Representations and Characterizations of Languages in Chomsky Hierarchy by Means of Insertion-Deletion Systems

Gheorghe Păun[(A,B)] Mario J. Pérez-Jiménez[(B,C)]
Takashi Yokomori[(D)]

[(A)]Institute of Mathematics of the Romanian Academy
PO Box 1-764 – 014700 Bucharest – Romania
george.paun@imar.ro, gpaun@us.es

[(B)]Department of Computer Science and Artificial Intelligence
Univ. of Sevilla – Avda Reina Mercedes s/n – 41012 Sevilla – Spain
marper@us.es

[(D)]Department of Mathematics
Faculty of Education and Integrated Arts and Sciences
Waseda University – 1-6-1 Nishiwaseda – Shinjyuku-ku
Tokyo 169-8050 – Japan
yokomori@waseda.jp

Abstract. Insertion-deletion operations are much investigated in linguistics and in DNA computing and several characterizations of Turing computability were obtained in this framework.

In this note we contribute to this research direction with a new characterization of this type, as well as with representations of regular and context-free languages, mainly starting from context-free insertion systems of as small as possible complexity. For instance, each recursively enumerable language L can be represented in a way similar to the celebrated Chomsky-Schützenberger representation of context-free languages, i.e., in the form $L = h(L(\gamma) \cap D)$, where γ is an insertion system of weight (3, 0) (at most three symbols are inserted in a context of length zero), h is a projection, and D is a Dyck language. A similar representation can be obtained for regular languages, involving insertion systems of weight (2,0) and star languages, as well as for context-free languages – this time using insertion systems of weight (3, 0) and star languages.

Keywords: insertion-deletion systems, recursively enumerable languages, context-free languages, regular languages

[(A)]Partially supported by Project BioMAT 2-CEx06-11-97/19.09.06.

[(C)]Partially supported by Project TIN2006-13425 of the Ministry of Education and Science of Spain, cofinanced by FEDER funds.

[(D)]Partially supported by Grant of Faculty Development Award, Waseda University and Grant-in-Aid for Scientific Research on Priority Area No. 14085202, Ministry of Education, Culture, Sports, Science and Technology, Japan.

1 Introduction

Insertion and deletion operations were investigated in linguistics and formal language theory since "old" times – see, e.g., [2], [6], [11]. Such operations can also be implemented, at least theoretically, in terms of DNA biochemistry (see [4], [5], [12] and the references therein), hence they can be the ground for performing computations with DNA molecules. Pleasantly enough, the power of computing models based on insertion-deletion is rather large: several characterizations of Turing computability (technically, of recursively enumerable languages, RE languages for short) were obtained in this framework. A recent characterization uses only context-free operations, i.e., with the strings to be inserted or deleted not being dependent on the context where the operations are performed – see [7].

An insertion or deletion operation is based on a triple of the form (u, w, v) (which is called a *rule*), where u, w, v are strings over a specified alphabet; when using such a triple as an insertion rule, we pass from a string $xuvy$ to $xuwvy$, while in the deletion mode we pass from $xuwvy$ to $xuvy$ (the string w is inserted in, respectively deleted from the context (u, v)). The length m of w and n, the maximal length of u and v, is called the *degree* of the rule – we say that the rule is of degree (m, n). An insertion-deletion (abbreviated, ins-del) system consists of a finite set of rules and a set of axiom strings; the language generated by such a system consists of all strings over a specified alphabet which can be produced by using the insertion-deletion rules, starting from axioms. The descriptional complexity of ins-del systems can be captured by various parameters, such as the length and the number of axioms, the length and the number of ins-del rules, and so on. In this paper we only consider the length of inserted strings (always used in a context-free manner).

With these notations, the result from [7] gives a characterization of RE languages in terms of ins-del systems with insertion rules of degree at most (3, 0) (resp., (2, 0)) and deletion rules of degree at most (2, 0) (resp., (3, 0)). The optimality of this result was recently proved in [15]: if both insertion and deletion rules are of degree at most (2, 0), then only context-free languages can be obtained.

In this paper we look for characterizations of RE languages based on only insertion or only deletion operations, applied in a context-free manner and as restricted as possible in what concerns the length of the inserted/deleted string. Because we separate insertion and deletion rules, in the case of devices based on only deletion operations, the produced language is defined in a "reduction mode", in the sense that it consists of all strings which can be *reduced* to an axiom by means of deletion operations. Following in some respect the proof technique from [7], we get characterizations of RE languages of a form which is classic in formal language theory for context-free languages, namely, the Chomsky-Schützenberger one: each context-free language L can be written in the form $L = h(L' \cap D)$, where h is a projection, L' is a regular language, and D is a Dyck language (details and references can be found in any formal language theory monograph; we use here as a general reference the handbook [13]). Here we prove that each RE language L

can be written in the same way, with L' being an insertion or a deletion language of degree (3, 0).

The construction from the proof can be particularized for context-free and regular languages – in the latter case, we obtain only a representation of family of regular languages.

The optimality of these results, in terms of the length of inserted or deleted strings remains to be checked.

In what concerns the characterizations of RE languages by using insertion systems and morphisms, a result of this type is already given in [8]: for any language $L \in RE$, there exist an insertion system γ of weight (4, 7) and morphisms h_1, h_2 such that $L = h_1(h_2^{-1}(L(\gamma)))$; this has been improved in [10] to an insertion system of weight (3, 3). It should be noted that our characterization has a different form, and uses only context-free insertion/deletion rules.

