a = 1$, $\alpha_b = 0$, $\alpha_a(\delta_1\delta_2) = (\delta_1 + \delta_2 + 1) \bmod 2$ and $\alpha_b(\delta_1\delta_2) = (\delta_1 + \delta_2) \bmod 2$, $h_{\mathscr{A}}^{-1}(1)$ is the set of Σ-expressions having an odd number of occurrences of a.

For the monadic species (cf. Example 2.2), an automaton is the usual kind of object: a set A of states, a monadic transition function $\alpha_\sigma : A \to A$ for each $\sigma \in \Sigma$, an initial state $\alpha_\Lambda \in A$, and a set $\bar{A}$ of final states.

Up to this point, we have the definitions that provide the basis for generalized finite automata theory. As indicated by various papers [4, 6, 7, 13, 14, 24, 27, 28] the results of the conventional theory carry over for the generalization.[8]

3. Local and Context-Free Sets

The local sets (to be defined below) form a proper subclass of the recognizable sets. They are of particular interest from two points of view. First, the concept generalizes the idea of "complete state sequences," R-sequences [15] or Γ-sequences [9], an idea that is fundamental and pervasive in finite automata theory. The local sets have the important property that under projection, they yield exactly the recognizable sets. Second, the local sets correspond to sets of derivation trees of context-free grammars.[9]

Let $\langle \Sigma, r \rangle$ be a ranked alphabet and define $\bar{r} = \{\langle \sigma, w \rangle \mid w \in \Sigma^* \text{ and } r(\sigma, lg(w))\}$.[10] A structure $G = \langle R, S_0 \rangle$, where $R \subseteq \bar{r}$ and $S_0 \subseteq \Sigma$ will be called a *context-free grammar* over Σ. R is called the set of relations or *productions* of G and S_0 is its set of *initial symbols*. The sets D_σ^G ($\sigma \in \Sigma$) of σ-derivations are defined by the following simultaneous inductive definition.

3.1. *If* $\langle \sigma, \sigma_1 \cdots \sigma_n \rangle \in R$ and $t_i \in D_{\sigma_i}^G$ $(i = 1, \ldots, n)$, then $\sigma(t_1 \cdots t_n) \in D_\sigma^G$.[11]

The subset of T_Σ generated by G is $T(G) = \bigcup_{\sigma \in S_0} D_\sigma^G$ and the language generated by G is $L(G) = fr\,T(G)$.

[8] Except in [27], the ranking relation was assumed to be functional and there, r was all of $\Sigma \times \omega$. Considering arbitrary finite (or regular) $r \subseteq \Sigma \times \omega$ is convenient and offers no significant deviation in the theoretical development. The situation of a nonfunctional r reduces to the functional case when r itself is considered to be the set of function symbols with *rank*: $r \to \omega$ defined by *rank* $\langle \sigma, n \rangle = n$.

[9] The correspondence is the usual one, representing a labeled ordered tree as a parenthesized string (see [27]).

[10] $lg(w)$ is the length of w as a string in Σ^*.

[11] The case $n = 0$ is included in this definition and yields the basis. In order to make sense out of this case, we adopt the convention that $\sigma(\lambda)$ is a notation for σ. No confusion should arise from this convention since even though both σ and $\sigma(\lambda) = \sigma(\ )$ are in $(\Sigma \cup \{(\ ,\ )\})^*$ it is σ that is a Σ-expression, i.e., $\sigma(\ ) \notin T_\Sigma$. The identification of $\sigma(\ )$ and σ makes this and some subsequent definitions simpler.

A subset $V \subseteq T_\Sigma$ will be called *local* if there exists a context-free grammar G over Σ such that $V = T(G)$. A language $W \subseteq \Sigma^*$ is *context-free* if $W = L(G)$ for some context-free grammar G.[12]

EXAMPLE 3.1. Given a context-free grammar according to the usual definition, $G' = \langle N, T, P, s_0\rangle$ where $P \subseteq N \times (N \cup T)^+$. Take

$$\Sigma = N \cup T, \qquad r = \{\langle\sigma, lg(w)\rangle \mid \langle\sigma, w\rangle \in P\} \cup T \times \{0\},$$
$$R = P \cup \{\langle\sigma, \lambda\rangle \mid \sigma \in T\} \qquad \text{and} \qquad S_0 = \{s_0\}.$$

Then with $G = \langle R, S_0\rangle$, the sets D_σ^G correspond to the derivation trees from σ as described in Ginsburg [18], and $L(G)$ is exactly $L(G')$ as usually described.

Let $\langle\Sigma, r\rangle$ and $\langle\Omega, s\rangle$ be two ranked alphabets and let $\pi : \Sigma \to \Omega$ be a map with $\pi\Sigma_n \subseteq \Omega_n$. π is extended to a map $\bar{\pi}$ from T_Σ into T_Ω by

3.2. $\bar{\pi}\sigma(t_1 \cdots t_n) = \pi(\sigma)\,(\bar{\pi}(t_1) \cdots \bar{\pi}(t_n))$.

A *projection* is any map obtained in this way.

The important relationship between local and recognizable sets is given in Theorem 3.6 below. Since the results (Lemmas 3.3–3.5) are contained in [27] (with only a slight variation in formulation, cf. footnote 1) and since the proofs follow familiar arguments from finite automata theory, they are abbreviated here.

LEMMA 3.3. *Local subsets of T_Σ are recognizable.*

Proof. The partition $\{D_{\sigma_1}^G, \ldots, D_{\sigma_m}^G, T_\Sigma - \cup D_\sigma^G\}$ is the partition of a congruence relation $\sim$ on $\mathscr{T}_\Sigma$ of finite index. The quotient $\mathscr{A} = \mathscr{T}_\Sigma/\sim$ is a finite Σ-algebra and choosing $\bar{A} = \{\tilde{D}_\sigma^G \mid \sigma \in S_0\}$, the automaton $\langle\mathscr{A}, \bar{A}\rangle$ recognizes $T(G)$.

LEMMA 3.4. *The recognizable sets are closed under projection.*

Proof. This is the usual subset construction. Given $\mathfrak{A} = \langle\mathscr{A}, \bar{A}\rangle$ recognizing $V \subseteq T_\Sigma$ and a projection π of T_Σ into T_Ω, define the Ω-algebra $\mathfrak{A}' = \langle\mathscr{A}', \bar{A}'\rangle$ where

$$A' = pA,^{13} \qquad \alpha_\omega'(u_1 \cdots u_n) = \{\alpha_\sigma(a_1 \cdots a_n) \mid a_i \in u_i \text{ and } \pi(\sigma) = \omega\},$$

and

$$\bar{A}' = \{u \mid u \cap \bar{A} \neq \emptyset\}.$$

[12] Strictly speaking, our use of the terminology "context-free grammar" is incorrect since, to start with, the standard definition requires a grammar to be presented as a quadruple. Aside from this technicality, we diverge slightly from Chomsky's [10] definition in that we allow an initial set and, in effect, the sets of terminals and nonterminals need not be disjoint.

[13] pA is the power set of A.

By induction on t it is easy to prove

3.4.1. $h_{\mathscr{A}'}(t) = \{h_{\mathscr{A}}(t') \mid \bar{\pi}t' = t\}$,

so that $T(\mathfrak{A}') = \bar{\pi}T(\mathfrak{A})$; that is, $\bar{\pi}V$ is recognized by $\mathfrak{A}^2$.

LEMMA 3.5. *Every recognizable subset of T_Σ is a projection of a local subset of T_Ω for some ranked alphabet Ω.*

Proof. This is the usual complete-state argument. Let $\mathfrak{A} = \langle \mathscr{A}, \bar{A} \rangle$ recognize $V \subseteq T_\Sigma$. Consider the ranked alphabet $\langle \Sigma', r' \rangle$ where $\Sigma' = A \times \Sigma$ and $r'(a, \sigma, n) \leftrightarrow r(\sigma, n)$. The context-free grammar is $G = \langle R, S_0 \rangle$ where

3.5.1. $R = \{\langle\langle a, \sigma\rangle, \langle a_1, \sigma_1\rangle \cdots \langle a_n, \sigma_n\rangle\rangle \mid \sigma \in \Sigma_n \text{ and } \alpha_\sigma(a_1 \cdots a_n) = a\}$.

The projection π is the natural one from $A \times \Sigma \rightarrow \Sigma$. By induction, one proves

3.5.2. $t \in \bigcup_\sigma D^G_{\langle a,\sigma\rangle} \leftrightarrow h_{\mathscr{A}}\bar{\pi}t = a$.

Then with $S_0 = \{\langle a, \sigma\rangle \mid a \in \bar{A}\}$, $\bar{\pi}T(G) = T(\mathfrak{A})$.

From Lemmas 3.3–3.5, we now have:

THEOREM 3.6. *A set of expressions is recognizable if and only if it is the projection of a local set.*

Any projection $\bar{\pi} : T_\Sigma \rightarrow T_\Omega$ induces a projection $\bar{\pi}' : \Sigma_0{}^* \rightarrow \Omega_0{}^*$ such that $fr\bar{\pi}V = \bar{\pi}' frV$. Since the context-free languages are closed under projections [5], from Theorem 3.6 we obtain

LEMMA 3.7. *If $V \subseteq T_\Sigma$ is recognizable, then $frV \subseteq \Sigma_0{}^*$ is context-free.*

From Lemmas 3.3, 3.7, and the definition of context-free sets, it follows that the context-free sets are exactly the frontier sets obtained from recognizable sets. Let $U \subseteq \Sigma^*$ be a context-free set. Then, in general, the context-free grammar G with $U = L(G)$ will be over a larger alphabet Ω where $\Sigma \subseteq \Omega_0$. However, as was pointed out by M. O. Rabin (personal communication), when recognizable sets are used, an increased alphabet is not necessary.

