No. 93-DP-0031

Title

FINITE STATE TRANSLATION SYSTEMS AND PARALLEL MULTIPLE CONTEXT-FREE GRAMMARS

Authors

Yuichi Kaji†,
Hiroyuki Seki†‡, and
Tadao Kasami‡†, members

Address

† Faculty of Engineering Science, Osaka University
Toyonaka, Osaka 560, Japan

‡ Nara Institute of Science and Technology
Takayama, Ikoma, Nara 630-01, Japan

Running head FSTS AND PMCFG

Keywords

finite state translation systems
parallel multiple context-free grammars
tree automata
computational complexity
formal languages

Related areas: **D. Information and Systems**

Automaton, Language and Theory of Computing
Algorithm and Computational Complexity

SUMMARY

Finite state translation systems (fsts') are a widely studied computational model in the area of tree automata theory. In this paper, the string generating capacities of fsts' and their subclasses are studied. First, it is shown that the class of string languages generated by deterministic fsts' equals to that of parallel multiple context-free grammars, which are an extension of context-free grammars. As a corollary, it can be concluded that the recognition problem for a deterministic fsts is solvable in $O(n^{e+1})$-time, where n is the length of an input word and e is a constant called the degree of the deterministic fsts'. In contrast to the latter fact, it is also shown that nondeterministic monadic fsts' with state-bound 2 can generate an $\mathcal{NP}$-complete language.

1. Introduction

Many researchers have investigated the "gap" between context-free languages (cfl's) and context-sensitive languages (csl's). Their studies are motivated by two different interests; an interest from the viewpoint of natural language processing, and an interest from the viewpoint of computational complexity theory.

In the field of natural language processing, it is fundamentally important to propose a well-defined grammatical formalism. It has been often claimed that cfg's do not have enough power to describe the syntax of natural languages; for example, discontinuous phrase structure such as "respectively" sentence cannot be described by cfg's in a simple manner. On the other hand, csg's have too much power for efficient handling. According to these considerations, a number of new grammatical formalisms of which generative power is stronger than that of cfg's have been proposed. These new grammars include head grammars (hg)[10], tree adjoining grammars (tag)[16] and generalized context-free grammars (gcfg's)[10]. Among them, gcfg's are a natural extension of cfg's and phrase structure is simply defined in gcfg's. However, it was shown to have generative power equal to that of type-0 grammars[7] and hence it cannot be handled efficiently.

Parallel multiple context-free grammars (*pmcfg's*) were introduced as a subclass of gcfg's[7]. For each nonterminal symbol A of a pmcfg G, A derives tuples of strings. Languages generated by pmcfg's are called *parallel multiple context-free languages* (*pmcfl's*). *Multiple context-free grammars* (*mcfg's*) are a subclass of pmcfg's and languages generated by mcfg's are called *multiple context-free languages* (*mcfl's*)[7]. *Linear context-free rewriting systems* (*lcfrs'*) introduced by Vijay-Shanker et al.[17] are essentially the same grammatical formalism as mcfg's. It has been shown that the class of languages generated by hg's (tag's) is properly included in the class of mcfl's[17], which in turn is properly included in the class of pmcfl's. The class of pmcfl's is properly included in the class of context-sensitive languages[7] and the former is recognizable in deterministic polynomial time[8].

Let us go back to the gap between cfl's and csl's. It has been known that cfl's can be recognized in deterministic polynomial time, while there is an $\mathcal{NP}$-complete language in csl's[12], and hence one may conjecture that there is a border between $\mathcal{P}$ and $\mathcal{NP}$ in the gap between cfl's and csl's. A number of computational models have been introduced to clarify the computational theoretic hierarchy in this gap. For example, tree automata and their variants, extensions of push down automata, and finite-state translation systems are widely studied models for this purpose.

Finite state translation systems (*fsts'*) were originally introduced as a model of transformational grammars[11]. Later it was found to be an interesting computational model, and properties of fsts' and their subclasses have been extensively investigated[2, 3, 19]. An fsts consists of a *tree transducer* M and a context-free grammar (cfg) G[11, 15]. A tree transducer M takes a tree as an input, starts from the initial state with its head scanning the root node of an input. According to the current state and the label of the scanned node, M transforms an input tree into an output tree in a top-down way. An fsts (M, G) is a tree transducer M with its input domain being the set of derivation trees of the cfg G[11, 15]. The output set of trees is called the *tree language generated by* (M, G), and the *yield language generated by* (M, G) is defined to be the set of strings obtained by concatenating (the labels of) leaves of a tree in the tree language.

As for generative power of fsts', Engelfriet has studied hierarchy of language classes generated by fsts' and their subclasses[3]. He has shown that the generative power of *deterministic fsts'* is properly stronger than that of *finite-copying* fsts', and is properly weaker than that of (nondeterministic) fsts'. He also introduced a class of *monadic* fsts' (ET0L) which has properly weaker generative power than nondeterministic fsts' (see Figure 1). In Ref.[19], it is shown that the class of yield languages generated by finite-copying fsts' equals to the class of languages generated by lcfrs', hence that of mcfl's.

In this paper, it is shown show that the class of languages generated by deterministic fsts' equals to the class of pmcfl's. It is also shown that there is an $\mathcal{NP}$-complete language in the class of string languages generated by nondeterministic monadic fsts' with state-bound 2. By our results, a number of known properties of pmcfl's and mcfl's will be used for the study of fsts' and their string languages, and vice versa. In fact, as a corollary of our results, it can be concluded that the recognition problem for a deterministic fsts is solvable in $O(n^{e+1})$-time, where n is the length of an input word and e is a constant called the degree of the deterministic fsts.

2. Definitions

2.1 Parallel Multiple Context-Free Grammars

A *parallel multiple context-free grammar* (*pmcfg*) is defined to be a 5-tuple $G = (V_N, V_T, F, P, S)$ which satisfies the following conditions (G1) through (G5)[7, 13].

(G1) V_N is a finite set of *nonterminal symbols*, and a positive integer $d(A)$ is given for each nonterminal symbol $A \in V_N$. The *dimension* of G is $\max\{d(A) \mid A \in V_N\}$.

(G2) V_T is a finite set of *terminal symbols* such that $V_N \cap V_T = \phi$.

(G3) F is a finite set of *functions* satisfying the following conditions. For a positive integer d, let $(V_T^*)^d$ denote the set of all the d-tuples of strings over V_T. Let $a(f)$ be the arity of $f \in F$. For each $f \in F$, positive integers $d_i(f)$ $(1 \leq i \leq a(f))$ and $r(f)$ are given, and f is a total function from $(V_T^*)^{d_1(f)} \times (V_T^*)^{d_2(f)} \times \cdots \times (V_T^*)^{d_{a(f)}(f)}$ to $(V_T^*)^{r(f)}$ which satisfies the following condition (f1). Let

$$\bar{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id_i(f)})$$

denote the ith argument of f for $1 \leq i \leq a(f)$.

(f1) For $1 \leq h \leq r(f)$, the hth component of f, denoted by $f^{[h]}$ is defined by a concatenation of some terminal strings in V_T^* and some components of arguments. That is, a nonnegative integer n_h is defines and

$$\begin{aligned} & f^{[h]}[\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{a(f)}] = \\ & \qquad w_{h,1} x_{\mu(h,1)\eta(h,1)} w_{h,2} \cdots w_{h,n_h} x_{\mu(h,n_h)\eta(h,n_h)} w_{h,n_h+1} \end{aligned} \qquad (1)$$

where $w_{h,k} \in V_T^*$ for $1 \leq k \leq n_h + 1$, $1 \leq \mu(h,j) \leq a(f)$ and $1 \leq \eta(h,j) \leq d_{\mu(h,j)}(f)$ for $1 \leq j \leq n_h$.

(G4) P is a finite set of *productions* of the form $A \to f[A_1, A_2, \ldots, A_{a(f)}]$ where $A, A_1, A_2, \ldots, A_{a(f)} \in V_N$, $f \in F, r(f) = d(A)$ and $d_i(f) = d(A_i)$ $(1 \leq i \leq a(f))$. If $a(f) = 0$, then f has no argument and $f[]$ equals to a tuple of strings over V_T. A production with a function f such that $a(f) = 0$ is called a *terminating production*, otherwise it is called a *nonterminating production*. A terminating production $A \to f[]$ is written as $A \to f$.

(G5) $S \in V_N$ is the *initial symbol*, and $d(S) = 1$.

If all the functions of a pmcfg G satisfy the following condition (f2), then G is called a *multiple context-free grammar* (*mcfg*).

(f2) For each component x_{ij} in the arguments, the total number of occurrences of x_{ij} in the right-hand sides of (1) from $h = 1$ through $r(f)$ is at most one.

If some variable occurs more than once in the right-hand side of the definition of f, the string substituted for the variable will be copied more than once. It has been shown that such copy operations increase the generative capacity of grammars[7] (see Example 2.2). Condition (f2) inhibits these copy operations.

The language generated by a pmcfg $G = (V_N, V_T, F, P, S)$ is defined as follows [1]. For $A \in V_N$, let us define $L_G(A)$ as the smallest set satisfying the following two conditions:

(L1) If a terminating production $A \to f$ with $f = \bar{\alpha} \in (V_T^*)^{d(A)}$ is in P, then $\bar{\alpha} \in L_G(A)$.