2 Preliminaries

We only use a few elements of formal language theory and most of them are recalled below. For unexplained details we refer to [13].

For an alphabet V, V^* is the set of all strings of symbols from V; λ is the empty string and $|x|$ is the length of $x \in V^*$. The mirror image (reversal) of $x \in V^*$ is denoted by x^R. For an alphabet V, let $\overline{V} = \{\overline{a} \mid a \in V\}$. If V contains k symbols, then the *Dyck* language (over $V \cup \overline{V}$) is the language D_k generated by the context-free grammar $G = (\{S\}, V \cup \overline{V}, S, P)$, where $P = \{S \to SS, S \to \lambda\} \cup \{S \to aS\overline{a} \mid a \in V\}$. When k is not relevant, we omit it.

A morphism $h : V^* \longrightarrow U^*$ such that $h(a) \in U$ for all $a \in V$ is called a coding, and it is a weak coding if $h(a) \in U \cup \{\lambda\}$ for all $a \in V$. A weak coding is a projection if $h(a) \in \{a, \lambda\}$ for each $a \in V$.

Because we separate here the insertion and the deletion operations, the insertion (ins) and deletion (del) systems have the same architecture: such a system is a triple $\gamma = (V, A, P)$, where V is an alphabet, A is a finite set of strings over V called axioms, and P is a finite set of triples (u, w, v), where $u, w, v \in V^*$. Two relations are defined on V^* with respect to γ:

$$x \Longrightarrow_{ins} y \text{ iff } x = x_1uvx_2, y = x_1uwvx_2 \text{ for some } (u, w, v) \in P, x_1, x_2 \in V^*,$$
$$x \Longrightarrow_{del} y \text{ iff } x = x_1uwvx_2, y = x_1uvx_2 \text{ for some } (u, w, v) \in P, x_1, x_2 \in V^*.$$

The reflexive and transitive closure of $\Longrightarrow_\alpha, \alpha \in \{ins, del\}$, is denoted by $\Longrightarrow_\alpha^*$. A sequence of n steps of $\Longrightarrow_\alpha$ is denoted by $\Longrightarrow_\alpha^n$. When this is clear from the context, the subscript *ins* or *del* is omitted.

For a system γ as above we define two languages:

$$L_{ins}(\gamma) = \{w \in V^* \mid z \Longrightarrow_{ins}^* w, z \in A\},$$
$$L_{del}(\gamma) = \{w \in V^* \mid w \Longrightarrow_{del}^* z, z \in A\}.$$

$L_{ins}(\gamma)$ and $L_{del}(\gamma)$ are called the insertion language and the deletion language specified by γ, respectively.

An ins/del system $\gamma = (V, P, A)$ is said to be of *weight* (m, n) iff

$$m = max\{|w| \mid (u, w, v) \in P\},$$
$$n = max\{|u| \mid (u, w, v) \in P \text{ or } (v, w, u) \in P\}.$$

By INS_m^n, DEL_m^n, we denote the families of languages $L_{ins}(\gamma), L_{del}(\gamma)$, respectively, generated by ins/del systems of weight (m', n'), where $m' \leq m$ and $n' \leq n$.

It is important to note that in [7], [12], etc., an ins-del system is a construct $\gamma = (V, T, A, P)$, where V is an alphabet, $T \subseteq V$ (terminal alphabet), $A \subset V^*$ is a finite set of axioms, and P is a finite set of insertion or deletion rules. The language generated by γ consists of all strings over T^* which can be obtained by starting from a string in A and using finitely many times insertion and deletion rules from P. The family of languages generated in this way by ins-del systems with insertion rules of degree at most (m, n) and deletion rules of degree at most (p, q) is denoted by $INS_m^n DEL_p^q$. When any of the parameters m, n, p, q is not bounded, it is replaced with $*$.

We denote by RE, CS, CF, REG, FIN the families of recursively enumerable, context-sensitive, context-free, regular, and finite languages, respectively.

With these notations, we recall some of the results reported in the literature about these families (those given without references can be found in [12]):

- $FIN \subset INS_*^0 \subset INS_*^1 \ldots \subset INS_*^* \subset CS$.
- REG is incomparable to all families INS_*^n, for $n \geq 0$, but we have $REG \subset INS_*^* DEL_0^0$.
- All families $INS_*^n, n \geq 0$, are anti-AFLs.
- INS_2^2 contains non-semilinear languages.
- $RE = INS_3^0 DEL_2^0 = INS_2^0 DEL_3^0 = INS_1^1 DEL_2^0 = INS_1^1 DEL_1^1$ ([7], [14]).
- $INS_*^1 DEL_0^0 \subseteq CF$.
- $INS_2^0 DEL_2^0 \subseteq CF$ ([15]).
- Each regular language is the coding of a language in $INS_*^1 DEL_0^0$.
- Each language $L \in RE$ can be written in the form $L = g(h^{-1}(L'))$, where g is a weak coding, h is a morphism, and $L' \in INS_3^3 DEL_0^0$ ([10]).

3 Characterizing RE Languages in Terms of Ins/Del Systems

Let us start by the observation that $INS_m^n = DEL_m^n$: starting from an axiom z from a given set A and growing strings w by insertion according to rules (u, w, v) from a given set P is the same with starting from strings w and reducing them by means of deletion operations until reaching a string z from A.