THEOREM 3.8. *$U \subseteq \Sigma^*$ is context-free if and only if $U = frV$ for some recognizable subset of $V \subseteq T_\Sigma$.*

Proof. Let G be a context-free grammar over $\langle \Omega, r\rangle$ with $L(G) = U$. We want to find a ranking r' on Σ and a projection $\bar{\pi} : T_\Omega \rightarrow T_\Sigma$ so that $fr\bar{\pi}T(G) = U$. Then by Lemmas 3.3 and 3.4, $\bar{\pi}T(G)$ is recognizable. Let σ_0 be a distinguished element of Σ. With $r'(\sigma_0, n)$ for all $n > 0$ such that $r(\omega, n)$ holds for some $\omega \in \Omega$. Also, let $r'(\sigma, 0)$

for all $\sigma \in \Sigma$. Then define $\pi(\omega) = \sigma_0$ for $\omega \in \Omega - \Sigma$ and $\pi(\sigma) = \sigma$ for $\sigma \in \Sigma$. This ranking r' and projection π satisfy the conditions described above.

4. Transformations and Translations

Generally, the term "transformation" will mean any map from T_Σ into T_Ω where $\Sigma = \langle \Sigma, r \rangle$ and $\Omega = \langle \Omega, s \rangle$ are ranked alphabets. We will define several types of transformations. In formulating these definitions, there were three principal considerations or objectives:

(1) It was intended that the transformations be natural in that they would fit the algebraic framework within which we are working.

(2) We should be able to generalize the conventional concept of finite state mapping (or generalized sequential machine mapping [18]) to the case for trees.

(3) It was hoped that the end result would be a unified approach taking into account various formulations of "transformation," "transduction," and "translation" which have appeared in the literature.

The different classes of transformations will be indicated by modifiers; for example, finite state transformations, nondeterministic finite state transformations, etc. For such a modifier X, a subset $W \subseteq T_\Omega$ will be called an *X-surface set* if it is the image of a recognizable subset of T_Σ under an X-transformation from T_Σ into T_Ω.[14] A subset $U \subseteq \Omega_0{}^*$ will be called an *X-transformational language* iff there exists an X-transformation $\bar{\tau} : T_\Sigma \to T_\Omega$ and a recognizable subset $V \subseteq T_\Sigma$ such that $U = fr\bar{\tau}V$.

Each X-transformation $\bar{\tau} : T_\Sigma \to T_\Omega$ and recognizable set $V \subseteq T_\Sigma$ determine a relation $\rho_{\bar{\tau}} \subseteq \Sigma_0{}^+ \times \Omega_0{}^+$ by

4.1. $\rho_{\bar{\tau}} = \{\langle fr(t), fr(\bar{\tau}t)\rangle \mid t \in V\}$.

$\rho_{\bar{\tau}}$ is called an *X-translation* (of frV). We will say that a recognizable set V is *unambiguous* if $fr \mid V$ is 1:1. Although $\rho_{\bar{\tau}}$ need not generally be a function, it will be if V is unambiguous.

5. Substitution and Expressions

The very simple and familiar concept of substitution (of expressions for variables) is key to our approach to transformations and translations. If there is simplification

[14] The terminology "surface set," used by Rounds [26] is motivated by the linguists' reference to surface structures as the results of applying transformations to what, in effect, are recognizable sets (c.f. Chomsky, [11]).

in our approach over others, then the formal use of substitution can be pinpointed as the primary reason for that simplification.

The definition of substitution comes as a special case of the homomorphism defined in 2.6. In particular, if $\mathscr{A}$ is taken to be $\mathscr{T}_{\Sigma,Z}$ and η is an assignment from Z into $T_{\Sigma,Z}$, then $h_\eta(t)$ is the result of simultaneously substituting $\eta(z)$ for each z in t.

We will write $h_\eta(t)$ as $t \cdot \eta$ and extend this operation to assignments (componentwise) to obtain a binary operation $\zeta \cdot \eta$ on assignments. We will prove that the resulting operation is associative, a property that is critical for later results.

To begin the discussion of substitution, we introduce an alternative way of working with $T_{\Sigma,Z}$. Let $\langle \Sigma, r\rangle$ be a ranked alphabet and define $\Sigma' = \{\sigma(x_1 \cdots x_n) \mid r(\sigma, n)\}$.

Thus Σ' is a subset of $T_{\Sigma,X}$ (recall that $X = \{x_1, x_2, \ldots\}$).

We will write σ_n for $\sigma(x_1 \cdots x_n)$ (cf. footnote 8). Now, for the simplest case of substitution, let Z be arbitrary and let η be an assignment from X into $T_{\Sigma,Z}$ and define

5.1. $\sigma_n \cdot \eta = \sigma(\eta(x_1) \cdots \eta(x_n)) = h_\eta(\sigma_n)$.

Again (cf. footnote 11), this definition includes the case $n = 0$ where $\sigma_0 \cdot \eta = \sigma(\lambda) = \sigma$.

Using the notation of 5.1 we can formulate an induction principle which will simplify some of the proofs in the sequel.

5.2. If (0) $Z \subseteq U$ and

(1) for all $\eta : X \to T_{\Sigma,Z}$ and $\sigma_n \in \Sigma'$, $\forall x[\eta(x) \in U]$ implies $\sigma_n \cdot \eta \in U$,

then $U \supseteq T_{\Sigma,Z}$.

This induction principle justifies definitions of the form 5.3 and proofs (e.g., 5.4) which will generally have the form of proving a property $P(z)$ for all $z \in Z$, proving an extended property $P'(\eta)$ using the inductive hypothesis (indicated by IH), $\forall z P(\eta(z))$, and finally proving $P(\sigma_n \cdot \eta)$ from $P'(\eta)$. It should be clear that there is nothing new here; 5.2 is not much more than a notational variant of the inductive definition (2.1) of $T_{\Sigma,Z}$.

By 5.1 the substitution operation is defined on $\Sigma' \times T^X_{\Sigma,Z}$. ($T^X_{\Sigma,Z}$ is, of course, the set of all assignments (functions) from X into $T_{\Sigma,Z}$.) Using the induction principle 5.2, we extend this operation to $T_{\Sigma,Z} \times T^Z_{\Sigma,Z'}$ and (componentwise) to $T^{Z''}_{\Sigma,Z} \times T^Z_{\Sigma,Z'}$ for arbitrary variable sets Z, Z' and Z''.

5.3. (0) For any $z \in Z$ and $\xi : Z \to T_{\Sigma,Z'}$,

$$z \cdot \xi = \xi(z).$$

(1) For any $\zeta : Z'' \to T_{\Sigma,Z}$, $\xi : Z \to T_{\Sigma,Z'}$ and $z \in Z''$,

$$(\zeta \cdot \xi)(z) = \zeta(z) \cdot \xi.$$

(2) For any $\sigma_n \in \Sigma'$, $\eta : X \to T_{\Sigma,Z}$, $\xi : Z \to T_{\Sigma \cdot Z'}$,

$$(\sigma_n \cdot \eta) \cdot \xi = \sigma_n \cdot (\eta \cdot \xi).$$

Let U be the subset of $T_{\Sigma,Z}$ for which $t \cdot \xi$ is defined for all $\xi \in T^Z_{\Sigma,Z'}$. $U \supseteq Z$ by (0). Assuming $\eta(x_i) \in U$ for each x_i, $\sigma_n \cdot \eta \in U$ because by (2) $(\sigma_n \cdot \eta) \cdot \zeta = \sigma_n \cdot (\eta \cdot \zeta)$. By 5.1, the rightside is $\sigma_n(\eta \cdot \xi(x_i) \cdots \eta \cdot \xi(x_n))$ and $\eta \cdot \zeta(x_i) = \eta(x_i) \cdot \xi$ by (2). The latter is defined by induction hypothesis. Thus by 5.2, $U = T_{\Sigma,Z}$ and (1) also gives the arbitrary componentwise extension to $T^{Z''}_{\Sigma,Z} \times T^Z_{\Sigma,Z'}$. That's how the induction principle works in definitions. This kind of presentation is far more detailed than necessary. Thus, paralleling the previous convention (see Definition 2.2 and footnote 9), the quantifiers in definitions and proofs will be omitted when they do not result in ambiguity. Thus we may write 5.3 using only the equations:

5.3′. (0) $z \cdot \xi = \xi(z)$;

(1) $(\zeta \cdot \xi)(z) = \zeta(z) \cdot \xi$;

(2) $(\sigma_n \cdot \eta) \cdot \xi = \sigma_n \cdot (\eta \cdot \xi)$;

and they are to be interpreted in the broadest sense of the universal quantifiers that makes them meaningful, subject to the conventions concerning the metamathematical variables. ξ, ζ, η, ψ range over assignments, σ ranges over Σ, z ranges over Z (arbitrary variable set), and x ranges over X.

EXAMPLE 5.1. Take $\Sigma = \Sigma_0 = \Sigma_2 = \{a, b\}$ and $Z = X$. The following table gives examples of substitution for assignments.

	x_1	x_2	x_3	x_i $(i > 3)$
η_1	a	$b(x_1x_1)$	$a(x_1x_3)$	x_i
η_2	b	$b(x_2x_1)$	x_2	x_i
$\eta_1 \cdot \eta_2$	a	$b(bb)$	$a(bx_2)$	x_i
$\eta_2 \cdot \eta_1$	b	$b(b(x_1x_1)a)$	$b(x_1x_1)$	x_i

Also,

$$a(x_1b(x_3a)) \cdot \eta_1 = a(ab(a(x_1x_3)\,a))$$

and

$$a(x_1b(x_3a)) \cdot \eta_2 = a(bb(x_2)).$$

When the set of variables is X, then a sequence of expressions, $\langle t_1 ,..., t_n\rangle$, can stand for the assignment $\eta : X \to T_{\Sigma,Z}$ when $\eta(x_i) = t_i$ for $1 \leqslant i \leqslant n$ and $\eta(x_j) = x_j$ for $j > n$. Then for this η, $t \cdot \langle t_1 \cdots t_n\rangle = t \cdot \eta$. In examples, it is sometimes convenient to use this sequence notation to denote the nonidentity part of an assignment η.