(L2) If $A \to f[A_1, A_2, \ldots, A_{a(f)}] \in P$ and $\bar{\alpha}_i \in L_G(A_i)$ $(1 \leq i \leq a(f))$, then $\bar{\alpha} = f[\bar{\alpha}_1, \bar{\alpha}_2, \ldots, \bar{\alpha}_{a(f)}] \in L_G(A)$. We say that $A \to f[A_1, A_2, \ldots, A_{a(f)}]$ is *the last production applied to obtain* $\bar{\alpha}$.

Define $L(G) \triangleq L_G(S)$. $L(G)$ is called the *parallel multiple context-free language (pmcfl) generated by* G. If G is an mcfg, $L(G)$ is called the *multiple context-free language (mcfl) generated by* G. Let PMCFL and MCFL denote the class of all pmcfl's and that of all mcfl's, respectively.

Example 2.1: Let $G_1 = (V_N, V_T, F, P, S)$ where $V_N = \{A, B, S\}$ $(d(A) = d(B) = 2, d(S) = 1)$, $V_T = \{a, b, c, d\}$, $F = \{f_\varepsilon, f_1, f_2, g\}$ and the productions in P be:

$$
\begin{array}{lllll}
r_1 & S & \to & g[A, B] & \text{where } g[(x_1, x_2), (y_1, y_2)] = x_1 y_1 x_2 y_2 \\
r_2 & A & \to & f_1[A] & \text{where } f_1[(x_1, x_2)] = (a x_1, c x_2) \\
r_3 & A & \to & f_\varepsilon & \text{where } f_\varepsilon = (\varepsilon, \varepsilon) \\
r_4 & B & \to & f_2[B] & \text{where } f_2[(x_1, x_2)] = (b x_1, d x_2) \\
r_5 & B & \to & f_\varepsilon &
\end{array}
$$

[1] Derivation of pmcfg's can be defined as rewriting steps of a sentential form[4, 5]. However, for $(\alpha_1, \ldots, \alpha_n) \in L_G(A)$, α_i's do not always appear consecutively in a sentential form, and hence this simple form of definition is used in this paper.

G_1 is an mcfg with dimension 2. The language generated by G_1 is defined as follows. By the rule r_3, $(\varepsilon, \varepsilon) \in L_{G_1}(A)$. By substituting ε's for x_1 and x_2 in r_2, $(a, c) \in L_{G_1}(A)$. By applying r_2 repeatedly, $(a^m, c^m) \in L_{G_1}(A)$ for $m > 0$. Similarly, $(b^n, d^n) \in L_{G_1}(B)$ for $n \geq 0$. $L_{G_1}(S) = \{a^m b^n c^m d^n | m, n \geq 0\}$ and this is the language generated by G_1. □

Example 2.2[7]**:** Let $G_2 = (V_N, V_T, F, P, S)$ where $V_N = \{S\}$ ($d(S) = 1$), $V_T = \{a\}$, $F = \{f_a, f\}$, $P = \{r_1 : S \to f_a,\ r_0 : S \to f[S]\}$, where $f_a = a$,$f[(x)] = xx$. G_2 is a pmcfg with dimension 1 but is not an mcfg since the function f does not satisfy the condition (f2). The language generated by G_2 is $\{a^{2^n} | n \geq 0\}$, which cannot be generated by any mcfg (see Lemma 6 of Ref.[7]). □

Lemma 2.1[7]**:** For a given pmcfg G (resp. mcfg G), we can construct an pmcfg G' (resp. mcfg G') which satisfies $L(G') = L(G)$ and the following *non-erasing condition* (f3).

(f3) For each function f of G', each variable x_{ij} appears at least once in the right-hand side of (1) for some h $(1 \leq h \leq r(f))$.

Sketch of Proof: The idea behind the construction is similar to that of ε-rule elimination procedure of a context-free grammar. For example, assume that there is a production $A \to f[B_1, \ldots, B_n]$ and x_{ij} does not appear in the right-hand side of (1). Then a new nonterminal B_i' with $d(B_i') = d(B_i) - 1$ is introduced, and this production is replaced by $A \to f'[B_1, \ldots, B_i', \ldots, B_n]$ where f' is identical to f except that the dimension of the ith argument is smaller by one than f. Furthermore, for each production whose left-hand side is B_i, add a new production whose left-hand side is B_i' and whose function in the right-hand side is defined by deleting jth component of the original one. For the formal proof, see Lemma 1 of Ref.[7]. □

2.2 Finite State Translation Systems

A set Σ of symbols is a *ranked alphabet* if, for each $\sigma \in \Sigma$, a unique non-negative number $\rho(\sigma)$ which is called the *rank* of σ is associated. Define $\mathcal{T}_\Sigma$ as the smallest set such that;

- If $\rho(\sigma) = 0$ for $\sigma \in \Sigma$, then $\sigma \in \mathcal{T}_\Sigma$.
- If $\rho(\sigma) = n \ (\geq 1)$ for $\sigma \in \Sigma$ and $t_1, \ldots, t_n \in \mathcal{T}_\Sigma$, then $t = \sigma(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma$. σ is called the *root symbol*, or shortly, the *root* of t.

Hereafter, a term in $\mathcal{T}_\Sigma$ may be called a *tree*.

Let $G = (V_N, V_T, P, S)$ be a context-free grammar (cfg) where V_N, V_T, P and S are a set of *nonterminal symbols*, a set of *terminal symbols*, a set of *productions* and the *initial symbol*, respectively. A *derivation tree of the cfg* G is a term defined as follows.

(T1) For every $a \in V_T$, a is a derivation tree of G.

(T2) Assume that there are a production $r : A \to X_1 \cdots X_n$ ($A \in V_N, X_1, \ldots, X_n \in V_N \cup V_T$) in P where r is the label of this production, and n derivation trees $t_1, \ldots t_n$ whose roots are labeled with $r_1, \ldots, r_n$, respectively, and

- if $X_i \in V_N$, then r_i ($1 \leq i \leq n$) is the label of a production $r_i : X_i \to \cdots$, whose left-hand side is X_i, and
- if $X_i \in V_T$, then $r_i = t_i = X_i$.

Then $r(t_1, \ldots, t_n)$ is a derivation tree of G.

(T3) There are no other derivation trees.

Let $\mathcal{R}(G)$ be the set of derivation trees whose root is the label of a production of which the left-hand side is the initial symbol S. Remark that if we take $\Sigma = \{$the labels of productions in $P\} \cup V_T$, and define

$\rho(r) = n$ for $r : A \to X_1 \cdots X_n \in P$ and $\rho(a) = 0$ for $a \in V_T$, then Σ is a ranked alphabet and $\mathcal{R}(G) \subseteq \mathcal{T}_\Sigma$.

A *tree transducer* is defined in Ref.[11] as a generalization of a generalized sequential machine, and it defines a mapping from trees to trees. But in this paper, since we are mainly interested in a string language generated by it, a "tree-to-string" version of transducer defined in Ref.[3] is reviewed. For sets Q and X, let

$$Q[X] \triangleq \{q[x] \mid q \in Q, x \in X\}.$$

A *tree-to-string transducer* (yT*-transducer* or simply *transducer*) is defined to be a 5-tuple $M = (Q, \Sigma, \Delta, q_0, R)$ where

- Q is a finite set of *states*,
- Σ is an *input ranked alphabet*,
- Δ is an *output alphabet*,
- $q_0 \in Q$ is the *initial state*, and
- R is a set of *rules* of the form

$$q[\sigma(x_1, \ldots, x_n)] \to v$$

  where $q \in Q, \sigma \in \Sigma, \rho(\sigma) = n$ and $v \in (\Delta \cup Q[\{x_1, \ldots, x_n\}])^*$.

If different rules in R have different left-hand sides, then M is called *deterministic*[3].

A *configuration* of a yT-transducer is an element in $(\Delta \cup Q[\mathcal{T}_\Sigma])^*$. *Derivation* of M is defined as follows. Let $c = \alpha_1 q[\sigma(t_1, \ldots, t_n)]\alpha_2$ be a configuration where $\alpha_1, \alpha_2 \in (\Delta \cup Q[\mathcal{T}_\Sigma])^*$, $q \in Q$, $\sigma \in \Sigma$, $\rho(\sigma) = n$ and $t_1, \ldots, t_n \in \mathcal{T}_\Sigma$. Assume that there is a rule $q[\sigma(x_1, \ldots, x_n)] \to v$ in R, and v' can be obtained from v by substituting $t_1, \ldots, t_n$ for $x_1, \ldots, x_n$, respectively, then $c \Rightarrow \alpha_1 v' \alpha_2$. Let $\stackrel{*}{\Rightarrow}$ be reflexive and transitive closure of $\Rightarrow$. For configurations c and c', if $c \stackrel{*}{\Rightarrow} c'$, then c *derives* c'. If there is no $c' \in \Delta^*$ such that $c \overset{*}{\Rightarrow} c'$, then c *derives no output*. For example, if there is no rule whose left-hand side is $q[\sigma(x_1, \ldots, x_n)]$, then $c = \alpha_1 q[\sigma(t_1, \ldots, t_n)]\alpha_2$ derives no output.