Therefore, all results given below are valid both for insertion and for deletion systems; however, we only formulate these results (and the corresponding proofs) for the insertion case.

The main result of this paper is the following Chomsky-Schützenberger-like characterization of RE languages:

Theorem 1. *Each language $L \in RE$ can be represented in the form $L = h(L' \cap D)$, where $L' \in INS_3^0$, h is a projection, and D is a Dyck language.*

Construction of an insertion system γ: Consider a language $L \subseteq T^*$, generated by a type-0 grammar $G = (N, T, S, P)$ in Kuroda normal form. That is, each rule in P is of one of the following types:

- $AB \to CD$, where $A, B, C, D \in N$ (type 1: context-sensitive rules),
- $A \to BC$, where $A, B, C \in N$ (type 2: context-free rules),
- $A \to a$, where $A \in N$ and $a \in T \cup \{\lambda\}$ (type 3: terminal and empty rules).

Assume that the rules of P are labeled in a one-to-one manner with elements of a set $Lab(P)$.

We construct an insertion system $\gamma = (V \cup \overline{V}, \{S\}, P')$, of degree (3, 0), with

$$V = N \cup T \cup Lab(P),$$

and with P' containing the following insertion rules.

- Group 1: For each rule $r : AB \to CD$ of type 1 in P we construct the following two insertion rules:
  $$(\lambda, CDr, \lambda) \text{ and } (\lambda, \overline{BA}\overline{r}, \lambda).$$
- Group 2: For each rule $r : A \to BC$ of type 2 in P we construct the following two insertion rules:
  $$(\lambda, BCr, \lambda) \text{ and } (\lambda, \overline{A}\overline{r}, \lambda).$$
- Group 3: For each rule $r : A \to a \in P$ of type 3 in P we construct the following two insertion rules:
  $$(\lambda, a\overline{a}r, \lambda) \text{ and } (\lambda, \overline{A}\overline{r}, \lambda), \text{ where } \overline{\lambda} = \lambda.$$

For a rule $r : u \to v$ in P we say that two rules (λ, vr, λ) and $(\lambda, \overline{u}^R\overline{r}, \lambda)$ in P' are *r-complementary*, and denote their labels by r_+ and r_-, respectively. Further, by $\mapsto_r$ we denote two consecutive derivation steps using r-complementary rules (i.e., done by using by r_+ and r_-).

We define a projection $h : (V \cup \overline{V})^* \to T^*$ by $h(a) = a$ for all $a \in T$, and $h(a) = \lambda$ otherwise. Let D be the Dyck language over V.

Now we will prove that $L(G) = h(L(\gamma) \cap D)$. We start by introducing some useful notions.

For any rule $r : u \to v \in P$, let $U_r(u) = ru\overline{u}^R\overline{r}$; we call this an *r-block*. Then, we extend this notion to define *U-structures* as follows:

(I) An r-block $U_r(u)$ is a U-structure.

(II) If U_1 and U_2 are U-structures, then U_1U_2 is a U-structure.

(III) Let $\alpha_i, i = 1, 2, 3$, be U-structures or empty, with at least one α_i being non-empty; consider a string of the form $r\alpha_1u_1\alpha_2u_2\alpha_3\overline{u}^R\overline{r}$, where $u = u_1u_2$ is such that $r : u \to v \in P$. Then, this string, denoted by $U_r(\alpha_1u_1\alpha_2u_2\alpha_3)$, is a U-structure.

(IV) Nothing else is a U-structure.

In order to prove the inclusion $L(G) \subseteq h(L(\gamma) \cap D)$, the following observation is useful:

Observation 2. *Suppose that a rule $r : u \to v \in P$ is applied in a derivation step $z = \alpha u \beta \Longrightarrow z' = \alpha v \beta$ in G. Let $\tilde{z}$ be a sentential form in γ corresponding to z. Then, we can simulate this rewriting by using the rules r_+ and r_- as follows.*
(1). If u appears as a substring in a sentential form $\tilde{z} = \tilde{\alpha} u \tilde{\beta}$ in γ, then we create

$$\tilde{z'} = \tilde{\alpha} \cdot vr \cdot u \cdot \overline{u}^R \overline{r} \tilde{\beta} = \tilde{\alpha} v U_r(u) \tilde{\beta}.$$

(2)-1. Since an insertion can occur at arbitrary location of $\tilde{z}$, it may happen that u has been separated by an insertion step of (1) in an earlier step of deriving $\tilde{z}$. That is, in this case $\tilde{z}$ is of the form $\tilde{\alpha} A \tilde{\delta} B \tilde{\beta}$, where $u = AB$.
(2)-2. Even in such a case, one can derive $\tilde{z}$ so that $\tilde{\delta}$ may contain only U-structures.
(2)-3. Therefore, if $u(= AB)$ appears in separate locations in $\tilde{z}$, then one can apply r_+ and r_- to immediately before A and immediately after B in $\tilde{z}$, respectively, in a derivation of γ. Thus, we have

$$\tilde{z'} = \tilde{\alpha} vr A \tilde{\delta} B \overline{BA} \overline{r} \tilde{\beta}.$$

Let us now define a mapping ϕ over $(V \cup \overline{V})$ as follows: For any $a \in V - T$, let $a\overline{a} \sim \lambda$, and for any $a \in T$, let $a\overline{a} \sim a$. Then, one can consider a reduction operation over $(V \cup \overline{V})^*$ by iteratively using the binary relation $\sim$. We define $\phi(w)$ as the string finally obtained as the *irreducible* string in terms of this reduction operation. (Because the symbols from T and from $V - T$ are subject of different "reduction rules", the irreducible string reached when starting from a given string is unique, hence the mapping ϕ is correctly defined.)