EXAMPLE 5.2. $b(x_2x_1) \cdot \langle a, b(x_1x_1), a(x_1x_3)\rangle = b(b(x_1x_1)\ a) = \eta_2 \cdot \eta_1(x_2)$ from Example 5.1.

From 5.3(2) the operation of substitution is associative $((x \cdot y) \cdot z) = x \cdot (y \cdot z))$ up to σ_n for x, $\eta : X \to T_{\Sigma,Z}$ for y and arbitrary $\xi : Z \to T_{\Sigma,Z'}$ for z. The following lemma extends this to arbitrary $t \in T_{\Sigma,Z}$ for x, $\zeta : Z \to T_{\Sigma,Z'}$ for y and $\xi : Z' \to T_{\Sigma,Z''}$ for z.

LEMMA 5.4. $(t \cdot \zeta) \cdot \xi = t \cdot (\zeta \cdot \xi)$.

Proof by induction on t.

(0) $(z \cdot \zeta) \cdot \xi = \zeta(z) \cdot \xi$ 5.3(0)

$= (\zeta \cdot \xi)(z)$ 5.3(1)

$= z \cdot (\zeta \cdot \xi)$ 5.3(0)

(1) Assuming 5.4 holds for $\eta(z)$ for all z, we obtain:

5.4.1. $(\eta \cdot \zeta) \cdot \xi = \eta \cdot (\zeta \cdot \xi)$.

Proof. $(\eta \cdot \zeta) \cdot \xi(z) = (\eta(z) \cdot \zeta) \cdot \xi$ 5.3(1)

$= \eta(z) \cdot (\zeta \cdot \xi)$ IH

$= \eta \cdot (\zeta \cdot \xi)(z)$ 5.3(1)

(2) $(\sigma_n \cdot \eta) \cdot (\zeta \cdot \xi) = \sigma_n \cdot (\eta \cdot (\zeta \cdot \xi))$ 5.3(2)

$= \sigma_n \cdot ((\eta \cdot \zeta) \cdot \xi)$ 5.4.1

$= ((\sigma_n \cdot \eta) \cdot \zeta) \cdot \xi$ 5.3(2)

The proof of Lemma 5.4 follows exactly the form described in the discussion of the induction principle. By induction, 5.4 is true for all $t \in T_{\Sigma,Z}$. Therefore, for any assignment $\eta : Z \to T_{\Sigma,Z}$, 5.4 is true for all $\eta(z)$ and by the way 5.4(1) was stated and proved, 5.4.1 is now true for arbitrary η, ζ, ξ (which make the expressions meaningful). We may, therefore, consider 5.4.1 to be a corollary of the proof of 5.4.

We have belabored the mechanics of this proof so that we won't have to for others. Many of the proofs to follow have exactly the same form.

By 5.4.1 we are dealing with a structure $\mathbf{T}_{\Sigma,Z} = \langle T^Z_{\Sigma,Z}, \cdot, 1_Z \rangle$ which is a monoid (1_Z is the identity function on Z) and which I will call a *pretheory*. The abstract definition of pretheory must here be left as a problem. It seems that the answer will give considerable insight into the algebraic properties of transformations and translations. The term "pretheory" is used here because these structures are very closely related to the "algebraic theories" of Eilenberg and Wright [14]. They are not algebraic theories, but corresponding to every object that we shall call a pretheory, it is possible to construct an algebraic theory which, in effect, serves the same purpose.

EXAMPLE 5.3. In the case where $Z = X \times S$, an algebraic theory is obtained by taking, as morphisms $[n] \xrightarrow{f} [m]$, all assignments f from $\{x_1, \dots, x_n\} \times S$ into

$T_{\Sigma,\{x_1,\ldots,x_m\}\times S}$ (see Eilenberg and Wright [14]). For $[1] \xrightarrow{f} [m]$ and $[m] \xrightarrow{g} [p]$, the composition gf is the result of simultaneously substituting $g(x_i, s)$ for each occurrence of $\langle x_i, s\rangle$ in f ($1 \leqslant i \leqslant m$, $s \in S$). With $Z = X$, the result of this construction is the free theory with base Σ'.

6. Finite State Transformations

In this section we will introduce the generalization for expressions (or trees) of (deterministic) finite state transformations. These functions from Σ-expressions to Ω-expressions (ranked alphabets Σ and Ω) generalize the conventional notion of generalized sequential machine mapping [18] from Σ^* into Ω^* (alphabets Σ and Ω) determined by $\langle S, s_0, \delta, \tau\rangle$ where S is a finite set of states, $s_0 \in S$ is the initial state, $\delta : \Sigma \times S \to S$ is the next state function, and $\tau : \Sigma \times S \to \Omega^*$ is the output function. The procedure is conventionally stated as follows. Given a state $s \in S$, an input symbol σ produces an output string $\tau(\sigma, s)$ and next state $\delta(\sigma, s)$. Thus the map $\bar{\tau} : \Sigma^* \times S \to \Omega^*$ is conventionally defined by induction:

6.1. (0) $\bar{\tau}(\lambda, s) = \lambda;$
$\bar{\delta}(\lambda, s) = s;$

(1) $\bar{\tau}(w\sigma, s) = \bar{\tau}(w, s)\, \tau(\sigma, \bar{\delta}(w, s));$
$\bar{\delta}(w\sigma, s) = \delta(\sigma, \bar{\delta}(w, s)).$

Then $\bar{\tau} : \Sigma^* \to \Omega^*$ is defined by $\bar{\tau}(w) = \bar{\tau}(w, s_0)$.

An equivalent definition can be obtained using induction on left successors.

6.2. (0) $\bar{\tau}(\lambda, s) = \lambda;$
(1) $\bar{\tau}(\sigma w, s) = \tau(\sigma, s)\, \bar{\tau}(w, \delta(\sigma, s)).$

This nonstandard formulation 6.2 is in fact simpler in that it does not require extension of the next state function to Σ^*.

As will be clear, the generalization is in the form of 6.2 and the fact that the extension of the next state function is not required seems to be important.

A *finite state transformation* (FST) from T_Σ into T_Ω is determined by a triple, $\langle S, s_0, \tau\rangle$ where S is a finite set of *states*, $s_0 \in S$ is the *initial state*, and

$$\tau : \Sigma' \times S \to T_{\Omega, X\times S}$$

is the *output function* subject to the condition:

6.3. $\tau(\sigma_n, s) \in T_{\Omega,\{x_1,\ldots,x_n\}\times S}$.

Thus, in effect, the FST $\langle S, s_0, \tau\rangle$ assigns to each triple $\langle \sigma, n, s\rangle$ of $\langle$symbol, rank, state$\rangle$ an output expression $\tau(\sigma_n, s)$ which has variables from the set $X_n \times S$. From the point of view of trees, the output Ω-tree may have leaves labeled from the set $X_n \times S$ as indicated in Fig. 3.

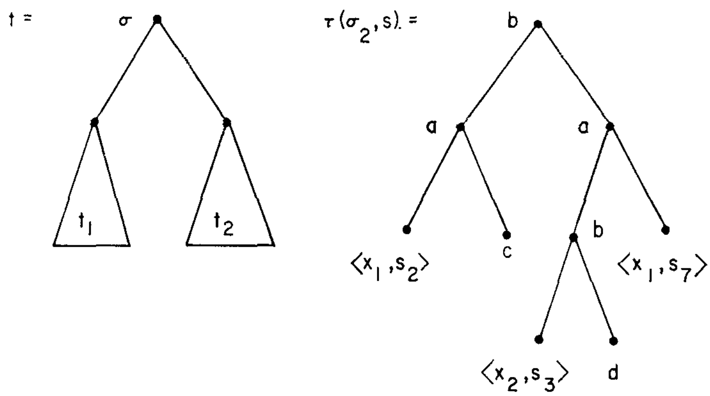


FIGURE 3

Although it may appear that there is no next-state function, it is actually included in the output function τ; the second component of the variable $\langle x_i, s_j\rangle$ specifies the "next-state." The extension of τ to a map $\bar{\tau} : T_{\Sigma,Z} \times S \to T_{\Omega,Z}$ is inductive. For

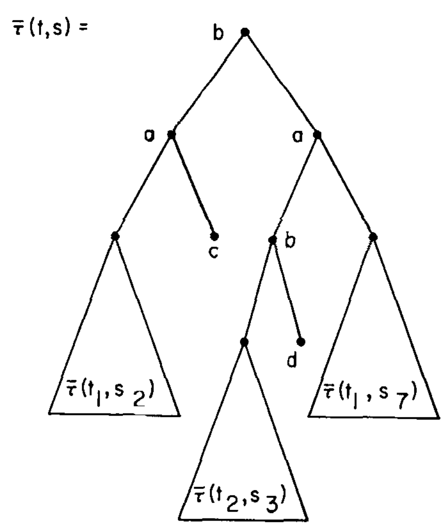


FIGURE 4

example, take $t = \sigma(t_1 t_2)$, and assume $\bar{\tau}(t_i, s)$ is defined for $i = 1, 2$ and all $s \in S$. Then $\bar{\tau}(t, s)$ is the result of substituting $\bar{\tau}(t_i, s_j)$ for $\langle x_i, s_j \rangle$ in $\tau(\sigma_2, s)$. This operation is pictured in Fig. 4.

In making the definition of $\bar{\tau}$ precise, the following preliminary definition is convenient. Let Z and Z' be arbitrary variable sets and let η be an assignment from Z' into $T_{\Sigma,Z}$. Then η^S is a map from $Z' \times S$ into $T_{\Sigma,Z} \times S$ which is the identity on S, i.e., $\eta^S = \eta \times 1_S$:

6.4. $\eta^S(z, s) = \langle \eta(z), s \rangle.$

Now the definition of $\bar{\tau}$ defined on $T_{\Sigma,Z} \times S$ and $T^{Z'}_{\Sigma,Z}$ is given by 6.5.