Example 2.3[11]**:** Let $M = (Q, \Sigma, \Delta, q_d, R)$ be a yT-transducer where

$$\begin{aligned} Q &= \{q_d, q_i\} \\ \Sigma &= \{c, y, +, \cdot\} \qquad (\rho(c) = \rho(y) = 0, \rho(+) = \rho(\cdot) = 2) \\ \Delta &= \Sigma \cup \{0, 1\} \end{aligned}$$

and the rules in R are:

$$\begin{aligned} q_i[c] \to c & \qquad q_i[y] \to y \\ q_i[+(x_1, x_2)] &\to q_i[x_1] + q_i[x_2] \\ q_i[\cdot(x_1, x_2)] &\to q_i[x_1] \cdot q_i[x_2] \\ q_d[c] \to 0 & \qquad q_d[y] \to 1 \\ q_d[+(x_1, x_2)] &\to q_d[x_1] + q_d[x_2] \\ q_d[\cdot(x_1, x_2)] &\to q_d[x_1] \cdot q_i[x_2] + q_i[x_1] \cdot q_d[x_2]. \end{aligned}$$

Intuitively, an element in $\mathcal{T}_\Sigma$ represents an arithmetic expression, and state q_d and q_i represent "differential" and "identity", respectively. Let $t = q_d[\cdot(y, +(c, y))]$ and $t' = q_d[y] \cdot q_i[+(c, y)] + q_i[y] \cdot q_d[+(c, y)]$, then $t \Rightarrow t'$, which corresponds to $\frac{d}{dy}(y \cdot (c + y)) = \frac{d}{dy} y \cdot (c + y) + y \cdot \frac{d}{dy}(c + y)$.

$\square$

A *tree-to-string finite state translation system* (yT*-fsts*, or *fsts* for short) is defined by a yT-transducer M and a cfg G, written as (M, G). (NOTE: In Ref.[11], a yT-fsts is defined by a yT-transducer and a *recognizable set of trees*. In Ref.[15], it is shown that the class of recognizable sets of trees is equal to the class of sets of derivation trees of cfg's. Hence a yT-fsts is defined by a yT-transducer and a cfg in this paper.)

Define $yL(M, G)$, called the *yield language generated by a* yT*-fsts* (M, G), as

$$yL(M, G) \triangleq \{t \in \Delta^* \mid \exists t' \in \mathcal{R}(G), q_0[t'] \overset{*}{\Rightarrow} t\}$$

where Δ is an output alphabet and q_0 is the initial state of M. Note that $\mathcal{R}(G)$ is a set of derivation trees of the cfg G and hence recognizable set of trees. An fsts is called *deterministic*[3] if the transducer M is deterministic. We use a terminology "*nondeterministic*" when we emphasize that we don't assume determinism of the transducer.

Next, a *state-bound* of fsts and *finite-copying* fsts'[3] are defined. Let (M, G) be an fsts with an output alphabet Δ and an initial state q_0. Let $t \in \mathcal{R}(G)$ and consider a derivation $\alpha : q_0[t] \stackrel{*}{\Rightarrow} w \in \Delta^*$. Let t' be a subtree of t. Now, delete from the original derivation α all the derivation steps which operates on t'. This leads to the following new derivation which keeps t' untouched:

$$\alpha' : q_0[t] \stackrel{*}{\Rightarrow} w_1 q_{i_1}[t'] w_2 \cdots w_n q_{i_n}[t'] w_{n+1}$$

where $w_i \in \Delta^*$ $(1 \leq i \leq n+1)$.

The *state sequence of* t' *in derivation* α is defined to be $\langle q_{i_1}, \ldots, q_{i_n} \rangle$. The derivation α has a *state-bound* s if, for each subtree of t, the number of different states in the state sequence is at most s. α has a *copying-bound* k if, for each subtree of t, the length of its state sequence is at most k. An fsts (M, G) has a *state-bound* s if for each $w \in yL(M, G)$, there is a derivation tree $t \in \mathcal{R}(G)$ such that the derivation $q_0[t] \stackrel{*}{\Rightarrow} w$ has a state-bound s. An fsts (M, G) is a *finite-copying fsts* if there is a constant k such that for each $w \in yL(M, G)$, there is a derivation tree $t \in \mathcal{R}(G)$ such that the derivation $q_0[t] \stackrel{*}{\Rightarrow} w$ has a copying-bound k.

An fsts (M, G) whose second component G is a regular grammar is called an ET0L system (see Ref.[3]). In this paper, we say a *monadic fsts* for an ET0L system.

Figure 1 shows relationship among the generative power of subclasses of fsts'. In the figure, d-fsts, fc-fsts and m-fsts denote the classes of deterministic fsts', finite-copying fsts' and monadic fsts', respectively, and fsts$_s$, d-fsts$_s$ and m-fsts$_s$ denote the classes of each fsts' with state-bound s, respectively. For finite-copying fsts', the subscript denotes their copying-bound. An arrow from a class A to another class B means that A has properly stronger power than B.

3. Generative Power of Deterministic FSTS'

In this section, we show that $yL(\text{DFSTS})$, the class of yield languages generated by deterministic fsts', equals to PMCFL. First we show that $yL(\text{DFSTS}) \subseteq \text{PMCFL}$. A part of the proof of $\text{PMCFL} \subseteq yL(\text{DFSTS})$ is stated in the appendix since the idea behind the proof is similar to that of $yL(\text{DFSTS}) \subseteq \text{PMCFL}$.

3.1 $yL(\textbf{DFSTS}) \subseteq \textbf{PMCFL}$

Let (M, G) be a deterministic yT-fsts where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G = (V_N, V_T, P, S)$. We assume that $Q = \{q_1, \ldots, q_\ell\}$, $V_T = \{a_1, \ldots, a_n\}$ and the productions in P are labeled with $r_1, \ldots, r_m$. Since the input domain of M is the set of derivation trees of G, we assume that $\Sigma = \{r_1, \ldots, r_m, a_1, \ldots, a_n\}$ without loss of generality.

A pmcfg $G' = (V'_N, V'_T, F', P', S')$ such that $yL(M, G) = L(G') \cap \Delta^*$ is constructed as follows. Let $V'_T = \Delta \cup \{b\}$ where b is a newly introduced symbol and let

$$V'_N = \{S', R_1, \ldots, R_m, A_1, \ldots, A_n\}$$

where $d(R_i) = d(A_j) = \ell$ for $1 \leq i \leq m$ and $1 \leq j \leq n$. Note that each R_i $(1 \leq i \leq m)$ and A_j $(1 \leq j \leq n)$ correspond to production r_i and terminal a_j of cfg G, respectively. Productions and functions of G' will be constructed to have the following property.

Property 3.1: There is $(\alpha_1, \ldots, \alpha_\ell) \in L_{G'}(R_h)$ (resp. $L_{G'}(A_h)$) such that

$$\begin{cases} \text{each of } \alpha_{s_1}, \ldots, \alpha_{s_u} \text{ does not contain } b \text{, and} \\ \text{each of the remaining } \alpha_{t_1}, \ldots, \alpha_{t_v} \text{ contains } b \end{cases}$$

if and only if there is a derivation tree t of G such that the root of t is r_h (resp. a_h) and

$$\begin{cases} q_{s_p}[t] \stackrel{*}{\Rightarrow} \alpha_{s_p} & (1 \leq p \leq u) \\ q_{t_p}[t] \text{ derives no output} & (1 \leq p \leq v). \end{cases}$$

$\Box$

The basic idea is to simulate the move of tree transducer M which is scanning a symbol r_h (resp. a_h) with state q_i by the ith component of the nonterminal R_h (resp. A_h) of pmcfg G'. During the move of M, it may happen that no rule is defined for a current configuration and hence no output will be derived. The symbol b is introduced to represent such an undefined move explicitly.

To construct productions and functions, Define RS(X) ($X \in V_N \cup V_T$) as follows.

$$\mathrm{RS}(X) = \begin{cases} \{R_h \mid \text{the left-hand side of } r_h \text{ is } X\} & \text{if } X \in V_N \\ \{A_h\} & \text{if } X = a_h \in V_T. \end{cases}$$

Productions and functions are defined as follows.

Step 1: For each production $r_h : Y_0 \to Y_1 \cdots Y_k$ ($Y_0 \in V_N, Y_u \in V_N \cup V_T$ for $1 \leq u \leq k$) of cfg G, construct nonterminating productions

$$R_h \to f_{r_h}[Z_1, \ldots, Z_k]$$

for arbitrary combinations of $Z_u \in \mathrm{RS}(Y_u)$ ($1 \leq u \leq k$) where f_{r_h} is defined as follows: For $1 \leq i \leq \ell$,

- if there is no rule whose left-hand side is $q_i[r_h(x_1, \ldots, x_k)]$, then

$$f_{r_h}^{[i]}[\bar{x}_1, \ldots, \bar{x}_k] \triangleq b, \tag{2}$$

- if the transducer M has a rule $q_i[r_h(x_1, \ldots, x_k)] \to w_{i,1}$ $q_{\eta(i,1)}[x_{\mu(i,1)}]w_{i,2} \cdots w_{i,n_i} q_{\eta(i,n_i)}[x_{\mu(i,n_i)}]w_{i,n_i+1}$, then

$$f_{r_h}^{[i]}[\bar{x}_1, \ldots, \bar{x}_k] \triangleq w_{i,1} x_{\mu(i,1)\eta(i,1)} w_{i,2} \cdots w_{i,n_i} x_{\mu(i,n_i)\eta(i,n_i)} w_{i,n_i+1} \tag{3}$$

where $\bar{x}_u = (x_{u1}, \ldots, x_{u\ell})$ $(1 \leq u \leq k)$.