We can prove now the following lemma.

Lemma 3. *Let $S \Longrightarrow^{n-1} z_{n-1}(= \alpha u \beta) \Longrightarrow z_n(= \alpha v \beta)$ in G, where $r : u \to v$ is used in the last step. Then, there exists a derivation of γ such that $S \Longrightarrow^{2n} \tilde{z}_n$ and $\phi(\tilde{z}_n) = z_n$.*

Proof: By induction on n. If $n = 1$, then we have $S \Longrightarrow v(= z_1)$ in G, where $r : S \to v$ in P, and $S \mapsto_r vrS\overline{S}\overline{r} = vU_r(S)$ is possible in γ. Let $\tilde{z_1} = vU_r(S)$; then $\phi(\tilde{z_1}) = v = z_1$. If $v = a \in T \cup \{\lambda\}$, then $S \mapsto_r a\overline{a}U_r(S) = \tilde{z}_1$ and $\phi(\tilde{z}_1) = z_1$.

Suppose that the claim holds true for up to $(n-1)$ and consider the derivation $S \Longrightarrow^{n-1} z_{n-1}(= \alpha u \beta) \Longrightarrow z_n(= \alpha v \beta)$ in G. By the induction hypothesis, there exists $\tilde{z}_{n-1}$ such that $S \Longrightarrow^{2(n-1)} \tilde{z}_{2(n-1)}$ and $\phi(\tilde{z}_{n-1}) = z_{n-1}$. Then, from (1) and (2)-1 of Observation 2 above, we have either
Case 1: there exist $\tilde{\alpha}$ and $\tilde{\beta}$ such that $\tilde{z}_{n-1} = \tilde{\alpha} u \tilde{\beta}$ and $\phi(\tilde{\alpha}) = \alpha$, $\phi(\tilde{\beta}) = \beta$, or
Case 2: there exist $\tilde{\alpha}$, $\tilde{\beta}$, and $\tilde{\delta}$ such that $\tilde{z}_{n-1} = \tilde{\alpha} A \tilde{\delta} B \tilde{\beta}$ and $\phi(\tilde{\alpha}) = \alpha$, $\phi(\tilde{\delta}) = \lambda$, $\phi(\tilde{\beta}) = \beta$.

In Case 1, we have:

$\tilde{z}_{n-1}(= \tilde{\alpha} u \tilde{\beta}) \mapsto_r \tilde{\alpha} vru\overline{u}^R \overline{r} \tilde{\beta} = \tilde{\alpha} v U_r(u) \tilde{\beta} = \tilde{z}_n$. Further, $\phi(\tilde{z}_n) = \alpha v \beta = z_n$.

If r is of type 3 (i.e., $r : A \to a$ with $a \in T \cup \{\lambda\}$), then

$$\tilde{z}_{n-1} \mapsto_r \tilde{\alpha} a \overline{a} U_r(A) \tilde{\beta} = \tilde{z}_n \text{ and } \phi(\tilde{z}_n) = \alpha a \beta = z_n.$$

In Case 2, we have:

$$\tilde{z}_{n-1}(= \tilde{\alpha} A \tilde{\delta} B \tilde{\beta}) \mapsto_r \tilde{\alpha} v r A \tilde{\delta} B \overline{BA} \overline{r} \tilde{\beta} = \tilde{\alpha} v U_r(A \tilde{\delta} B) \tilde{\beta} = \tilde{z}_n,$$
$$\text{and } \phi(\tilde{z}_n) = z_n,$$

and this completes the proof. □

Example 4. Consider the context-sensitive grammar G with the rule set P :

$$r_0 : S \to AY,\ r_1 : S \to AS',\ r_2 : S' \to SX,\quad r_3 : YX \to BY',$$
$$r_4 : Y \to BC,\ r_5 : Y' \to YC,\ r_6 : CX \to XC,\ r_7 : A \to a,$$
$$r_8 : B \to b,\quad r_9 : C \to c.$$

The generated language is $L(G) = \{a^n b^n c^n \mid n \geq 1\}$.
We construct an ins-system γ with the set of insertion rules P':

$$(\lambda, AYr_0, \lambda), (\lambda, \overline{S}\overline{r_0}, \lambda),\quad (\lambda, AS'r_1, \lambda), (\lambda, \overline{S}\overline{r_1}, \lambda),\ (\lambda, SXr_2, \lambda), (\lambda, \overline{S'}\overline{r_2}, \lambda),$$
$$(\lambda, BY'r_3, \lambda), (\lambda, \overline{XY}\overline{r_3}, \lambda),\ (\lambda, BCr_4, \lambda), (\lambda, \overline{Y}\overline{r_4}, \lambda),\ (\lambda, YCr_5, \lambda), (\lambda, \overline{Y'}\overline{r_5}, \lambda),$$
$$(\lambda, XCr_6, \lambda), (\lambda, \overline{XC}\overline{r_6}, \lambda),\ (\lambda, a\overline{a}r_7, \lambda), (\lambda, \overline{A}\overline{r_7}, \lambda),\quad (\lambda, b\overline{b}r_8, \lambda), (\lambda, \overline{B}\overline{r_8}, \lambda),$$
$$(\lambda, c\overline{c}r_9, \lambda), (\lambda, \overline{C}\overline{r_9}, \lambda).$$