6.5. (0) $\bar{\tau}(z, s) = \langle z, s \rangle;$

(1) $\bar{\tau}(\eta) = \bar{\tau}\eta^S;$

(2) $\bar{\tau}(\sigma_n \cdot \eta, s) = \tau(\sigma_n, s) \cdot \bar{\tau}(\eta).$

One should quickly check that this definition is meaningful; indeed, since $\tau(\sigma_n, s) \in T_{\Omega,X\times S}$ and $\bar{\tau}(\eta) = \bar{\tau}\eta^S$ is an assignment from $X \times S \to T_{\Omega,Z\times S}$ the substitution operation in (2) is well-defined.

We can refer to $\bar{\tau}(t, s)$ as the transformation operating on t with root state s. $\bar{\tau}(\eta)$ is a complete picture of how $\bar{\tau}$ operates on each component of η in all possible states; $\bar{\tau}(\eta)(z, s) = \bar{\tau}(\eta(z), s)$. As is conventionally done, we can define $\bar{\tau}$ on $T_{\Sigma,Z}$ by $\bar{\tau}(t) = \bar{\tau}(t, s_0)$ i.e., $\bar{\tau}(t)$ is the value of the transformation operating on t with root state s_0. Our main interest is in the map $\bar{\tau} : T_\Sigma \to T_\Omega$ $(Z = \emptyset)$ and we will refer to transformations in this way. The extensions to $T_{\Sigma,Z}$ and to $T^{Z'}_{\Sigma,Z}$ are automatic and necessary for the definitions.

A transformation will be called *rank-preserving* if all of the variables $\{x_1, \ldots, x_n\}$ occur in $\tau(\sigma_n, s)$, i.e., for each σ_n, s and i, $1 \leqslant i \leqslant n$, $\langle x_i, s' \rangle \in \operatorname{var} \tau(\sigma_n, s)$ for some s'.

EXAMPLE 6.1. If the ranked alphabets Σ and Ω are both monadic ($\Omega_n = \Sigma_n = \emptyset$ for $n > 1$ and $\Omega_0 = \Sigma_0 = \{\Lambda\}$), then Σ' is in 1:1 correspondence with Σ and the restriction 6.3 requires that $\tau(\Lambda, s)$ be of the form $w(\Lambda)$ and that $\tau(\sigma, s)$ be of the form $w(x_1, s')$ or $w(\Lambda)$. (Here $w \in \Omega_1{}^*$ and $\lambda(\Lambda)$ or $\lambda(x_1, s')$ are taken as notations for Λ or $\langle x_1, s' \rangle$, respectively, cf. Example 2.2.) Consider a rank-preserving finite state transformation $\bar{\tau} : T_\Sigma \to T_\Omega$ determined by $\langle S, s_0, \tau \rangle$ in which $\tau(\Lambda, s) = \Lambda$ for all s. Such $\bar{\tau}$ are exactly the gsm mappings from $\Sigma_1{}^*$ into $\Omega_1{}^*$. Indeed, the corresponding gsm is $\langle S, s_0, \tau', \delta \rangle$ where for each s and $\sigma \in \Sigma_1$, if $\tau(\sigma, s) = w(x_1, s')$, then $\tau'(\sigma, s) = w$ and $\delta(\sigma, s) = s'$. Conversely, given a gsm $\langle S, s_0, \tau', \delta \rangle$, the corresponding finite state transformation is $\bar{\tau}$ determined by $\langle S, s_0, \tau \rangle$ when $\tau(\sigma, s) = \tau'(\sigma, s)(x_1, \delta(\sigma, s))$ and $\tau(\Lambda, s) = \Lambda$. If the condition on preserving rank is removed, then 6.3 could allow $\tau(\sigma, s) = w(\Lambda)$ which has the effect in the finite state transformation of "stopping" the output. An equivalent rank-preserving transformation can easily be obtained by introducing a new state s^* and having $\tau'(\sigma, s) = w(x_1, s^*)$, $\tau'(\sigma, s^*) = \langle x_1, s^* \rangle$ and τ' like τ otherwise. Then the new $\bar{\tau}'$ produces exactly the same map from T_Σ into T_Ω as the original $\bar{\tau}$. Therefore, with the restriction that $\tau(\Lambda, s) = \Lambda$, the finite state transformations (in the monadic case) coincide with the gsm mappings. Finally, removing the restriction, $\tau(\Lambda, s) = \Lambda$, takes us out of the domain of the gsm mappings. However, it should be clear that any finite state transformation can be represented as $\bar{\tau}(w)\, \rho(w)$ where $\bar{\tau}$ is a gsm map and $\rho(w)$ is a word in $\Omega_1{}^*$ depending only on $\bar{\delta}(w, s_0)$.

Before considering other examples of finite state transformations, we will prove some of the basic properties of these maps. Lemma 6.7 is the analog of the property,

6.6. $\bar{\tau}(wv, s) = \bar{\tau}(w, s)\, \bar{\tau}(v, \bar{\delta}(w, s))$,

in the conventional theory and Lemma 6.11 yields closure under composition.

LEMMA 6.7. *For any assignment $\xi : Z \to T_{\Sigma,Z}$ and expression $t \in T_{\Sigma,Z}$,*

$$\bar{\tau}(t \cdot \xi, s) = \bar{\tau}(t, s) \cdot \bar{\tau}(\xi).$$

Proof by induction on t.

(0) $\bar{\tau}(z \cdot \xi, s) = \langle z, s \rangle \cdot \bar{\tau}\xi^S$ — 5.3(0), 6.4

$\quad = \bar{\tau}(z, s) \cdot \bar{\tau}(\xi)$ — 6.5 (0, 1)

(1) Assuming 6.7 holds for $t = \eta(z)$ for all z, we obtain

6.7.1. $\bar{\tau}(\eta \cdot \xi) = \bar{\tau}(\eta) \cdot \bar{\tau}(\xi)$.

Proof. $\bar{\tau}(\eta \cdot \xi)(z, s) = \bar{\tau}(\eta \cdot \xi)^S (z, s)$ — 6.5(1)

$\quad = \bar{\tau}((\eta(z) \cdot \xi, s)$ — 6.4, 5.3(1)

$\quad = \bar{\tau}(\eta(z), s) \cdot \bar{\tau}(\xi)$ — IH

$\quad = (\bar{\tau}(\eta) \cdot \bar{\tau}(\xi))(z, s)$ — 6.4, 5.3(1)

(2) $\bar{\tau}((\sigma_n \cdot \eta) \cdot \xi, s) = \bar{\tau}(\sigma_n \cdot (\eta \cdot \xi), s)$ — 5.3(2)

$\quad = \tau(\sigma_n, s) \cdot \bar{\tau}(\eta \cdot \xi)$ — 6.5(2)

$\quad = \tau(\sigma_n, s) \cdot (\bar{\tau}(\eta) \cdot \bar{\tau}(\xi))$ — 6.7.1

$\quad = (\tau(\sigma_n, s) \cdot \bar{\tau}(\eta)) \cdot \bar{\tau}(\xi)$ — 5.4

$\quad = \bar{\tau}(\sigma_n \cdot \eta, s) \cdot \bar{\tau}(\xi)$. — 6.5(2)

This last result (6.7.1) says that every finite state transformation on T_Σ induces a morphism of pretheories, $\mathbf{T}_{\Sigma,Z}$ into $\mathbf{T}_{\Omega,Z\times S}$ and with the appropriate definition of morphism, the converse will probably hold. I believe this fact attests to the "naturalness" of the concept of finite state transformations. Where we can go from here and what interest there is in this result will have to await further analysis of the question on pretheories posed in Section 5.

Since we are dealing, in effect, with morphisms of algebraic structures, the fact that we obtain closure under composition is not surprising. The method is straightforward. Let $\bar{\tau} : T_\Sigma \to T_\Omega$ and $\bar{\rho} : T_\Omega \to T_\Delta$ be determined by $\langle S, s_0, \tau\rangle$ and $\langle U, u_0, \rho\rangle$, respectively. Define $\bar{\mu}$ by $\langle S \times U, \langle s_0, u_0\rangle, \mu\rangle$ where

6.8. $\mu(\sigma_n, s, u) = \bar{\rho}(\tau(\sigma_n, s), u)$.[15]

The following lemma yields the required result concerning the composition of $\bar{\tau}$ and $\bar{\rho}$.

LEMMA 6.9. *For finite state transformations* $\bar{\tau} : T_\Sigma \to T_\Omega$ *and* $\bar{\rho} : T_\Omega \to T_\Delta$, *the finite state transformation* $\bar{\mu} : T_\Sigma \to T_\Delta$ *defined by* 6.8 *is the composition* $\bar{\rho}\bar{\tau}$. *In particular, for all* $t \in T_{\Sigma,Z}$, $s \in S$ *and* $u \in U$,

$$\bar{\mu}(t, s, u) = \bar{\rho}(\bar{\tau}(t, s), u).$$

Proof by induction on t.