(Since M is deterministic, there exists at most one rule whose left-hand side is $q_i[r_h(\cdots)]$ and hence the above construction is consistent.)

Step 2: For each $a_h \in V_T$, construct a terminating production $A_h \to f_{a_h}$ where f_{a_h} is defined as follows: For $1 \leq i \leq \ell$,

- if there is no rule whose left-hand side is $q_i[a_h]$, then $f_{a_h}^{[i]} \triangleq b$.
- if $q_i[a_h] \to w_i$, then $f_{a_h}^{[i]} \triangleq w_i$.

Step 3: For each $R_h \in \mathrm{RS}(S)$, construct $S' \to f_{\mathrm{first}}[R_h]$ where $f_{\mathrm{first}}[(x_1, \ldots, x_\ell)] \triangleq x_1$. Intuitively, the right-hand side of this production corresponds to the configuration that M is in an initial state q_1 and scanning the root symbol r_h of a derivation tree, where r_h is the label of a production of G whose left-hand side is the initial symbol S.

In the following, it is shown that the pmcfg G' defined as above has Property 3.1.

(Only if part) It is shown by induction on the number of applications of (L1) and (L2) in section 2 to obtain a tuple of strings $(\alpha_1, \ldots, \alpha_\ell)$. For the basis, assume that $\bar{\alpha} = (\alpha_1, \ldots, \alpha_\ell) \in L_{G'}(X)$ is obtained by only one application of (L1). It is clear that the applied (terminating) production is constructed in Step 2, and hence there is some h such that $X = A_h$, $A_h \to f_{a_h}$ and $f_{a_h} = \bar{\alpha}$. Let $t = a_h$ and consider how derivations proceed from $q_i[t]$ for $1 \leq i \leq \ell$. If $\alpha_i = b$ then $f_{a_h}^{[i]} = b$ and hence there should be no rule whose left-hand side is $q_i[a_h]$. If α_i does not contain b, then transducer M has a rule $q_i[a_h] \to \alpha_i$, and the property holds.

Assume that the property holds for every tuple of strings which can be obtained by d' applications or less, and suppose the case that

$(\alpha_1, \ldots, \alpha_\ell) \in L_{G'}(X)$ is obtained by $d'+1$ applications. The last (non-terminating) production applied in (L2) must be constructed in Step 1, hence there is some h such that $X = R_h$, and the applied production is

$$R_h \to f_{r_h}[Z_1, \ldots, Z_k]. \tag{4}$$

Furthermore, there exist $\bar{\beta}_u = (\beta_{u1}, \ldots, \beta_{u\ell}) \in L_{G'}(Z_u)$ for $1 \le u \le k$ such that $(\alpha_1, \ldots, \alpha_\ell) = f_{r_h}[\bar{\beta}_1, \ldots, \bar{\beta}_k]$. For each u $(1 \le u \le k)$, if $Z_u = R_{h_u}$ for some h_u (resp. $Z_u = A_{h_u}$ for some h_u), then $\bar{\beta}_u \in L_{G'}(R_{h_u})$ (resp. $\bar{\beta}_u \in L_{G'}(A_{h_u})$), and by the inductive hypothesis there is a derivation tree t_u which satisfies Property 3.1 with $\bar{\beta}_u$. That is, the root of t_u is r_{h_u} (resp. a_{h_u}), and for v $(1 \le v \le \ell)$,

$$\begin{cases} q_v[t_u] \overset{*}{\Rightarrow} \beta_{uv} & \text{if } \beta_{uv} \text{ does not contain } b, \\ q_v[t_u] \text{ derives no output} & \text{if } \beta_{uv} \text{ contains } b. \end{cases} \tag{5}$$

We note that since (4) is constructed in Step 1 as a production of pmcfg G', cfg G has a production $r_h : Y_0 \to Y_1 \cdots Y_k$ and $Z_u \in \mathrm{RS}(Y_u)$ holds for $1 \le u \le k$. Now, $Z_u = R_{h_u} \in \mathrm{RS}(Y_u)$ (resp. $Z_u = A_{h_u} \in \mathrm{RS}(Y_u)$) holds and it follows that the left-hand side of production r_h is Y_u by the definition of RS (resp. Y_u is the terminal symbol a_{h_u}). Hence, if we take $t = r_h(t_1, \ldots, t_k)$ then t is a derivation tree of cfg G. Now, consider a derivation of M from $q_i[t]$ for $1 \le i \le \ell$.

- If α_i contains b, then there are two cases.
  - $f^{[i]}_{r_h}$ is defined to be b by (2). In this case, there exists no rule whose left-hand side is $q_i[r_h(x_1, \ldots, x_k)]$. Hence $q_i[t]$ derives no output and the property holds.
  - $f^{[i]}_{r_h}$ is defined by (3). In this case, α_i can be written as $\alpha_i = w_{i,1}\beta_{\mu(i,1)\eta(i,1)}w_{i,2}\cdots w_{i,n_i}\beta_{\mu(i,n_i)\eta(i,n_i)}w_{i,n_i+1}$ and $\beta_{\mu(i,j)\eta(i,j)}$ contains b for some j $(1 \le j \le n_i)$. By the construction of function f_{r_h} in Step 1, there is a derivation from $q_i[t]$;

    $$q_i[t] = q_i[r_h(t_1, \ldots, t_k)] \Rightarrow$$

$$w_{i,1}q_{\eta(i,1)}[t_{\mu(i,1)}]w_{i,2}\cdots w_{i,n_i}q_{\eta(i,n_i)}[t_{\mu(i,n_i)}]w_{i,n_i+1}$$

and there are no other derivation since M is deterministic. If $\beta_{\mu(i,j)\eta(i,j)}$ contains b, then by (5), $q_{\eta(i,j)}[t_{\mu(i,j)}]$ derives no output and hence $q_i[t]$ also cannot derive output.

- If α_i does not contain b, then $f^{[i]}_{r_h}$ is defined by (3), α_i can be written as $\alpha_i = w_{i,1}\beta_{\mu(i,1)\eta(i,1)}w_{i,2}\cdots w_{i,n_i}\beta_{\mu(i,n_i)\eta(i,n_i)}w_{i,n_i+1}$ and each $\beta_{\mu(i,j)\eta(i,j)}$ $(1 \leq j \leq n_i)$ does not contain b. By (5), $q_{\eta(i,j)}[t_{\mu(i,j)}] \overset{*}{\Rightarrow} \beta_{\mu(i,j)\eta(i,j)}$ holds for $1 \leq j \leq n_i$, hence

$$\begin{aligned} q_i[t] &= q_i[r_h(t_1,\ldots,t_k)] \\ &\Rightarrow w_{i,1}q_{\eta(i,1)}[t_{\mu(i,1)}]w_{i,2}\cdots w_{i,n_i}q_{\eta(i,n_i)}[t_{\mu(i,n_i)}]w_{i,n_i+1} \\ &\overset{*}{\Rightarrow} w_{i,1}\beta_{\mu(i,1)\eta(i,1)}w_{i,2}\cdots w_{i,n_i}\beta_{\mu(i,n_i)\eta(i,n_i)}w_{i,n_i+1} \\ &= \alpha_i \end{aligned}$$

  and the property holds.

<u>*(If part)*</u> If part is shown by induction on the size of a derivation tree t of G. (The *size of a tree t* is the number of occurrences of symbols of the ranked alphabet appearing in t.) For the basis, assume that the size of t is one, that is, $t = a_h$ for some $a_h \in V_T$. By Step 2, there is a production $A_h \to f_{a_h}$ and the property holds.

For the inductive step, assume that the property holds for every derivation tree whose size is not greater than d', and consider a derivation tree $t = r_h(t_1,\ldots,t_k)$ with size $d'+1$. Since t is a derivation tree of cfg G, r_h is a production of the form $Y_0 \to Y_1\cdots Y_k$ and the root of t_u $(1 \leq u \leq k)$ is;

$$\begin{cases} r_{h_u} \text{ (label of a production whose left-hand side is } Y_u) & \text{if } Y_u \in V_N \\ a_{h_u} & \text{if } Y_u = a_{h_u} \in V_T. \end{cases}$$

By the definition of RS, $R_{h_u} \in \mathrm{RS}(Y_u)$ (or $A_{h_u} \in \mathrm{RS}(Y_u)$) holds for $1 \leq u \leq k$, and hence pmcfg G' has a production $R_h \to f_{r_h}[Z_1,\ldots,Z_k]$

where $Z_u = R_{h_u}$ (or $Z_u = A_{h_u}$). (See the construction of productions in Step 1.)

Here, the size of each subtree t_u $(1 \le u \le k)$ equals to or less than d', by the inductive hypothesis, there exist $\bar{\beta}_u = (\beta_{u1}, \ldots, \beta_{u\ell}) \in L_{G'}(R_{h_u})$ (or $L_{G'}(A_{h_u})$) such that $\bar{\beta}_u$ and t_u satisfy Property 3.1. That is, for v $(1 \le v \le \ell)$,

$$\begin{cases} \beta_{uv} \text{ does not contain } b & \text{if } q_v[t_u] \overset{*}{\Rightarrow} \beta_{uv}, \\ \beta_{uv} \text{ contains } b & \text{if } q_v[t_u] \text{ derives no output.} \end{cases} \tag{6}$$

Now, let

$$\bar{\alpha} = (\alpha_1, \ldots, \alpha_\ell) = f_{r_h}[\bar{\beta}_1, \ldots, \bar{\beta}_k] \in L_{G'}(R_h)$$

and consider how α_i is defined for $1 \le i \le \ell$.