Then, a successful derivation in G is, for instance,

$$\begin{aligned}
S &\Longrightarrow_{r_1} AS' \Longrightarrow_{r_2} ASX \Longrightarrow_{r_1} AAS'X \Longrightarrow_{r_2} AASXX \\
&\Longrightarrow_{r_0} AAAYXX \Longrightarrow_{r_3} AAABY'X \Longrightarrow_{r_5} AAABYCX \Longrightarrow_{r_6} AAABYXC \\
&\Longrightarrow_{r_3} AAABBY'C \Longrightarrow_{r_5} AAABBYCC \Longrightarrow_{r_4} AAABBBCCC \Longrightarrow^3_{r_7} aaaBBBCCC \\
&\Longrightarrow^3_{r_8} aaabbbCCC \Longrightarrow^3_{r_9} aaabbbccc.
\end{aligned}$$

A corresponding derivation in γ is as follows:

$$\begin{aligned}
&S \mapsto_{r_1} AS'r_1S\overline{S}\overline{r_1} \mapsto_{r_2} ASXr_2S'\overline{S'}\overline{r_2}U_{r_1}(S) \\
&\mapsto_{r_1} AAS'r_1S\overline{S}\overline{r_1}XU_{r_2}(S')U_{r_1}(S) \\
&\mapsto_{r_2} AASXr_2S'\overline{S'}\overline{r_2}U_{r_1}(S)XU_{r_2}(S')U_{r_1}(S) \\
&\mapsto_{r_0} AAAYr_0S\overline{S}\overline{r_0}XU_{r_2}(S')U_{r_1}(S)XU_{r_2}(S')U_{r_1}(S) \\
&\mapsto_{r_3} AAABY'r_3YU_{r_0}(S)X\overline{XY}\overline{r_3}U_{r_2}(S')U_{r_1}(S)XU_{r_2}(S')U_{r_1}(S) \\
&\mapsto_{r_5} AAABYCr_5Y'\overline{Y'}\overline{r_5}U_{r_3}(YU_{r_0}(S)X)U_{r_2}(S')U_{r_1}(S)XU_{r_2}(S')U_{r_1}(S) \\
&\mapsto_{r_6} AAABYXCr_6CU_{r_5}U_{r_3}(YU_{r_0}(S)X)U_{r_2}(S')U_{r_1}(S)X\overline{XC}\overline{r_6}U_{r_2}(S')U_{r_1}(S) \\
&\mapsto_{r_3} AAABBY'r_3YX\overline{XY}\overline{r_3}CU_{r_6}(CU_{r_5}U_{r_3}(YU_{r_0}X)U_{r_2}U_{r_1}X)U_{r_2}U_{r_1} \\
&\mapsto_{r_5} AAABBYCr_5Y'\overline{Y'}\overline{r_5}U_{r_3}CU_{r_6}(CU_{r_5}U_{r_3}(YU_{r_0}X)U_{r_2}U_{r_1}X)U_{r_2}U_{r_1} \\
&\mapsto_{r_4} AAABBBCr_4Y\overline{Y}\overline{r_4}CU_{r_5}U_{r_3}CU_{r_6}U_{r_2}U_{r_1} \\
&\mapsto_{r_7} a\overline{a}r_7A\overline{A}\overline{r_7}AABBBCU_{r_4}CU_{r_5}U_{r_3}CU_{r_6}U_{r_2}U_{r_1} \\
&\mapsto^2_{r_7} a\overline{a}U_{r_7}a\overline{a}U_{r_7}a\overline{a}U_{r_7}BBBCU_{r_4}CU_{r_5}U_{r_3}CU_{r_6}U_{r_2}U_{r_1} \\
&\mapsto^3_{r_8} a\overline{a}U_{r_7}a\overline{a}U_{r_7}a\overline{a}U_{r_7}b\overline{b}U_{r_8}b\overline{b}U_{r_8}b\overline{b}U_{r_8}CU_{r_4}CU_{r_5}U_{r_3}CU_{r_6}U_{r_2}U_{r_1} \\
&\mapsto^3_{r_9} a\overline{a}U_{r_7}a\overline{a}U_{r_7}a\overline{a}U_{r_7}b\overline{b}U_{r_8}b\overline{b}U_{r_8}b\overline{b}U_{r_8}c\overline{c}U_{r_9}U_{r_4}c\overline{c}U_{r_9}U_{r_5}U_{r_3}c\overline{c}U_{r_9}U_{r_6}U_{r_2}U_{r_1}.
\end{aligned}$$

The following observations are useful for the proof of the reverse inclusion.

Observation 5. *For a rule $r : u \to v$ in P, let $r_+ : (\lambda, vr, \lambda)$ and $r_- : (\lambda, \overline{u}^R\overline{r}, \lambda)$ be the two r-complementary rules.*

(1). Any successful derivation of γ requires the use of both of r-complementary rules r_+ and r_-.

(2). Let $\tilde{z}$ be any sentential form in a successful derivation of γ. Then, it must hold that for any prefix α of $\tilde{z}$, $\#_{vr}(\alpha) \geq \#_{\overline{u}^R\overline{r}}(\alpha)$, where $\#_x(\alpha)$ denotes the number of occurrences of a string x in α.