(0) $\bar{\mu}(z, s, u) = \langle z, s, u\rangle$ 6.5(0)

$\quad = \bar{\rho}(\bar{\tau}(z, s), u)$ 6.5(0)

(1) Assuming 6.9 is true for all $\eta(z)$, $z \in Z$; 6.4 and 6.5(1) yield

6.9.1. $\bar{\mu}(\eta) = \bar{\rho}(\bar{\tau}(\eta))$.

$$\begin{aligned}
(1)\quad \bar{\mu}(\sigma_n \cdot \eta, s, u) &= \mu(\sigma_n, s, u) \cdot \bar{\mu}(\eta) && 6.5(2)\\
&= \bar{\rho}(\tau(\sigma_n, s), u) \cdot \bar{\mu}(\eta) && 6.8\\
&= \bar{\rho}(\tau(\sigma_n, s), u) \cdot \bar{\rho}(\bar{\tau}(\eta)) && 6.9.1\\
&= \bar{\rho}(\tau(\sigma_n, s) \cdot \bar{\tau}(\eta), u) && 6.7\\
&= \bar{\rho}(\bar{\tau}(\sigma_n \cdot \eta, s), u) && 6.5(2)
\end{aligned}$$

EXAMPLE 6.2. (This will be used for Theorem 6.15.) Let $\bar{\tau} : a^* \to T_\Omega$ be determined by $\langle\{s_0, s_1\}, s_0, \tau\rangle$ where

τ	a	Λ
s_0	$a(s_1 s_0)$	Λ
s_1	$a(s_1)$	Λ

Here, Ω is the ranked alphabet $\{a, \Lambda\}$ with $\Omega_1 = \Omega_2 = \{a\}$ and $\Omega_0 = \{\Lambda\}$. Since the values of τ are in $T_{\Omega,\{x_1\}\times S}$ (cf. 6.3) and since $\{x_1\} \times S \cong S$, the occurrences of x_1 in the values of τ have, for convenience, been dispensed with. $\bar{\tau}(a^n(\Lambda), s_0) = \bar{\tau}(a^n(\Lambda))$ is the tree indicated by Fig. 5 of depth n.

[15] $\mu(\sigma_n, s, u)$ is $\mu(\sigma_n, (s, u))$ under the usual identification of $A \times (B \times C)$ with $A \times B \times C$.

Finite state transformations in the special case when $|S| = 1$ will be called ***pure.*** It is clear that when $|S| = 1$, talking about S at all is superfluous since $Z \times S$, $T_{\Sigma,Z\times S}$ and $\mathbf{T}_{\Sigma,Z\times S}$ are indistinguishable, respectively, from Z, $T_{\Sigma,Z}$ and $\mathbf{T}_{\Sigma,Z}$ in this development.

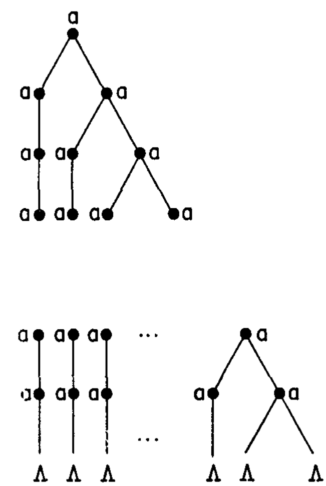


FIGURE 5

For convenience of reference, we restate 6.5 in this case. A pure finite state transformation (PFST) $\bar{\tau}$ is determined by a map $\tau : \Sigma' \to T_{\Omega,X}$. $\bar{\tau}$ is defined inductively by

6.10. (0) $\bar{\tau}(z) = z$;

(1) $\bar{\tau}(\eta) = \bar{\tau}\eta$;

(2) $\bar{\tau}(\sigma_n \cdot \eta) = \tau(\sigma_n) \cdot \bar{\tau}(\eta)$.

The pure transformations generalize what are commonly referred to as homomorphisms in language theory and clearly they include the projections (cf., 3.2) as the special case. It should also be clear that for $Z = X$, the pure FST's are exactly the morphisms of the corresponding free algebraic theories (cf. Example 5.3).

LEMMA 6.11. *Every projection is a (pure) finite state transformation.*

Recall the definition of X-surface set from Section 4; a finite state surface set is the image of a recognizable set under a finite state transformation. From Lemmas 3.5, 6.11, and 6.9, we obtain

THEOREM 6.12. *Any finite state surface set can be obtained as the image of a local set.*

THEOREM 6.13. *The pure transformations are properly a special case of the finite state transformations.*

Proof. The FST of Example 6.2 cannot be obtained as a PFST.

EXAMPLE 6.3. (This will be used for Theorem 6.14 and Lemmas 8.7.) Define the pure transformation $\bar{\tau} : \{a, b\}^* \to T_\Omega$ where $\Omega_0 = \{\Lambda\}$, $\Omega_2 = \{a, b\}$, $\tau(\Lambda) = \Lambda$, $\tau(a) = a(x_1 x_1)$, and $\tau(b) = b(x_1 x_1)$. For $w \in \{a, b\}^*$, $\bar{\tau}(w(\Lambda))$ is the balanced binary tree with each level labeled entirely by either a or b depending on whether the corresponding symbol of w is a or b. The leaves of $\bar{\tau}(w(\Lambda))$ are labeled by Λ and $fr\bar{\tau}(w(\Lambda)) = \Lambda^{2^n}$ where $n = lg(w)$.

THEOREM 6.14. *Pure finite state transformational languages properly include the context-free languages and the pure finite state surface sets properly include the recognizable sets.*

Proof. Every recognizable subset of T_Σ is a pure finite state surface set under the identity transformation. Thus by Theorem 3.8, every context-free language is a pure finite state transformational language. Both proper inclusions are given by Example 6.3 since the language $\{\Lambda^{2^n}\}$ is not context-free [5] and thus the surface set $\bar{\tau}(\{a, b\}^*)$ is not recognizable.

7. SUBSTITUTION AND SETS OF EXPRESSIONS

This section is an extension of Section 5 to sets of terms. Like the relationship between Sections 5 and 6, it will serve as the basis for considerations of nondeterministic finite state transformations in Section 8. This is not the only interest. The material of this section will also provide the framework for a general treatment of certain kinds of definability similar to regularity and the minimal fixed point definitions of Ginsburg and Rice [17]; Rose [25]; and Mezei and Wright [21].

Let $\langle \Sigma, r \rangle$ be a ranked alphabet. For any Σ-algebra $\mathscr{A} = \langle A, \alpha \rangle$ we construct another Σ-algebra, the *subset algebra* $\hat{\mathscr{A}} = \langle pA, \hat{\alpha} \rangle$ (cf. [21], Definition 2.2) where for $u_i \subseteq A$.

7.1. $\hat{\alpha}_\sigma(u_1, \ldots, u_n) = \{\alpha_\sigma(a_1 \cdots a_n) \mid a_i \in u_i\}$.

As is well-known, these induced "complex functions" are completely distributive (cf. [21]).

LEMMA 7.2. *For any set A and $f : A^n \to A$,*

$$\hat{f}\Big(\bigcup_{i_1} u_{1i_1} \cdots \bigcup_{i_n} u_{ni_n}\Big) = \bigcup_{i_1} \cdots \bigcup_{i_n} \hat{f}(u_{1i_1} \cdots u_{ni_n}).$$

In particular, if $\mathscr{A} = \mathscr{T}_{\Sigma,Z}$, then $\hat{\mathscr{T}}_{\Sigma,Z}$ has $pT_{\Sigma,Z}$ as carrier and functions $\hat{\iota}_\sigma$ defined by

7.3 $\hat{\iota}_\sigma(u_1 \cdots u_n) = \{\sigma(t_1 \cdots t_n) \mid t_i \in u_i\}$.

The subalgebra of $\hat{\mathscr{T}}_{\Sigma,Z}$ with carrier consisting of singleton sets $\{t\}$ for $t \in T_{\Sigma,Z}$ is obviously isomorphic to $\mathscr{T}_{\Sigma,Z}$. Under this isomorphism, we will identify t with $\{t\}$ in $\hat{\mathscr{T}}_{\Sigma,Z}$.

Given $\xi : Z \to pT_{\Sigma,Z}$ the homomorphism (2.6) $h_\xi : \mathscr{T}_{\Sigma,Z} \to \hat{\mathscr{T}}_{\Sigma,Z}$ is uniquely determined. Analogous to the substitution operation, $\cdot$, we will define a binary operation $\triangle$ by $t \triangle \xi = h_\xi(t)$. This also gives rise to the induced term function of Mezei and Wright [21], which they denoted $\hat{t}$. In particular, where $\operatorname{var}(t) = \{x_{i_1}, \ldots, x_{i_n}\}$, $\hat{t}(\xi(x_{i_1}) \cdots \xi(x_{i_n})) = t \triangle \xi$. Although we already have the definition of $t \triangle \xi$ by 2.6, for purposes of reference, we will repeat it here and, at the same time, extend the definition to assignments which have sets as values. First, for $\sigma_n \in \Sigma'$ and $\xi : X \to pT_{\Sigma,Z}$, we introduce the notation,

7.4. $\sigma_n \triangle \xi = \hat{\iota}_\sigma(\xi(x_1) \cdots \xi(x_n))$.

Then the extension of 7.4 to expressions, sets of expressions, and assignments to sets is as follows.

7.5. (0) $z \triangle \xi = \xi(z)$;

(1) For $V \in pT_{\Sigma,Z}$, $V \triangle \xi = \bigcup_{t \in V} t \triangle \xi$;

(2) $(\zeta \triangle \xi)(z) = \zeta(z) \triangle \xi$;

(3) $(\sigma_n \cdot \eta) \triangle \xi = \sigma_n \triangle (\eta \triangle \xi)$.

Observe that 7.5 successfully defines $t \triangle \xi$ for arbitrary t because for $\eta : X \to T_{\Sigma,Z}$, assuming $\eta(x) \triangle \xi$ has been defined for all x, then $(\eta \triangle \xi)(x) = \eta(x) \triangle \xi$ is defined and then $(\sigma_n \cdot \eta) \triangle \xi = \sigma_n \triangle (\eta \triangle \xi)$ is defined by 7.4.