- If there is no rule whose left-hand side is $q_i[r_h(x_1, \ldots, x_k)]$, then $q_i[t]$ derives no output. In this case, $f_{r_h}^{[i]}$ is defined to be b and hence $\alpha_i = b$, the property holds.

- If the transducer M has a rule $q_i[r_h(x_1, \ldots, x_k)] \to w_{i,1} q_{\eta(i,1)}[x_{\mu(i,1)}]$ $w_{i,2} \cdots w_{i,n_i} q_{\eta(i,n_i)}[x_{\mu(i,n_i)}] w_{i,n_i+1}$, then we can write α_i as

$$\alpha_i = w_{i,1} \beta_{\mu(i,1)\eta(i,1)} w_{i,2} \cdots w_{i,n_i} \beta_{\mu(i,n_i)\eta(i,n_i)} w_{i,n_i+1} \tag{7}$$

  by the construction of functions in Step 1. There are two cases:

  - For some j $(1 \le i \le n_i)$, $q_{\eta(i,j)}[t_{\mu(i,j)}]$ derives no output and hence $q_i[t]$ also. In this case, $\beta_{\mu(i,j)\eta(i,j)}$ contains b by (6) and it follows from (7) that α_i also contains b, the property holds.

  - For every j $(1 \le j \le n_i)$, $q_{\eta(i,j)}[t_{\mu(i,j)}]$ derives some output. Since M is deterministic, and by (6), the derived string should be $\beta_{\mu(i,j)\eta(i,j)}$ which does not contain b.

$$\begin{aligned} q_i[t] &= q_i[r_h(t_1, \ldots, t_k)] \\ &\Rightarrow w_{i,1} q_{\eta(i,1)}[t_{\mu(i,1)}] w_{i,2} \cdots w_{i,n_i} q_{\eta(i,n_i)}[t_{\mu(i,n_i)}] w_{i,n_i+1} \\ &\overset{*}{\Rightarrow} w_{i,1} \beta_{\mu(i,1)\eta(i,1)} w_{i,2} \cdots w_{i,n_i} \beta_{\mu(i,n_i)\eta(i,n_i)} w_{i,n_i+1} \\ &= \alpha_i \end{aligned}$$

and the property holds.

The proof of if part is completed and Property 3.1 has been proved.

∎

Lemma 3.1: $yL(\text{DFSTS}) \subseteq \text{PMCFL}$.

Proof. Let $L_1 \triangleq L(G') \cap \Delta^*$. Since pmcfl's are closed under intersection with a regular set[7], L_1 is also a pmcfl. We show that $yL(M, G) = L_1$. By Property 3.1 and the productions constructed in Step 3, $w \in L_1$ if and only if there is a derivation tree t of G such that

- the root of t is r_h,
- the left-hand side of r_h is the initial symbol S, and
- $q_1[t] \overset{*}{\Rightarrow} w$

and the lemma holds. ∎

3.2 PMCFL $\subseteq yL($DFSTS$)$

Let $G = (V_N, V_T, F, P, S)$ be a pmcfg with dimension ℓ. Without loss of generality, G is assumed to satisfy the non-erasing condition of Lemma 2.1. Also suppose that the nonterminating productions of G are labeled with $r_1, \ldots, r_m$, and the terminating productions are labeled with $r'_1, \ldots, r'_n$. Furthermore, for each nonterminal production r_h $(1 \leq h \leq m)$, we suppose that the function of the right-hand side of rule r_h is f_h (the suffix of the function is identical to that of the production), hence each nonterminal production can be written as $r_h : Y_0 \rightarrow f_h[Y_1, \ldots, Y_k]$ $(a(f_h) = k, Y_0, \ldots, Y_k \in V_N)$. We also suppose that each terminating production can be written as $r'_h : Y_0 \rightarrow f'_h$. A yT-fsts (M, G') such that $yL(M, G') = L(G)$ is constructed as follows.

First, define a cfg $G' = (V'_N, V'_T, P', S')$ with $V'_N \triangleq \{S', R_1, \ldots, R_m\}$ and $V'_T \triangleq \{a_1, \ldots, a_n\}$. Note that each nonterminal R_i $(1 \leq i \leq m)$

and terminal a_j $(1 \leq j \leq n)$ of cfg G' correspond to nonterminating production and terminating production of pmcfg G, respectively. To construct productions, $\mathrm{RS}'(X) \subseteq V'_N \cup V'_T$ for $X \in V_N$ is defined as follows.

$$\begin{aligned} \mathrm{RS}'(X) &= \{R_h \mid \text{the left-hand side of } r_h \text{ is } X\} \\ &\cup \{a_h \mid \text{the left-hand side of } r'_h \text{ is } X\}. \end{aligned}$$

By using RS′, productions P' of cfg G' are defined as follows.

Step A: For each nonterminating production $r_h : Y_0 \to f_h[Y_1, \ldots, Y_k]$ of pmcfg G, construct productions

$$\hat{r}_{hZ_1\cdots Z_k} : R_h \to Z_1 \cdots Z_k \tag{8}$$

for $Z_u \in \mathrm{RS}'(Y_u)$ $(1 \leq u \leq k)$.

Step B: For each $Z \in \mathrm{RS}'(S)$, construct

$$\hat{r}_{\mathrm{start}} : S' \to Z. \tag{9}$$

Note that each element in $\mathrm{RS}'(S)$ corresponds to the production of pmcfg G whose left-hand side is the initial symbol S of G.

Define $\Sigma \triangleq \{\text{the labels of productions in } P'\} \cup V'_T$, $\rho(\hat{r}_{hZ_1\cdots Z_k}) = k$ for $\hat{r}_{hZ_1\cdots Z_k} : R_h \to Z_1 \cdots Z_k$, $\rho(a_h) = 1$ for $a_h \in V'_T$ and $\rho(\hat{r}_{\mathrm{start}}) = 1$, then Σ is a ranked alphabet.

Next, we define yT-transducer $M = (Q, \Sigma, \Delta, q_1, R)$ with Σ defined above and $\Delta \triangleq V_T$. Q is defined to be $\{q_1, \ldots, q_\ell\}$ (note that ℓ is the dimension of G).

The rules in R will be defined to have the following property.

Property 3.2: There is $\bar{\alpha} = (\alpha_1, \ldots, \alpha_s) \in L_G(X)$ and the last production applied to obtain $\bar{\alpha}$ is $r_h : X \to f_h[Y_1, \ldots, Y_k]$ (resp. $r'_h : X \to f'_h$) if and only if there is a derivation tree t of G' such that the root is

$\hat{r}_{hZ_1\cdots Z_k}$: $R_h \to Z_1 \cdots Z_k$ ($Z_u \in \mathrm{RS}'(Y_u), 1 \le u \le k$) (resp. terminal symbol a_h) and $q_i[t] \stackrel{*}{\Rightarrow} \alpha_i$ for $1 \le i \le s$. ($q_i[t]$ derives no output for $i > s$.) $\square$

Intuitively saying, a derivation tree of cfg G' represents how to apply productions to obtain tuple of string. The rules of transducer M are constructed to "expand" the tree into string. The rules in R are defined as follows.

Step I: For each nonterminating production $r_h : Y_0 \to f_h[Y_1, \ldots, Y_k]$ with f_h defined as

$$f_h^{[i]}[\bar{x}_1, \ldots, \bar{x}_k] = w_{i,1} x_{\mu(i,1)\eta(i,1)} w_{i,2} \cdots w_{i,n_i} x_{\mu(i,n_i)\eta(i,n_i)} w_{i,n_i+1}$$

where $\bar{x}_u = (x_{u1}, \ldots, x_{ud(Y_u)})$ $(1 \le u \le k)$, define rules

$$q_i[\hat{r}_{hZ_1\cdots Z_k}(x_1, \ldots, x_k)] \to \qquad (10)$$
$$w_{i,1} q_{\eta(i,1)}[x_{\mu(i,1)}] w_{i,2} \cdots w_{i,n_i} q_{\eta(i,n_i)}[x_{\mu(i,n_i)}] w_{i,n_i+1}$$

where $Z_u \in \mathrm{RS}'(Y_u)$ $(1 \le u \le k)$ and $1 \le i \le d(Y_0)$.

Step II: For each terminating production $r'_h : Y_0 \to f'_h$ with f'_h defined as $f'^{[i]}_h = w_i$, define rules $q_i[a_h] \to w_i$ for $1 \le i \le d(Y_0)$.

Step III: Define

$$q_1[\hat{r}_{\mathbf{start}}(x)] \to q_1[x]. \qquad (11)$$

It is clear that the constructed transducer M is deterministic. A transducer M and a cfg G' defined as above have Property 3.2. The idea behind the proof is similar to that of Property 3.1, and its proof is shown in the appendix.

Theorem 3.2: $yL(\mathrm{DFSTS}) = \mathrm{PMCFL}$.

Proof. In Lemma 3.1, it has been shown that $yL(\mathrm{DFSTS}) \subseteq \mathrm{PMCFL}$, and hence it suffices to show that $\mathrm{PMCFL} \subseteq yL(\mathrm{DFSTS})$. We show that $L(G) = yL(M, G')$ for M and G' constructed as above.