(3). Applying insertion rules within a U-structure leads to only invalid strings. Indeed, suppose that r_+ and r_- are applied on some occurrence of u appearing in a U-structure $U_{r'}(\delta_1 u \delta_2) = r'\delta_1 u \delta_2 \overline{u}^R\overline{r'}$, where $r' : u \to v'$. This derives a string $r'\delta_1 v U_r(u)\delta_2\overline{u}^R\overline{r'}$ which leads to an invalid string (i.e., not in D) unless $u = v$. This also occurs in the case where u appears in separate locations in the U-structure.

(4). A location in $\tilde{z}$ is called valid for two r-complementary rules if it is either immediately before u_1 for r_+ or immediately after u_2 for r_-, by ignoring U-structures in $\tilde{z}$, where $u = u_1u_2$. Then, applying insertion rules at valid locations only leads to valid strings. This is seen as follows: from (1), (2), (3) above, the locations for r_+ and r_- to be inserted are restricted to somewhere in the left and right, respectively, of u. In order to derive a valid string from $\tilde{z}$, it is necessary to apply r_+ and r_- to u so that these two rules together with u may eventually lead to forming a U-structure.

Now, we can prove the following result:

Lemma 6. *Let $S \Longrightarrow^{2n} \tilde{z}$ in γ and $\phi(\tilde{z}) = z(\in (N \cup T)^*)$. Then, we have $S \Longrightarrow^n z$ in G.*

Proof: By induction on n. In case $n = 1$, there exists r-complementary rules r_+ and r_- such that $r : u \to v \in P$ and $S \mapsto_r vrS\overline{S}\overline{r} = vU_r(S) = \tilde{z}$. Further, it holds that $\phi(\tilde{z}) = v = z$. Then, it is clear that we have $S \Longrightarrow v$ in G.

Suppose that the claim holds true for up to $(n-1)$ and consider the derivation $S \Longrightarrow^{2(n-1)} \tilde{z}_{n-1} \mapsto_r \tilde{z}_n = \tilde{z}$, and $\phi(\tilde{z}) = z$. (Without loss of generality, we may assume here that the last two steps are performed by r-complementary rules for some r.) Then, there exists $r : u \to v$ for which r_+ and r_- are used to derive $\tilde{z}_n$ in γ such that $\phi(\tilde{z_n}) = z \in (N \cup T)^*$. By induction hypothesis, we have $S \Longrightarrow^{n-1} z_{n-1}$ in G, where $z_{n-1} = \phi(\tilde{z}_{n-1})$. Since $\tilde{z}_{n-1} \mapsto^2 \tilde{z}_n$, there exists $r : u \to v$ for which r_+ and r_- are used to derive $\tilde{z}_n$ in γ such that $\phi(\tilde{z_n}) = z \in (N \cup T)^*$. There are two cases:

Case 1: $\tilde{z}_{n-1}$ is of the form $\tilde{\alpha}u\tilde{\beta}$, where u is in $N^2 \cup N$. Since $z_{n-1} = \phi(\tilde{z}_{n-1})$, one can write $z_{n-1} = \alpha u\beta$ with $\phi(\tilde{\alpha}) = \alpha$ and $\phi(\tilde{\beta}) = \beta$. Further, $\tilde{z}_{n-1} \mapsto_r \tilde{\alpha}vru\overline{u}^R\overline{r}\tilde{\beta} = \tilde{z}_n$ and $\phi(\tilde{z}_n) = \alpha v\beta = z_n$. Thus, we have $S \Longrightarrow^{n-1} z_{n-1}(= \alpha u\beta) \Longrightarrow_r z_n(= \alpha v\beta)$ in G.

Case 2: $\tilde{z}_{n-1}$ is of the form $\tilde{\alpha}A\tilde{\delta}B\tilde{\beta}$, where $u = AB$. As above, from $z_{n-1} = \phi(\tilde{z}_{n-1})$, one can write $z_{n-1} = \alpha AB\beta$ with $\phi(\tilde{\alpha}) = \alpha$, $\phi(\tilde{\beta}) = \beta$ and $\phi(\tilde{\delta}) = \lambda$, because $\tilde{\delta}$ only contains U-structures. Further, $\tilde{z}_{n-1} \mapsto_r \tilde{\alpha}vrA\delta B\overline{(AB)}^R\overline{r}\tilde{\beta} = \tilde{z}_n$

and $\phi(\tilde{z}_n) = \alpha v \beta = z_n$. Thus, we have $S \Longrightarrow^{n-1} z_{n-1}(= \alpha AB\beta) \Longrightarrow_r z_n(= \alpha v \beta)$ in G. □

From the two previous lemmas, we are now in a position to prove the main theorem.

Proof: [Proof of Theorem 1.] For any $w \in L(G)$, suppose that $S \Longrightarrow^* w$. Then, by Lemma 3 there exists a derivation $S \Longrightarrow^* \tilde{w}$ in γ such that $\phi(\tilde{w}) = w$. Since ϕ deletes only U-structures and elements of $\overline{T}$, this implies that $\tilde{w} \in D$ and $h(\tilde{w}) = w \in T^*$. Thus, $w \in h(L(\gamma) \cap D)$. Hence, we have $L(G) \subseteq h(L(\gamma) \cap D)$.