EXAMPLE 7.1. Let ξ_i be defined by the following table (t identified with $\{t\}$).

	x_1	x_2	$x_i\ (i > 2)$
ξ_1	a	a	x_i
ξ_2	b	b	x_i
ξ_3	$\{a, b\}$	$\{a, b\}$	x_i

Then

$$a(x_1x_2) \triangle \xi_3 = \{a(ab), a(ba), a(aa), a(ab)\}$$

and

$$a(x_1x_2) \triangle \xi_1 \cup a(x_1x_2) \triangle \xi_2 = \{a(aa), a(bb)\};$$

also,

$$\begin{aligned} a(x_1x_1) \triangle \xi_3 &= (a(x_1x_2) \cdot \langle x_1 , x_1 \rangle) \triangle \xi_3 \\ &= a(x_1x_2) \triangle (\langle x_1 , x_1 \rangle \triangle \xi_3) \\ &= a(x_1x_2) \triangle (\langle \{a, b\}, \{a, b\} \rangle) \\ &= a(x_1x_2) \triangle \xi_3 . \end{aligned}$$

Before proving the analog of 5.4, we must verify a distributivity property of $\triangle$.

LEMMA 7.6. $(\cup V_i) \triangle \xi = \bigcup_i (V_i \triangle \xi)$.

Proof by 7.5(1).

We are now in a position to prove the associativity of $\triangle$, first in the limited form, for $\sigma_n \in \Sigma'$.

LEMMA 7.7. $(\sigma_n \triangle \zeta) \triangle \xi = \sigma_n \triangle (\zeta \triangle \xi)$.

Proof.

$$\begin{aligned} (\sigma_n \triangle \zeta) \triangle \xi &= \hat{\iota}_\sigma(\zeta(x_1) \cdots \zeta(x_n)) \triangle \xi && 7.4 \\ &= \bigcup_{t_1 \in \zeta(x_1)} \cdots \bigcup_{t_n \in \zeta(x_n)} \sigma(t_1 \cdots t_n) \triangle \xi && 7.5(1),\ 7.3 \\ &= \bigcup_{t_1 \in \zeta(x_1)} \cdots \bigcup_{t_n \in \zeta(x)_n} \hat{\iota}_\sigma(t_1 \triangle \xi \cdots t_n \triangle \xi) && 7.5(3),\ 7.4 \\ &= \hat{\iota}_\sigma\left(\bigcup_{t_1 \in \zeta(x_1)} t_1 \triangle \xi \cdots \bigcup_{t_n \in \zeta(x_n)} t_n \triangle \xi \right) && 7.2 \\ &= \hat{\iota}_\sigma(\zeta(x_1) \triangle \xi \cdots \zeta(x_n) \triangle \xi) && 7.5(1) \\ &= \hat{\iota}_\sigma(\zeta \triangle \xi(x_1) \cdots \zeta \triangle \xi(x_n)) && 7.5(2) \\ &= \sigma_n \triangle (\zeta \triangle \xi). && 7.4 \end{aligned}$$

And the extension of 7.7 is given by

LEMMA 7.8. $(t \triangle \zeta) \triangle \xi = t \triangle (\zeta \triangle \xi)$.

Proof by induction on t paralleling the proof of Lemma 5.4. Analogous to 5.4.1, the detailed inductive proof yields as corollaries:

7.8.1. $V \triangle (\zeta \triangle \xi) = (V \triangle \zeta) \triangle \xi$;

7.8.2. $\psi \triangle (\zeta \triangle \xi) = (\psi \triangle \zeta) \triangle \xi$.

With the associativity stated in 7.8.2, we have another algebraic structure $\mathbf{pT}_{\Sigma,Z} = \langle (pT_{\Sigma,Z})^Z, \triangle, 1_Z \rangle$ which is again a monoid and which we will also call a pretheory. The nondeterministic finite state transformations will involve morphisms into this pretheory.

EXAMPLE 7.3. For $\eta : X \to pT_{\Sigma,X}$, assume that for some m, $\eta(x_i) = x_i$ for $i > m$ and that $\eta(x)$ is finite for every x. Define η^* in the natural way, $\eta^0 = 1_X$, $\eta^{k+1} = \eta^k \triangle \eta$ and $\eta^*(x) = \bigcup_i \eta^i(x)$. With reference to Mezei and Wright [21], η corresponds to a "system of equations" (written $x_i = \eta(x_i)$). In $\mathbf{pT}_{\Sigma,Z}$, the "system function" (as a function of ξ) is $\eta^* \triangle \xi$ and the minimal fixed point of the system of equations is $\eta^* \triangle \emptyset$, the sets defined by η are $\eta^* \triangle \emptyset(x_i) = \eta^*(x_i) \triangle \emptyset$, $i = 1,..., m$.

8. NONDETERMINISTIC FINITE STATE TRANSFORMATIONS

Nondeterminism is, as usual, introduced by letting the basis function have values which are sets. Thus, a *nondeterministic finite state transformation* (NFST) $\bar{\tau}$ from T_Σ into T_Ω is determined by a triple $\langle S, S_0, \tau \rangle$ where S is the finite set of states, the initial states are S_0 and $\tau : \Sigma' \times S \to pT_{\Omega,X\times S}$ with $\tau(\sigma_n, s)$ a finite subset of $T_{\Omega,\{x_1,\dots,x_n\}\times S}$. The extension to $\bar{\tau} : T_{\Sigma,Z} \times S \to pT_{\Omega,Z\times S}$ is identical in form to 6.5.

8.1. (0) $\bar{\tau}(z, s) = \langle z, s \rangle$;

(1) $\bar{\tau}(\eta) = \bar{\tau}\eta^S$;

(2) $\bar{\tau}(\sigma_n \cdot \eta, s) = \tau(\sigma_n, s) \triangle \bar{\tau}(\eta)$.

For $t \in T_{\Sigma,Z}$,

$$\bar{\tau}(t) = \bigcup_{s\in S_0} \bar{\tau}(t, s) = \hat{\bar{\tau}}(t, S_0)$$

and by the restriction above, for $t \in T_\Sigma$, $\bar{\tau}(t) \subseteq T_\Omega$.

As in Section 6, the following lemma can be interpreted as saying that $\bar{\tau}$ is extended to a morphism of $\mathbf{T}_{\Sigma,Z}$ into $\mathbf{pT}_{\Sigma,Z}$.

LEMMA 8.2. $\bar{\tau}(t \cdot \xi, s) = \bar{\tau}(t, s) \triangle \bar{\tau}(\xi)$.

Proof. The proof is identical in form to that of 6.7 with $\triangle$ replacing $\cdot$ (where the operation is in $\mathbf{pT}_{\Omega,Z\times S}$), 8.1 replacing 6.5 and 7.8 replacing 5.4.

As a corollary to the proof of 8.2, analogous to 6.7.1, we obtain

COROLLARY 8.2.1. $\bar{\tau}(\eta \cdot \xi) = \bar{\tau}(\eta) \triangle \bar{\tau}(\xi)$.

As in Section 6, the *pure* NFST's are obtained when $|S| = 1$ and the states can be eliminated from the notation.

EXAMPLE 8.1. Let $\bar{\tau}$ be a pure NFST from $\{a, b, c\}^*$ into T_Ω, where $\Omega_0 = \{0, 1\}$, $\Omega_1 = \Omega_2 = \Omega_3 = \{a\}$ and τ is given by

	a	b	c	Λ
τ	$a(x_1x_1)$	$\{a(1), a(x_1 1 x_1)$ $a(x_1 1), a(1x_1)\}$	$\{a(0), a(0x_1)\}$	0

Then

$$fr\bar{\tau}(c^n(\Lambda)) = \{0^k \mid 1 \leqslant k \leqslant n + 1\}$$

because

$$fr\bar{\tau}(\Lambda) = fr\{0\} = 0$$

and

$$\bar{\tau}(c^{n+1}(\Lambda)) = \{a(0), a(0, x_1)\} \triangle \langle\bar{\tau}(c^n)\rangle = \{a(0)\} \cup \{a(0, t) \mid t \in \bar{\tau}(c^n)\}.$$

Also,

$$fr\bar{\tau}(bc^n(\Lambda)) = \{0^m10^k \mid 0 \leqslant m, k \leqslant n + 1\}.$$

Finally, $\bar{\tau}(a^p(x_1))$ is the balanced binary tree labeled with "a" such that $fr\bar{\tau}(a^p(x_1)) = x_1^{2^p}$. Looking at $\bar{\tau}(a^pbc^n(\Lambda))$, by Lemma 8.2, this is $\bar{\tau}(a^p(x_1)) \triangle \langle\bar{\tau}(bc^n(\Lambda))\rangle$ and the frontier of any expression in this set has 2^p 1's with an arbitrary number ($\leqslant n$) of 0's, between each occurrence of 1.

To attack the question of the composition of nondeterministic transformations, we will first consider a special case. Let $\bar{\tau}$ be a (deterministic) FST from T_Σ into T_Ω and let $\bar{\rho}$ be a nondeterministic FST from T_Ω into T_Δ where $\bar{\tau}$ and $\bar{\rho}$ are determined by $\langle S, s_0, \tau\rangle$ and $\langle U, U_0, \rho\rangle$, respectively. Define (nondeterministic) $\bar{\mu}$ by $\langle S \times U, \{s_0\} \times U_0, \mu\rangle$ where μ is defined by 6.10, (i.e., $\mu(\sigma_n, s, u) = \bar{\rho}(\tau(\sigma_n, s), u)$).

LEMMA 8.3. *With the definitions above,*

$$\bar{\mu}(t, s, u) = \bar{\rho}(\bar{\tau}(t, s), u).$$

Proof. Again, the proof exactly parallels that of Lemma 6.9 with appropriate replacements of $\triangle$ for $\cdot$ and of 8.1 for 6.5 and 8.2 for 6.7.

With Lemma 8.3, we can now show that $\bar{\mu}$ yields the composition of $\bar{\tau}$ and $\bar{\rho}$.

LEMMA 8.4. $\bar{\mu}(t) = \bar{\rho}\bar{\tau}(t)$.

Proof is immediate from Definitions 8.1 and 6.5 and Lemma 8.3.

COROLLARY 8.5. *Any nondeterministic finite state surface set is obtained as the image of a local set.*

Proof. The recognizable sets are projections of local sets (3.5) and a projection (cf. 6.11) followed by an NFST is still an NFST (8.4).