If $w \in L_G(S)$, then there is a production of pmcfg

$$r_h : S \to f_h[Y_1, \ldots Y_k] \tag{12}$$

which is the last production applied to obtain w. By Property 3.2, there is a derivation tree t of G' such that the root is $\hat{r}_{hZ_1 \cdots Z_k} : R_h \to Z_1 \cdots Z_k$ $(Z_u \in \mathrm{RS}'(Y_u), 1 \leq u \leq k)$ and $q_1[t] \overset{*}{\Rightarrow} w$ holds. Let $t' = \hat{r}_{\mathrm{start}}(t)$ then, since $R_h \in$ RS'(S), t' is also a derivation tree of cfg G'. Hence, $w \in yL(M, G')$ holds by (11).

In a reverse way, we can prove that if $w \in yL(M, G')$ then $w \in L_G(S)$, and the theorem holds. $\square$

3.3 Recognition of yL(DFSTS)

In the previous sections, we show that yL(DFSTS) equals to PMCFL. Since deterministic polynomial time recognition algorithm for PMCFL has been proposed[8], it can be concluded that yL(DFSTS) is in $\mathcal{P}$ of computational complexity. This result has been noted in an earlier paper Ref.[1] as a corollary of its main result, but the running-time required for recognition was not analyzed.

By combining the recognition algorithm for PMCFL[8] and the construction procedure described in Section 3.1, we obtain an effective procedure to recognize yL(DFSTS). In the rest of this section, we investigate the complexity of the recognition of yL(DFSTS). First, we review results on the recognition of PMCFL.

Refer to the condition (G3) and expression (1) in the definition of pmcfg's in Section 2.1. The *degree* of a function f has been defined as $\sum_{h=1}^{r(f)}(n_h + 1)$, which equals to the dimension of f plus the total number of variables appearing in the right-hand side of f[8]. If the maximum degree among the functions of G is e, then G is called a *pmcfg with degree* e. In the same way, an *mcfg with degree* e is defined.

Lemma 3.3[8]**:** A pmcfl which is generated by pmcfg with degree e can be recognized in $O(|w|^{e+1})$-time where $|w|$ denotes the length of an input. $\square$

Next we define the *degree* of a deterministic yT-transducer $M = (Q, \Sigma, \Delta, q_0, R)$. For $\sigma \in \Sigma$ and $q \in Q$, let $n_{q,\sigma}$ denote the number of occurrences of variables in the right-hand side of a rule whose left-hand side is $q[\sigma(x_1, \ldots, x_n)]$. If no rule is defined for $q[\sigma(x_1, \ldots, x_n)]$, then $n_{q,\sigma} = 0$. Since the yT-transducer is deterministic, $n_{q,\sigma}$ can be defined uniquely. For example, in Example 2.3, $n_{q_i,+} = 2$ and $n_{q_d,\cdot} = 4$. Define the *degree* of a symbol $\sigma \in \Sigma$ as $|Q| + \sum_{q \in Q} n_{q,\sigma}$. If the maximum degree among the symbol in Σ is e, then M is called a *yT-transducer with degree e*. An *fsts with degree e* is an fsts of which yT-transducer is with degree e.

The readers can easily verify that a deterministic fsts with degree e is translated into a pmcfg with degree e by using the construction described in Section 3.1. Hence the following theorem holds.

Theorem 3.4: The yield language generated by an fsts with degree e can be recognized in $O(|w|^{e+1})$-time where $|w|$ denotes the length of an input. $\square$

4. Monadic FSTS' and $\mathcal{NP}$-completeness

In previous sections, the class of yield languages generated by deterministic fsts' is shown to be in $\mathcal{P}$. In Ref.[12], it has been shown that there is an $\mathcal{NP}$-complete language in the class of yield languages generated by nondeterministic fsts'. In this section, we give an $\mathcal{NP}$-complete language in a more "restricted" class of languages, $yL(\mathrm{NMFSTS}_2)$, the class of yield languages generated by nondeterministic monadic fsts' with state-bound 2 (this class is denoted as m-fsts$_2$ in Figure 1). First, a language called *Unary-3SAT*[9], which is $\mathcal{NP}$-complete, is reviewed, and then it is shown to belong to $yL(\mathrm{NMFSTS}_2)$.

A *Unary-3CNF* is a (nonempty) 3CNF in which the subscripts of variables are represented in unary. A positive literal x_i is represented by $1^i\$$ in a Unary-3CNF. Similarly, a negative literal $\neg x_i$ is represented by $1^i\#$. For example, a 3CNF

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (x_3 \vee \neg x_1 \vee \neg x_2)$$

is represented by

$$1\$11\$111\# \wedge 111\$1\#11\#$$

in a Unary-3CNF. *Unary-3SAT* is the set of all satisfiable Unary-3CNF's. Clearly, Unary-3SAT is $\mathcal{NP}$-complete.

A nondeterministic monadic yT-fsts (M, G) with state-bound 2 which generates Unary-3SAT is defined as follows. First, define a cfg $G = (V_N, V_T, P, S)$ where $V_N = \{S, T, F\}$, $V_T = \{e\}$ and the productions in P are as follows:

$$\begin{array}{lclclcl}
r_{SS} & : & S \to S & \quad & r_{Te} & : & T \to e \\
r_{ST} & : & S \to T & & r_{FT} & : & F \to T \\
r_{SF} & : & S \to F & & r_{FF} & : & F \to F \\
r_{TT} & : & T \to T & & r_{Fe} & : & F \to e \\
r_{TF} & : & T \to F. & & & &
\end{array}$$

Note that G is a regular grammar, and hence this fsts is monadic. Let u be a derivation tree of G. Then u has a following form;

$$u \triangleq \underbrace{r_{SS}(\cdots(r_{SS}}_{m-1}(r_{Sp_1}(u')))\cdots)$$

where

$$u' = r_{p_1p_2}(r_{p_2p_3}(\cdots(r_{p_ne}(e))\cdots))$$

and $p_i \in \{T, F\}$ for $1 \leq i \leq n$. The outer m ($m \geq 1$) symbols of u are the rules whose left-hand side is S, and the next n symbols are the rules whose left-hand side is T or F.

Next, a yT-transducer $M = (Q, \Sigma, \Delta, q_0, R)$ is constructed to transduce u into a Unary-3CNF E such that

- E has m clauses,
- there are at most n distinct variables $x_1, \ldots, x_n$ in E, and
- the value of E becomes true if values are assigned to the variables as

$$x_i = \begin{cases} \text{TRUE} & \text{if } p_i \text{ is } T \\ \text{FALSE} & \text{if } p_i \text{ is } F \end{cases} \qquad (1 \le i \le n). \tag{13}$$

Let $Q = \{q_0, q_c, q_t, q_a\}$, $\Sigma = \{r_{SS}, \ldots, r_{Fe}, e\}$ and $\Delta = \{1, \wedge, \$, \#\}$. Since there are many rules in R, we will use an abbreviated notation. For example, the following four rules

$$q_a[r_{Te}(x)] \to 1\$, \quad q_a[r_{Te}(x)] \to 1\#$$
$$q_a[r_{Fe}(x)] \to 1\$, \quad q_a[r_{Fe}(x)] \to 1\#$$

are abbreviated as "$q_a[r_{Te}(x)] = q_a[r_{Fe}(x)] \to 1\$$ or $1\#$". By using this notation, define R as following rules (R1) through (R9):

(R1) $q_0[r_{SS}(x)] \to q_c[x] \wedge q_0[x]$.

(R2) $q_c[r_{SS}(x)] \to q_c[x]$.

By the rules (R1) and (R2), M transduces $q_0[u]$ into

$$\underbrace{q_c[r_{Sp_1}(u')] \wedge \cdots \wedge q_c[r_{Sp_1}(u')] \wedge q_0[r_{Sp_1}(u')]}_{m}. \tag{14}$$

As we explain later, each $q_c[\cdots]$ and $q_0[\cdots]$ derives one clause. Hence m clauses will be derived from $q_0[u]$.

(R3) $q_0[r_{ST}(x)] = q_0[r_{SF}(x)] = q_c[r_{ST}(x)] = q_c[r_{SF}(x)] \to$
$q_t[x]q_a[x]q_a[x]$ or $q_a[x]q_t[x]q_a[x]$ or $q_a[x]q_a[x]q_t[x]$.

By these rules, each $q_c[r_{Sp_1}(u')]$ ($q_0[r_{Sp_1}(u')]$) in (14) derives one of $q_t[u']q_a[u']q_a[u']$, $q_a[u']q_t[u']q_a[u']$ or $q_a[u']q_a[u']q_t[u']$.

(R4) $q_t[r_{TT}(x)] = q_t[r_{TF}(x)] \to 1q_t[x]$ or $1\$$.

(R5) $q_t[r_{Te}(x)] \to 1\$$.

(R6) $q_t[r_{FT}(x)] = q_t[r_{FF}(x)] \to 1q_t[x]$ or $1\#$.

(R7) $q_t[r_{Fe}(x)] \to 1\#$.

Suppose that a derivation from $q_t[u']$ has been proceeded and the current configuration is $q_t[r_{p_i p_{i+1}}(\cdots)]$. Now, transducer M has two choices (see rules (R4) and (R6));

- generate 1 and continue a translation of subtree, or
- generate 1\$ if p_i is T, $1\#$ if p_i is F and complete a translation.