Conversely, suppose that let $S \Longrightarrow^* \tilde{w}$ in γ and $\phi(\tilde{w}) = w(\in T^*)$. Then, by Lemma 6 we have $S \Longrightarrow^* w$ in G. Again, $\phi(\tilde{w}) \in T^*$ implies that $\tilde{w} \in D$ and $h(\tilde{w}) = w$. Thus, we have $h(L(\gamma) \cap D) \subseteq L(G)$. □

In the proof above, starting from $G = (N, T, S, P)$ as above, instead of constructing the insertion system γ, we can construct the pure context-free grammar $G' = (V, S, P')$ with

$$\begin{aligned} V &= N \cup T \cup \overline{N} \cup \overline{T} \cup Lab(P) \cup \overline{Lab(P)}, \\ P' &= \{A \to CDrA,\ B \to B\overline{B}\overline{A}\overline{r} \mid r : AB \to CD \in P\} \\ &\cup \{A \to BCrA\overline{A}\overline{r} \mid r : A \to BC \in P\} \\ &\cup \{A \to a\bar{a}rA\overline{A}\overline{r} \mid r : A \to a \in P\}. \end{aligned}$$

Then, it is easy to derive the following corollary.

Corollary 7. *Any recursively enumerable language L can be represented in the form $L = h(L' \cap D)$, where h is a projection, L' is a pure context-free language, and D is a Dyck language.*

4 Representations/Characterizations of Regular and Context-free Languages

Because any Dyck language belongs to $INS_2^0 = DEL_2^0$, we can replace the Dyck language with a language in INS_2^0 in the Chomsky-Schützenberger characterization of context-free languages. However, we can do better (but using slightly more complex insertion systems), also restricting the type of regular languages used. Namely, it is enough to use *star* languages, i.e., languages of the form F^*, where F is a finite set of strings.

Then, we can prove the following:

Theorem 8. *A language L is context-free if and only if it can be written in the form $L = h(L' \cap R)$, where $L' \in INS_3^0$, R is a star language, and h is a projection.*

Proof: (i) Let $G = (N, T, S, P)$ be a context-free grammar in Chomsky normal form.

We construct the insertion system $\gamma = (V \cup \overline{V}, P', \{S\})$, of weight (3, 0), in a similar way as before:

$$V = N \cup T \cup Lab(P)$$

and P' contains the following insertion rules.

- For each rule $r : A \to BC$ in P, we construct the insertion rules
  (λ, BCr, λ) and $(\lambda, \overline{A}\overline{r}, \lambda)$.
- For each rule $r : A \to a$ in P, we construct the insertion rules:
  (λ, ar, λ) and $(\lambda, \overline{A}\overline{r}, \lambda)$.

Further, we define the projection $h : (V \cup \{\overline{a} \mid a \in N \cup Lab(P)\})^* \to T^*$ by $h(a) = a$ for all $a \in T$, and $h(a) = \lambda$ otherwise.

Finally, let $R = (T \cup \{rA\overline{A}\overline{r} \mid r : A \to \alpha \in P\})^*$.

From the proof of Theorem 1, it holds that $S \Longrightarrow^* z$ in G iff $S \Longrightarrow^* \overline{z}$ in γ and $\phi(\overline{z}) = z (\in (N \cup T)^*)$. From the way of constructing γ, we observe that only r-blocks appear in $\overline{z}$. Therefore, D in the proof of Theorem 1 can be replaced with the star language R.

(ii) Conversely, because $INS_3^0 = INS_3^0 DEL_0^0 \subseteq CF$ and the family CF is closed under intersection with regular languages and arbitrary morphisms, any language which can be written in the form $h(L' \cap R)$ as above is context-free. □

The previous representation can be particularized for regular languages, and in this case the insertion system will be of degree (2, 0).

Theorem 9. *Any regular language L can be represented in the form $L = h(L' \cap R)$, where $L' \in INS_2^0$, R is a star language, and h is a weak coding.*

Proof: Let $G = (N, T, S, P)$ be a regular grammar. Without loss of generality, we may assume that each rule in P is of the form either $A \to Ba$ or $A \to \lambda$, for $A, B \in N,\ a \in T$.

We construct the insertion system $\gamma = (V \cup \overline{V}, \{S_\lambda\}, P')$ of weight (2, 0) with

$$V = \{A_x \mid A \in N,\ x \in T\} \cup Lab(P)$$

and P' containing the following insertion rules.

- For each rule $r : A \to Ba$ in P, we construct the following insertion rules, for all $x \in T$,
  $(\lambda, B_a r, \lambda)$ and $(\lambda, \overline{A}_x \overline{r}, \lambda)$.
- For each rule $r : A \to \lambda$ in P, we construct the following insertion rules, for all $x \in T$,
  (λ, r, λ) and $(\lambda, \overline{A}_x \overline{r}, \lambda)$.
- For each rule $r : S \to Ba$ in P, we construct the following two rules:
  $(\lambda, B_a r, \lambda)$ and $(\lambda, \overline{S}_\lambda \overline{r}, \lambda)$.

Further, we define the morphism $h : (V \cup \overline{V})^* \to T^*$ by $h(A_a) = a$ for all $A \in N, a \in T \cup \{\lambda\}$, and $h(b) = \lambda$ otherwise.

Finally, let $R = \{rB_a\overline{B}_a\overline{r} \mid r : A \to Ba \in P,\ a \in T\}^*$.