We have proved a special case of a desirable composition result. The following lemma would make that general result particularly interesting.

LEMMA 8.6. *Any local subset of T_Σ is the image of a^* under a nondeterministic finite state transformation.*

Proof. Since we will be looking at $\bar{\tau}$ defined on a^*, the values of τ are in $T_{\Sigma,\{x_1\}\times S}$ and we can identify $\langle x_1, s\rangle$ with s. Let $G = \langle R, S_0\rangle$ be a context-free grammar over Σ. Let $\bar{\tau}$ be determined by $\langle \Sigma, S_0, \rho\rangle$ where

8.6.1. (0) $\rho(\Lambda, \sigma) = \begin{cases}\{\sigma\} & \text{if } \langle \sigma, \lambda\rangle \in R;\\ \emptyset & \text{otherwise.}\end{cases}$

(1) $\rho(a, \sigma) = \{\sigma(w) \mid w \in \Sigma^+ \text{ and } \langle \sigma, w\rangle \in R\} \cup \{\sigma\}$.

Then we claim

8.6.2. $\hat{\rho}(a^*, \sigma) = D_\sigma^G$.

This, of course, implies $\hat{\rho}(a^*) = T(G)$ since $T(G)$ is just the union of D_σ^G for $\sigma \in S_0$. We will omit the details of the proof of 8.6.2. It consists of showing that $\hat{\rho}(a^*, \sigma) \subseteq D_\sigma^G$ and that the collection $\{\hat{\rho}(a^*, \sigma)\}_\sigma$ satisfies 3.1. Because 3.1 is inductive, this yields the required result.

With Lemma 8.6 and a general composition result, we would have that every nondeterministic surface set could be obtained as the image of a^*, under a nondeterministic transformation, a very interesting and useful fact. Of course, we are leading to a negative result.

EXAMPLE 8.2. Let $\bar{\rho} : a^* \to p\{a, b\}^*$ be determined by $\rho(a) = \{a(x_1), b(x_1)\}$ and $\rho(\Lambda) = \Lambda$. Then $\rho(a^n(\Lambda)) = \{w(\Lambda) \mid w \in \{a, b\}^* \wedge lg(w) = n\}$.

LEMMA 8.7. *The nondeterministic transformations are not closed under composition.*

Proof. We will consider $\bar{\rho} : a^* \to p\{a, b\}^*$ of Example 8.2 and $\bar{\tau} : \{a, b\}^* \to T_\Omega$ of Example 6.3. These are both very special cases; $\bar{\tau}$ is deterministic (and pure) and thus we have a situation dual to the condition of Lemma 8.4. Also $\bar{\rho}$ is pure and, in fact, $\bar{\rho}(a^*)$ is local. But the composition, $\hat{\bar{\tau}}\bar{\rho}$ is not a nondeterministic finite state transformation.

Recall that $\bar{\rho}(a^n(\Lambda)) = \{w(\Lambda) \mid w \in \{a, b\}^n\}$ and $\bar{\tau}\ (w(\Lambda))$ is the balanced binary tree with the n-th level labeled with the n-th symbol of w. So $\hat{\bar{\tau}}\bar{\rho}(a^n(\Lambda))$ is the set of balanced binary trees of height $n+1$ labeled with $\{a, b\}$ subject to the restriction that all nodes on a given level have the same label. Call this set E_n. Observe that if t is not a variable and if $t \triangle \eta \subseteq E_n$ then each $\eta(z)$ for $z \in \text{var}(t)$ must be a singleton. For if $t_1, t_2 \in \eta(z)$ then one can substitute for all but one occurrence of z in t—call the result $t'(z)$—and obviously there is a unique t'' such that $t(t'') \in E_n$, i.e., not both $t'(t_1)$ and $t'(t_2)$ can be in E_n for $t_1 \neq t_2$.

Now assume that $\hat{\tau}\bar{\rho}$ is a NFST determined by $\langle S, S_0, \mu\rangle$. Let $E'_{n,s} = \bar{\mu}(a^n(\Lambda), s)$. Then we must have $\bigcup_{s\in S_0} E'_{n,s} = E_n$. But by Lemma 8.2, $E'_{n+1}, s = \mu(a(x_1), s)\, \Delta\eta$, when $\eta(x_1, s') = \bar{\mu}(a^n(\Lambda), s')$. By the observation above, all $\bar{\mu}(a^n(\Lambda), s')$ must be singletons when $\langle x_1, s'\rangle$ occurs as a variable in $t \in \mu(a(x_1), s)$. Thus $| E'_{n+1,s} | \leqslant | \mu(a(x_1), s)|$ and therefore $| \bigcup_{s\in S_0} E'_{n,s} |$ is bounded, whereas $| E_{n+1} | = 2^n$. This is a contradiction and therfore $\hat{\tau}\bar{\rho}$ is not an NFST.

9. Conclusion

Language theory has, to a large extent, dealt with classification and recognition problems for subsets of a free monoid. Quite frequently a family of languages is defined by some automaton or production model and a standard list of closure, decidability, and inclusion (relative to other families) questions are investigated. This is a familiar procedure and it has proved fruitful. The introduction of translations and transformations [19, 12, 31, 22, 20, 26, 2, 23] opens a very interesting area for language theorists. It appears to be an area with considerable promise of application to questions of syntax and semantics of programming languages and to the analysis of natural languages.

It is the author's contention that "transformation theory" is distinctly a new area of language theory, one in which caution should be exercised in applying the judgments and procedures of the past. Without such caution, transformation theory will only provide a new cycle of applications of the old familiar procedure. For each family of languages Y (with which one can associate derivation trees) and for each transformation definition X, one obtains a new family of languages, the X-translations of Y-languages, a veritable Pandora's box of families of languages (PFL).

These possibilities do not nearly exhaust the potential applications of the above procedures to transformation theory. For example, one can generalize Post production systems for expressions. For purposes of completeness and in the hope of promoting interest in the subject, we include the definition here.

A *production system* over Σ consists of a triple $\mathscr{P} = \langle \Omega, P, A\rangle$ where Ω, disjoint from Σ, is a ranked alphabet of *syntactic variables*, $A \subseteq T_\Sigma$ is a set of *axioms* and $P \subseteq T_{\Omega\cup\Sigma} \times T_{\Omega\cup\Sigma}$ the set of *productions*. For $t_1, t_2 \in T_\Sigma$, $t_1 \rightarrow_{\mathscr{P}} t_2$ if and only if there exists a pure transformation $\bar{\tau} : T_{\Omega\cup\Sigma} \rightarrow T_\Sigma$ which is the identity on T_Σ and a production $(t_1', t_2') \in P$ such that $\bar{\tau}(t_i') = t_i$. The relation $\Rightarrow_{\mathscr{P}}$ is the transitive reflexive closure of $\rightarrow_{\mathscr{P}}$. The set of *theorems* of $\mathscr{P}$ is $T(\mathscr{P}) = \{t \mid t' \Rightarrow_{\mathscr{P}} t \text{ for } t' \in A\}$.

In the monadic case, this definition reduces to Post canonical systems. The general definition lends itself to all the restrictive forms (with even more potential for variation in each category) that have been considered for Post systems.

Although this area is quite open for investigation, it is not completely untouched. Brainerd, in his thesis, [7], considered "regular tree grammars" which, in the context of the definition above, formally generalize Post systems with productions of the form $Xw \rightarrow Xv$ (X, a syntactic variable, w, $v \in \Sigma^*$), the left regular systems considered (together with right-regular) by Büchi [8]. Brainerd proves the result generalizing Büchi's, that a set is recognizable if and only if it is the set of theorems of a regular tree grammar. Further development of transformation theory or generalized finite automata theory should explain this predictable result without recourse to a reconstruction of the original proof in the generalized context.

A second paper, which is related to generalized production systems, is by Peters and Ritchie [22]. These authors, attempting to formalize the notion of grammatical transformation (cf. [11]) arrived at a rather complicated definition which is similar to the concept of production specified above. The exact nature of the relationship requires further investigation, but it is clear from the results in Peters and Ritchie, [22] that their transformational languages fall outside the area of finite state transformational languages defined here.

The principal definition given here, that of nondeterministic finite state transformation, is closely related to the one investigated by Rounds [26]. Consider a finite state machine (gsm) with two adjacent reading heads on its input tape. Depending on the current state and two current input symbols, the machine changes state, produces an output word, and moves the two adjacent reading heads one square to the right. In the same sense that this model generalizes the one described in the beginning of Section 6, so Rounds' definition generalizes ours.

By way of example, a two-level (deterministic) transformation $\bar{\tau}$ is determined by a basis map $\tau : \Sigma'' \times S \rightarrow T_{\Sigma, X \times S}$ where $\Sigma'' = \{\sigma(w) \mid \sigma \in \Sigma,\ w \in \Sigma^* \text{ and } r(\sigma, lg(w))\}$. The extension of τ parallels 6.5.

9.1. (0) $\bar{\tau}(z, s) = \langle z, s \rangle$;

(1) $\bar{\tau}(\eta) = \bar{\tau}\eta^S$;

(2) $\bar{\tau}(\sigma_n \cdot \eta, s) = \tau(\sigma(\text{top}(\eta(x_1)) \cdots \text{top}(\eta(x_n)), s) \cdot \bar{\tau}(\eta)$.

So Round's definition includes 6.5 as a special case, but it certainly does not limit the possibilities. There are n-level transformations generalizing the finite state machine model with n adjacent reading heads.