If M completes a translation and p_i is T (resp. F), then $q_t[u']$ has derived $1^i\$$ (resp. $1^i\#$). Note that this is a literal x_i (resp. $\bar{x}_i$) which becomes "true" under the assignment (13).

(R8) $q_a[r_{TT}(x)] = q_a[r_{TF}(x)] = q_a[r_{FT}(x)] = q_a[r_{FF}(x)] \to$
$1q_a[x]$ or 1\$ or $1\#$.

(R9) $q_a[r_{Te}(x)] = q_a[r_{Fe}(x)] \to 1\$$ or $1\#$.

These are similar to the rules (R4) through (R7); $q_a[u']$ derives some literal but it is not guaranteed to become "true".

Now, the readers can easily verify that this fsts has a state-bound 2, and that this fsts can derive an arbitrary satisfiable Unary-3CNF.

Theorem 4.1: Unary-3SAT is in $yL(\text{NMFSTS}_2)$. □

5. Conclusions

We have shown that the class of yield languages generated by deterministic fsts' equals to the class of parallel multiple context-free languages. Also we have shown that the class of yield languages generated by nondeterministic monadic fsts' with state-bound 2 contains at least one $\mathcal{NP}$-complete language. These results together with already known results are summarized in Figure 2. The hierarchy with respect to state-bounds (and copying-bounds) is omitted from the figure for simplicity. In the figure, D-FSTS, FC-FSTS and M-FSTS denote the classes of yield languages generated by deterministic fsts', finite-copying fsts' and monadic fsts', respectively. Corresponding to the hierarchy in Figure 2, some subclasses of lexical-functional grammars (lfg's) can be defined. Relations between the generative capacities of lfg's, fsts' and pmcfg's are investigated in Ref.[14]. For further study, it remains to clarify the relation between dimensions of pmcfg's and state-bounds (and also copying-bounds) of deterministic fsts'.

Acknowledgement

We'd like to thank to an anonymous referee and Prof. David Weir of Sussex University for their helpful suggestions.

References

[1] Engelfriet J.: "The Complexity of Languages Generated by Attribute Grammars", SIAM J. Comput., **15**-1, pp.70–86 (Feb. 1986).

[2] Engelfriet J. and Heyker L.: "The String Generating Power of Context-Free Hypergraph Grammars", J. Comput. & Syst. Sci., **43**, pp.328–360 (1991).

[3] Engelfriet J., Rosenberg G. and Slutzki G.: "Tree Transducers, L Systems, and Two-Way Machines", J. Comput. & Syst. Sci., **20**, pp.150–202 (1980).

[4] Kaji Y., Nakanishi R., Seki H. and Kasami T.: "The universal recognition problems for multiple context-free grammars and for linear context-free rewriting systems", IEICE Trans. on Information and Systems, **E75**-D, 1, pp.78–88 (Jan. 1992).

[5] Kaji Y., Nakanishi R., Seki H. and Kasami T.: "The universal recognition problems for parallel multiple context-free grammars and for their subclasses", IEICE Trans. on Information and Systems, **E75**-D, 7, pp.499–508 (Jul. 1992).

[6] Kaji Y., Nakanishi R., Seki H. and Kasami T.: "Parallel Multiple Context-Free Grammars and Finite State Translation Systems", IEICE Technical Report, **COMP92-34** (Sep. 1992).

[7] Kasami T., Seki H. and Fujii M.: "Generalized Context-Free Grammars and Multiple Context-Free Grammars", Trans. IEICE, **J71-D-I**, 5, pp.758–765 (May 1988) (in Japanese).

[8] Kasami T., Seki H. and Fujii M.: "On the Membership Problem for Head Languages and Multiple Context-Free Languages, Trans. IEICE, **J71-D-I**, 6, pp. 935–941 (June 1988) (in Japanese).

[9] Nakanishi R., Seki H. and Kasami T.: "On the Generative Capacity of Lexical-Functional Grammars", IEICE Trans. Inf. and Syst., **75**-D, 7, pp.509–516 (July 1992).

[10] Pollard C.J.: "Generalized Phrase Structure Grammars, Head Grammars and Natural Language", Ph.D. dissertation, Stanford University (1984).

[11] Rounds W.C.: "Context-Free Grammars on Trees", Proc. of ACM Symp. on Theory of Computing, pp.143–148 (May 1969).

[12] Rounds W.C.: "Complexity of Recognition in Intermediate-Level Languages", IEEE 14th Annual Symp. on SWAT., pp.145–158, (Oct. 1973).

[13] Seki H., Matsumura T., Fujii M. and Kasami T.: "On Multiple Context-Free Grammars", Theoretical Computer Science, **88**, 2, pp.191-229 (Oct. 1991).

[14] Seki H., Nakanishi R., Kaji Y., Ando S. and Kasami T.: "Parallel Multiple Context-Free Grammars, Finite-State Translation Systems, and Polynomial-Time Recognizable Subclasses of Lexical-Functional Grammars", Proc. of 31st meeting of Assoc. Comput. Ling. pp.130–139 (June 1993).

[15] Thatcher J.W.: "Characterizing Derivation Trees of Context-Free Grammars through a Generalarization of Finite Automata Theory", J. Comput. & Syst. Sci., **1**, pp.317–322 (Dec. 1967).

[16] Vijay-Shanker K., Weir D.J. and Joshi A.K.: "Tree Adjoining and Head Wrapping", Proc. 11th Intl. Conf. on Comput. Ling., pp.202–207 (1986).

[17] Vijay-Shanker K., Weir D.J. and Joshi A.K.: "Characterizing structural descriptions produced by various grammatical formalisms", Proc. of 25th meeting of Assoc. Comput. Ling., pp.104–111 (June 1987).

[18] Weir D.J. : "Characterizing Mildly Context-Sensitive Grammar Formalisms", Ph.D. thesis, University of Pennsylvania (1988).

[19] Weir D.J.: "Linear Context-Free Rewriting Systems and Deterministic Tree-Walking Transducers", Proc. of 30th meeting of Assoc.

Comput. Ling. (June 1992).

Appendix

A. Proof of Property 3.2

<u>(*Only if part*)</u> It is shown by induction on the number of applications of (L1) and (L2) to obtain a tuple of strings $(\alpha_1, \ldots, \alpha_s)$. For the basis, assume that $\bar{\alpha} = (\alpha_1, \ldots, \alpha_s) \in L_G(X)$ and it is obtained by one application of (L1). Then the applied terminating production is $r'_h : X \to f'_h$ where $f'_h = \bar{\alpha}$. If we take $t = a_h$, then t is a derivation tree of cfg G' and the property holds by the construction of rules in Step II.

Next, assume that the property holds for every tuple of strings which can be obtained by d' or less applications of (L1) and (L2), and consider the case that $\bar{\alpha} = (\alpha_1, \ldots, \alpha_s) \in L_G(X)$ is obtained by $d'+1$ applications. Let

$$r_h : X \to f_h[Y_1, \ldots, Y_k] \tag{15}$$

be the last production applied to obtain $\bar{\alpha}$ where f_h is defined as

$$f_h^{[i]}[\bar{x}_1, \ldots, \bar{x}_k] = w_{i,1}x_{\mu(i,1)\eta(i,1)}w_{i,2}\cdots w_{i,n_i}x_{\mu(i,n_i)\eta(i,n_i)}w_{i,n_i+1} \tag{16}$$

for $1 \leq i \leq d(X)$. Then, there are $\bar{\beta}_u = (\beta_{u1}, \ldots, \beta_{ud(Y_u)}) \in L_G(Y_u)$ $(1 \leq u \leq k)$ such that

$$\bar{\alpha} = f_h[\bar{\beta}_1, \ldots, \bar{\beta}_k]. \tag{17}$$

Each $\bar{\beta}_u$ can be obtained by d' applications or less, and there is a nonterminating production $r_{h_u} : Y_u \to f_{h_u}[Y_{u1}, \ldots, Y_{uk_u}]$ (or terminal production $r'_{h_u} : Y_u \to f'_{h_u}$) which is the last production applied to obtain $\bar{\beta}_u$. By the inductive hypothesis, there are derivation trees t_u $(1 \leq u \leq k)$ such that

$$q_v[t_u] \stackrel{*}{\Rightarrow} \beta_{uv} \tag{18}$$

for $1 \leq v \leq d(Y_u)$, and the root of t_u is $\hat{r}_{h_u Z_{u1}\cdots Z_{uk_u}} : R_{h_u} \to Z_{u1}\cdots Z_{uk_u}$ (or a_{h_u}). Note that $R_{h_u} \in$ RS'(Y_u) (or $a_{h_u} \in$ RS'(Y_u)) holds for $1 \leq u \leq$

k. Since pmcfg G has a nonterminating production r_h (see (15)), cfg G' has a production $\hat{r}_{hZ_1\cdots Z_k} : R_h \to Z_1 \cdots Z_k$ such that

$$\begin{cases} Z_u = R_{h_u} & \text{if the root of } t_u \text{ is } \hat{r}_{h_u Z_{u1}\cdots Z_{uk_u}}, \\ Z_u = a_{h_u} & \text{if the root of } t_u \text{ is } a_{h_u}. \end{cases} \tag{19}$$

Hence if we take $t = \hat{r}_{hZ_1\cdots Z_k}(t_1, \ldots, t_k)$ then t is a derivation tree of G' and

$$\begin{array}{rcll} q_i[t] & = & q_i[\hat{r}_{hZ_1\cdots Z_k}(t_1,\ldots,t_k)] & \\ & \Rightarrow & w_{i,1}q_{\eta(i,1)}[t_{\mu(i,1)}]w_{i,2}\cdots w_{i,n_i}q_{\eta(i,n_i)}[t_{\mu(i,n_i)}]w_{i,n_i+1} & \text{by (10)} \\ & \stackrel{*}{\Rightarrow} & w_{i,1}\beta_{\mu(i,1)\eta(i,1)}w_{i,2}\cdots w_{i,n_i}\beta_{\mu(i,n_i)\eta(i,n_i)}w_{i,n_i+1} & \text{by (18)} \\ & = & f_h^{[i]}[\bar{\beta}_1,\ldots,\bar{\beta}_k] & \text{by (16)} \\ & = & \alpha_i & \text{by (17)} \end{array}$$

for $1 \le i \le d(X)$. That is, $q_i[t] \stackrel{*}{\Rightarrow} \alpha_i$ $(1 \le i \le d(X))$ and the property holds.