From the proof of Theorem 1, it holds that $S \Longrightarrow^* z$ in G iff $S \Longrightarrow^* \overline{z}$ in γ and $\phi(\overline{z}) = z(\in (N \cup T)^*)$. From the way of constructing γ, we observe that only r-blocks appear in $\overline{z}$. Therefore, D in the proof of Theorem 1 can be replaced with R, and this completes the proof. □

Because $INS_2^0 - REG \neq \emptyset$ and V^* is a star language, this theorem gives only a representation of regular languages, not a characterization.

In the above proof, it is obvious that we can replace R with a Dyck language D (like in the proof of Theorem 1). Thus, we have:

Corollary 10. *Any regular language L can be expressed in the form $L = h(L' \cap D)$, where $L' \in INS_2^0$, D is a Dyck language, and h is a weak coding.*

In the proof above, instead of using the star language R and the projection h, we can consider a finite substitution g defined by

$$g(a) = \{rB_a\overline{B}_a\overline{r} \mid r : A \rightarrow Ba \in P\} \text{ for all } a \in T,$$

and then we have:

Corollary 11. *(i) Any regular language L can be expressed in the form $L = g^{-1}(L')$, where $L' \in INS_2^0$ and g is a finite substitution.*
(ii) Any context-free language L can be expressed in the form $L = h(g^{-1}(L') \cap D)$, where $L' \in INS_2^0$, h is a projection, and g is a finite substitution.

The previous representations can be combined with known representations/ characterizations of context-free and of RE languages. For instance, each RE language is the projection of the intersection of two context-free languages ([1]) and several similar results are also found in Theorem 4.14 of [16]; each of these context-free languages can then be written as in Theorem 8, etc. However, we leave the details to the reader.

5 Final Discussion

In a morphic characterization for a family of languages in the form $L = h(L' \cap D)$ with L' being from a smaller family and D being a Dyck language, one typical instance is the Chomsky-Schützenberger characterization for the family CF. As for the family RE, L can be expressed in that form with L' being a minimal linear language ([3]), while there is no such a characterization for CS ([9]).

In this paper, we have contributed to the study of insertion/deletion systems with new characterizations of context-free and recursively enumerable languages and a representation of regular languages. In all cases, context-free insertion (symmetrically, deletion) systems were used, at most of degree (3, 0). Specifically, we have shown that (i) L is in RE iff $L = h(L' \cap D)$, (ii) L is in CF iff $L = h(L' \cap R)$, and (iii) any L in REG can be expressed in the form $L = h(L' \cap D)$,

where L' is insertion/deletion language of weight (3,0) for RE and CF, and of weight (2,0) for REG, respectively, and R is a star regular language.

Finally, it remains left open whether or not these results can be improved, by decreasing the degree to (2, 0) in representing RE and CF.

References

[1] B.S. Baker, R.V. Book: Reversal-bounded multipushdown machines. *Journal of Computer and System Sciences*, 8 (1974), 315–332.

[2] B.S. Galiukschov: Semicontextual grammars (in Russian). *Mat. logica i mat. ling.*, Kalinin Univ., 1981, 38–50.

[3] S. Hirose, S. Okawa, M. Yoneda: A homomorphic characterization of recursively enumerable languages. *Theoretical Computer Science*, 35 (1985), 261–269.

[4] L. Kari, Gh. Păun, G. Thierrin, S. Yu: At the crossroads of DNA computing and formal languages: Characterizing RE using insertion-deletion systems. *Proc. 3rd DIMACS Workshop on DNA Based Computing*, Philadelphia, 1997, 318–333

[5] L. Kari, G. Thierrin: Contextual insertion/deletion and computability. *Information and Computation*, 131, 1 (1996), 47–61

[6] S. Marcus: Contextual grammars. *Rev. Roum. Math. Pures Appl.*, 14 (1969), 1525 – 1534.

[7] M. Margenstern, Gh. Păun, Y. Rogozhin, S. Verlan: Context-free insertion-deletion systems. *Theoretical Computer Science*, 330 (2005), 339–348.

[8] C. Martin-Vide, Gh. Păun, A. Salomaa: Characterizations of recursively enumerable languages by means of insertion grammars. *Theoretical Computer Science*, 205 (1998), 195–205.

[9] S. Okawa, S. Hirose, M. Yoneda: On the impossibility of the homomorphic characterization of context-sensitive languages. *Theoretical Computer Science*, 44 (1986), 225–228.

[10] K. Onodera: A note on homomorphic representation of recursively enumerable languages with insertion grammars. *IPSJ Journal*, 44, 5 (2003), 1424–1427.

[11] Gh. Păun: *Marcus Contextual Grammars*, Kluwer, Dordrecht, Boston, 1998.

[12] Gh. Păun, G. Rozenberg, A. Salomaa: *DNA Computing: New Computing Paradigms*. Springer-Verlag, Berlin, 1998.

[13] G. Rozenberg, A. Salomaa, eds.: *Handbook of Formal Languages*, Springer-Verlag, Berlin, 1997.

[14] A. Takahara, T. Yokomori: On the computational power of insertion-deletion systems. *Natural Computing*, 2:4, (2003), 321–336.

[15] S. Verlan: On minimal context-free insertion-deletion systems. In *Proc. Seventh International Workshop on Descriptional Complexity of Formal Systems*, Como, Italy, 2005 (C. Mereghetti, B. Palano, G. Pighizzini, D. Wotschke, eds.), Technical report 06-05, University of Milan, 285–292.

[16] K. Wagner, G. Wechsung: *Computational Complexity*, Reidel Publishing Co., 1986.