From the point of view of language theory, one immediately observes that nothing new is obtained with these generalizations. For any recognizable set V over Σ and two-level transformation $\bar{\tau}$, one can find a new recognizable set V' over an extended alphabet (say, $\{\langle \sigma, w\rangle \mid \sigma \in \Sigma, w \in \Sigma^* \text{ and } r(\sigma, lg(w))\}$) such that $\bar{\tau}(V) = \bar{\tau}'(V')$ for some transformation $\bar{\tau}'$ as defined in Section 6. But the distinctions are not to be dismissed because there are new transformations with increased number of levels and if V happens to be the set of derivations of some given programming language and one is interested in translations of that language, then a new V' and a simpler $\bar{\tau}'$ might not be of interest.

Rounds' thesis is illustrative of another aspect of what we are calling transformation theory. Whereas conventional language theory could be assessed as having, as a basic component, the theory of Σ^* under concatenation, likewise, transformation theory (or general automata theory) is based on the theory of the set of Σ-expressions under substitution. One is more familiar with the former structure and some care must be taken with the latter; Sections 5 and 7 deal specifically with that area.

In the abstract, which preceded this paper, it was stated that every nondeterministic transformational language is context-sensitive. The author's proof of that result depended on closure under composition of the nondeterministic transformations. But the closure theorem which Rounds had stated turned out to be false[16] (Lemma 8.7) so that the inclusion of the nondeterministic languages in the class of context sensitive languages remains a strong conjecture. It appears as though Rounds' erroneous statement and the author's conviction about the result can be traced to an imprecise treatment of substitution.

Rounds suggests a connection between the indexed languages defined by Aho [1] and the nondeterministic transformational languages. M. Fischer [16] defines macrogrammars and from this concept, the classes of outside-in (OI) and inside-out (IO) languages. He proves that a language is indexed if and only if it is OI. Because of the algebraic framework of Fischer's work, the connection between the macrolanguages and the transformational languages is even more striking. Example 8.1 yields a nondeterministic transformational language which is Fischer's example of an OI language which is not IO. An attempt to investigate the relationship between the IO languages and the transformational language led to another area of interest. In applications to translations of programming languages, it would not be unreasonable to consider two (or more) transformations, $\bar{\tau}_1$ and $\bar{\tau}_2$, acting in parallel, with the final result of the operation being a map $\bar{\tau}$ obtained as some combination of the $\bar{\tau}_i$. For example, consider $\bar{\tau}_1$ on a^* determined by $\langle\{s_0, s_1\}, s_0, \tau\rangle$ where

$$\tau_1(a, s_0) = a(s_0 s_1), \qquad \tau_1(a, s_1) = a(s_1 s_1),$$

$$\tau_1(\Lambda, s_0) = a(x_1 c x_1) \qquad \text{and} \qquad \tau_1(\Lambda, s_1) = a(c x_1).$$

[16] Contrary to Rounds' claim (and proof), neither are the two-level deterministic transformations closed under composition. Indeed, an earlier version of this paper contained a proof of the closure of the linear nondeterministic transformations under composition. That proof also turned out to be in error. The author is grateful to B. K. Rosen for pointing out this problem.

Also, let $\bar{\tau}_2$ be defined by $\tau_2(a) = a(x_1 1)$ and $\tau_2(\Lambda) = 1$. Then the frontier of $\bar{\tau}_1(a^n, s_0)$ is $x_1(cx_1)^{2^{n+1}-1}$ and if we define $\bar{\tau}(t) = \bar{\tau}_1(t, s_0) \cdot \langle \bar{\tau}_2(t) \rangle$, then $fr\bar{\tau}(a^*)$ is the language L_2 [16] which is IO but not OI. Such compositions of transformations provide a broad area for further study.

We have already mentioned a connection with the work of Eilenberg and Wright [14] and Mezei and Wright [21] (Examples 5.3 and 7.3, respectively) and indicated the need for further investigation in the former case with the discussion of "pretheories" in Section 5. In Section 7, we used the notation $\mathbf{pT}_{\Sigma,Z}$ for the monoid obtained with the substitution operation $\triangle$. One might have expected to have seen the notation $\hat{\mathbf{T}}_{\Sigma,Z}$. This latter notation already has a meaning (cf. 7.1); it denotes the monoid with carrier $p(T^Z_{\Sigma,Z})$ and complex operation $u \cdot v = \{\eta \cdot \psi \mid \eta \in u \text{ and } \psi \in v\}$. Substitution in this subset structure gives rise to a new concept of nondeterministic transformation which bears the same relationship to the definition in Section 8 as Rose's work [25] (extended definable sets) bears to that of Mezei and Wright [21] and Ginsburg and Rice [17]. These relationships will be investigated in a forthcoming paper [30].

Studies in the area of syntax-directed translations have been restricted to very special cases of the finite state transformations. We have already seen the importance of linear transformations in Section 9. As a further restriction, when

$$fr_{X \times S} t = \langle x_1 s_1 \rangle \cdots \langle x_n , s_{i_n} \rangle$$

for every $t \in \tau(\sigma_n , s)$, τ will be called *simple*. Thus, every simple transformation is linear and rank-preserving, but not conversely.

The class of syntax-directed translations (Irons, [19]; Lewis and Stearns, [20]; Aho and Ullman, [2]) correspond to the class of linear rank-preserving pure transformations. The simple syntax-directed translations correspond to the simple pure transformations. The generalized Syntax-directed translation schemes of Aho and Ullman [3] correspond to the two-level deterministic finite state transformations.

Acknowledgment

The author is very grateful to two referees who offered many suggestions for improvements in the manuscript. It is unfortunate that under the questionable standards of the profession, this acknowledgment is meaningless since the referees must remain anonymous.

References

1. A. V. Aho, Indexed grammars—An extension of the context-free grammars, *J. ACM* **15** (1968), 647–671.
2. A. V. Aho and J. D. Ullman, Automaton analogs of syntax directed translation schemata, *Proc. 9th Ann. Symp. on Switching and Automata Theory*, October 1968.

3. A. V. Aho and J. D. Ullman, Translating on a context-free grammar, *Proc. ACM Symp. Theory of Computing* (1968), 93–112.
4. M. A. Arbib and Y. Give'on, Algebra Automata I: Parallel programming as a prolegomena to the categorical approach, *Information and Control* **12** (1968), 331–345.
5. Y. Bar-Hillel, M. Perles, and E. Shamir, On formal properties of phrase structure grammars, *Z. Phonetik, Sprach. Kummunikationsforsch.* **14** (1961), 143–172.
6. W. S. Brainerd, "Tree Generating Systems and Tree Automata," Ph.D. dissertation, Purdue University, June 1967.
7. W. S. Brainerd, Minimalization of tree automata, snbmittud for publication.
8. J. R. Büchi, Regular canonical systems, *Arch. Math. Logik Grundlagenforschung* **6** (1964), 91–111.
9. A. W. Burks and J. B. Wright, Sequence generators and digital computers, "Recursive Function Theory," Proc. Symposia Pure Math., Vol. V, pp. 139–199, Amer. Math. Soc., Providence, R. I., 1962.
10. N. Chomsky, On the notion of 'rule of grammar,' "Structure of Language and Its Mathematical Aspects," Proc. Symposia Appl. Math., Vol. XII, pp. 6–24, Amer. Math. Soc., Providence, R. I., 1961.
11. N. Chomsky, "Aspects of the Theory of Syntax," The M.I.T. Press, Cambridge, Massachusetts, 1965.
12. K. Culik, On some transformations in context-free grammars and languages, *Czechoslovak Math. J.* **17** (1967), 278–311.
13. J. Doner, Tree acceptors and some of their applications, submitted for publication, 1965.
14. S. Eilenberg and J. B. Wright, Automata in general algebras, *Information and Control* **11** (1967), 452–470.
15. C. C. Elgot, Decision problems of finite automaton design and related arithmetics, *Trans. Amer. Math. Soc.* **98** (1961), 21–51.
16. Michael J. Fischer, Grammars with macro-like productions, *Proc. 9th Ann. Symp. on Switching and Automata Theory* (October 1968).
17. S. Ginsburg and H. G. Rice, Two families of languages related to ALGOL, *J. ACM* **9** (1962), 350–371.
18. S. Ginsburg, "The Mathematical Theory of Context-Free Languages," McGraw-Hill Book Co., New York, 1966.
19. E. T. Irons, A syntax directed compiler for ALGOL 60, *C. ACM* **4** (1961), 51–55.
20. P. M. Lewis and R. E. Stearns, Syntax directed transduction, *J. ACM* **15** (1968), 465–488.
21. J. Mezei and J. B. Wright, Algebraic automata and context-free sets, *Information and Control* **11**, (1967), 3–29.
22. S. Peters and R. W. Ritchie, On the generative power of transformational grammars, unpublished manuscript, 1967.
23. L. Petrone, Syntax directed mapping of context-free languages, *Proc. 9th Ann. Symp. on Switching and Automata Theory* (October 1968).
24. M. O. Rabin, Mathematical theory of automata, "Mathematical Aspects of Computer Science," Proc. Symp. Appl. Math., Vol. XIX, pp. 153–175, Am. Math. Soc., Providence, R. I., 1967.
25. G. F. Rose, An extension of ALGOL-like languages, *C. ACM* **7** (1964), 52–61; Also, SDC TM-738/003/00 May, 1963.
26. W. C. Rounds, "Trees, Transducers, and Transformations," Ph.D. dissertation, Stanford University, August 1968.
27. J. W. Thatcher, Characterizing derivation trees of context-free grammars through a generalization of finite automata theory, *J. Comp. System Sci.* **1** (1967), 317–322.

28. J. W. THATCHER AND J. B. WRIGHT, Generalized finite automata theory with an application to a decision problem of second-order logic, *Math. Systems Theory* **2** (1968), 57–81.
29. J. W. THATCHER, Transformations and translations from the point of view of generalized finite automata theory, *Proc. ACM Symp. Theory of Computing* (1969), 129–142.
30. J. W. THATCHER, Two approaches to generalized nondeterministic finite-state transformations, forthcoming, 1970.
31. D. H. YOUNGER, Recognition and parsing of context-free languages in time n^3, *Information and Control* **10** (1967), 189–208.