<u>*(If part)*</u> The only if part is shown by induction on the size of a derivation tree t of cfg G'. For the basis, consider a derivation tree of size one, that is, $t = a_h$ for some $a_h \in V_T$, and assume that $q_i[t] \stackrel{*}{\Rightarrow} \alpha_i$ for $1 \le i \le s$ and it derives no output for $i > s$. Then, there are rules $q_i[a_h] \to \alpha_i$ $(1 \le i \le s)$ and by the construction of rules of M in Step II, pmcfg G has a terminating production $r'_h : Y_0 \to f'_h$ with $d(Y_0) = s$ and $f'^{[i]}_h = \alpha_i$ for $1 \le i \le s$. Hence, $(\alpha_1, \ldots, \alpha_s) \in L_G(Y_0)$ and the property holds.

Assume that the property holds for every derivation tree whose size is d' or less, and consider a derivation tree $t = \hat{r}_{hZ_1\cdots Z_k}(t_1, \ldots, t_k)$ of size $d' + 1$ such that

$$\begin{array}{rcll} q_i[t] & \Rightarrow & w_{i,1}q_{\eta(i,1)}[t_{\mu(i,1)}]w_{i,2}\cdots w_{i,n_i}q_{\eta(i,n_i)}[t_{\mu(i,n_i)}]w_{i,n_i+1} & (20) \\ & \stackrel{*}{\Rightarrow} & w_{i,1}\beta_{\mu(i,1)\eta(i,1)}w_{i,2}\cdots w_{i,n_i}\beta_{\mu(i,n_i)\eta(i,n_i)}w_{i,n_i+1} & \\ & = & \alpha_i & (21) \end{array}$$

for $1 \le i \le s$. Let $\hat{r}_{h_u Z_{u1}\cdots Z_{uk_u}}$ (or a_{h_u} possibly) be the root of subtree t_u $(1 \le u \le k)$. To apply the inductive hypothesis to each subtree

t_u $(1 \leq u \leq k)$, we first investigate the nonterminal on the left-hand side of r_{h_u} which is a corresponding production of pmcfg G. Since t is a derivation tree of cfg G' and the left-hand side of $\hat{r}_{h_u Z_{u1} \cdots Z_{uk_u}}$ is R_{h_u} (see 8), there is a production $\hat{r}_{hZ_1 \cdots Z_k} : R_h \rightarrow Z_1 \cdots Z_k$ such that (19) holds. By the construction of productions of G' in Step A, pmcfg G has a production

$$r_h : Y_0 \rightarrow f_h[Y_1, \ldots, Y_k] \tag{22}$$

such that $Z_u \in \mathrm{RS}'(Y_u)$ holds for $1 \leq u \leq k$. By the definition of RS' and (19), it follows that the left-hand side of r_{h_u} (or r'_{h_u}) is Y_u.

Next, consider the rules of transducer M which are used in (20). Apparently, the rules used are defined in Step I, and it follows that the function f_h in (22) is defined as

$$f_h^{[i]}[\bar{x}_1, \ldots, \bar{x}_k] = w_{i,1} x_{\mu(i,1)\eta(i,1)} w_{i,2} \cdots w_{i,n_i} x_{\mu(i,n_i)\eta(i,n_i)} w_{i,n_i+1} \tag{23}$$

for $1 \leq i \leq d(Y_0) = s$ where $\bar{x}_u = (x_{u1}, \ldots, x_{ud(Y_u)})$ $(1 \leq u \leq k)$. Since pmcfg G satisfies the non-erasing condition, for every u $(1 \leq u \leq k)$ and v $(1 \leq v \leq d(Y_u))$, the variable x_{uv} appears at least once on the right-hand side of (23) for some i $(1 \leq i \leq s)$. Hence, $q_v[t_u]$ appears at least once on the right-hand side of (20) for some i $(1 \leq i \leq s)$, and it follows that $q_v[t_u] \overset{*}{\Rightarrow} \beta_{uv}$ holds for every u $(1 \leq u \leq k)$ and v $(1 \leq v \leq d(Y_u))$. Since the size of t_u $(1 \leq u \leq k)$ equals to d' or less, by the inductive hypothesis,

$$\bar{\beta}_u = (\beta_{u1}, \ldots, \beta_{ud(Y_u)}) \in L_G(Y_u) \tag{24}$$

for each u $(1 \leq u \leq k)$. (Remind that the root of t_u is $\hat{r}_{h_u Z_{u1} \cdots Z_{uk_u}}$ (or a_{h_u}) and the left-hand side of r_{h_u} (or r'_{h_u}) is Y_u.) Now, replacing $\bar{x}$'s with $\bar{\beta}$'s in (23), and by (21),

$$\begin{aligned} & f_h^{[i]}[\bar{\beta}_1, \ldots, \bar{\beta}_k] \\ = \;& w_{i,1} \beta_{\mu(i,1)\eta(i,1)} w_{i,2} \cdots w_{i,n_i} \beta_{\mu(i,n_i)\eta(i,n_i)} w_{i,n_i+1} \\ = \;& \alpha_i \end{aligned} \tag{25}$$

for $1 \leq i \leq d(Y_0)$. By (22),(24) and (25), $(\alpha_1, \ldots, \alpha_s) \in L_G(Y_0)$ and the property holds. Hence, Property 3.2 has proved. $\square$

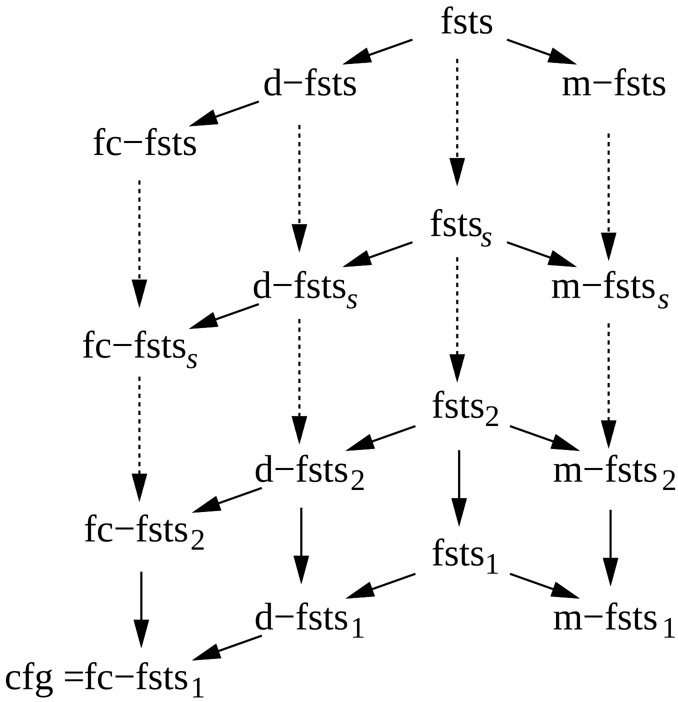
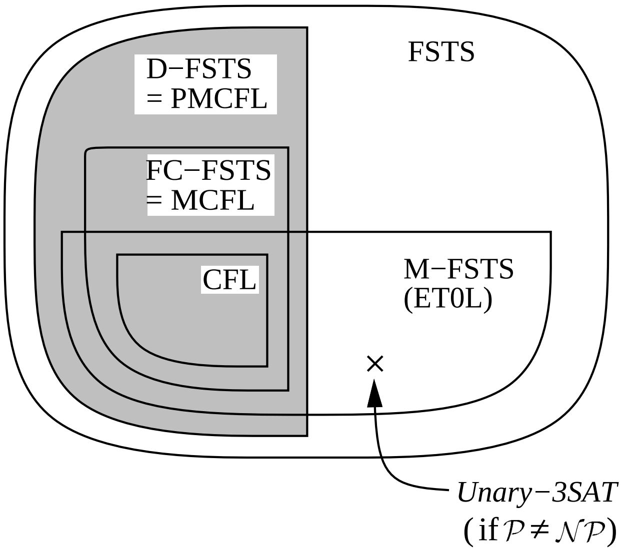


Figure 1: Generative power hierarchy of subclasses of fsts'[3].



A language which belongs to the shaded region in the figure is recognizable in deterministic polynomial time.

Figure 2: The inclusion relations of the classes of languages.