Linearizer and Doubler: Two Mappings to Unify Molecular Computing Models Based on DNA Complementarity

Kaoru Onodera[1] and Takashi Yokomori[2]

[1] Mathematics Major, Graduate School of Education, Waseda University, 1-6-1 Nishiwaseda, Shinjyuku-ku, Tokyo 169-8050, Japan
kaoru@akane.waseda.jp
[2] 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. Two specific mappings called *doubler* f_d and *linearizer* f_ℓ are introduced to bridge two domains of languages. That is, f_d maps string languages into (double-stranded) molecular languages, while f_ℓ transforms in the other way around. Using these mappings, we give new characterizations for the families of sticker languages and of Watson-Crick languages, which leads to not only a unified view of the two families of languages but also a clarified view of the computational capability of the DNA complementarity. One of the results implies that any recursively enumerable language can be expressed as the projective image of $f_d(L)$ for a minimal linear language L.

1 Introduction

In the late 1990's history of theoretical research on molecular computing models, sticker systems have been proposed to model the behaviors of biomolecules with sticky ends and to investigate the computational capability of those molecules based on the biomolecular property of DNA complementary. On the other hand, almost in parallel a new type of machine model called Watson-Crick automaton was introduced and studied, which is taken as a finite state machine working on double-stranded molecules (rather than linear strings). Similarly, a sticker system was introduced as one of the generative systems by using the DNA complementarity. The above two systems have a great deal of potential to provide the promising models for DNA computings. One can find a huge amount of interesting results on a variety of families of these systems and automata in, e.g., [3].

The present paper concerns a new approach to unifying a great variety of these models of computation based on DNA complementarity. The purpose of this paper is twofold : One is to explore the computational power of annealing operations between complementary molecules in terms of notions in formal language theory. The other is to clarify the current (chaotic) landscape of a variety of existing computational models based on DNA complementarity, by providing a unified view of those models.

A. Carbone and N.A. Pierce (Eds.): DNA11, LNCS 3892, pp. 224–235, 2006.

For our purpose, we introduce two specific mappings "doubler f_d" and "linearizer f_ℓ" that can bridge the two worlds of *string* languages and of *double-stranded molecular* languages. Using these mappings, we will give new characterizations for the families of sticker languages and of Watson-Crick languages. For example, through the mapping f_d, we show that the difference between sticker systems and Watson-Crick automata is essentially reduced to the one between minimal linear and regular grammars, respectively.

2 Preliminaries

We assume the reader to be familiar with the rudiments on Watson-Crick finite automata and sticker systems as well as basic notions in formal language theory (see, e.g., [3, 4]).

For an alphabet V, $\rho \subseteq V \times V$ is a symmetric relation. We denote an element $(x_1, x_2) \in V^* \times V^*$ by $\binom{x_1}{x_2}$. Instead of using a notation $V^* \times V^*$, we often use $\binom{V^*}{V^*}$. For elements $\binom{x_1}{y_1}, \binom{x_2}{y_2} \in \binom{V^*}{V^*}$, by $\binom{x_1}{y_1}\binom{x_2}{y_2}$, we represent a double stranded molecule $(x_1x_2, y_1y_2) \in V^* \times V^*$.

Let $\begin{bmatrix} V \\ V \end{bmatrix}_\rho = \{\binom{a}{b} \mid a, b \in V,\ (a, b) \in \rho\}$ and $WK_\rho(V) = \begin{bmatrix} V \\ V \end{bmatrix}_\rho^*$ (the set of all complete double stranded molecules over V including $\binom{\epsilon}{\epsilon}$).

For an element $\binom{a_1}{b_1}\binom{a_2}{b_2}\cdots\binom{a_n}{b_n} \in WK_\rho(V)$, we also write in the form $\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$, where $w_1 = a_1a_2\cdots a_n$, $w_2 = b_1b_2\cdots b_n$.

We define a set of incomplete molecules over V: $W_\rho(V) = L_\rho(V) \cup R_\rho(V) \cup LR_\rho(V)$, where

$$L_\rho(V) = \{ \boxed{\begin{matrix} x_1\ y_1 \\ y_2 \end{matrix}}, \boxed{\begin{matrix} y_1 \\ x_2\ y_2 \end{matrix}} \mid x_1, x_2 \in V^*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \in \begin{bmatrix} V \\ V \end{bmatrix}_\rho^* \},$$

$$R_\rho(V) = \{ \boxed{\begin{matrix} y_1\ z_1 \\ y_2 \end{matrix}}, \boxed{\begin{matrix} y_1 \\ y_2\ z_2 \end{matrix}} \mid z_1, z_2 \in V^*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \in \begin{bmatrix} V \\ V \end{bmatrix}_\rho^* \},$$

$$LR_\rho(V) = \{ \boxed{\begin{matrix} x_1\ y_1\ z_1 \\ y_2 \end{matrix}}, \boxed{\begin{matrix} y_1 \\ x_2\ y_2\ z_2 \end{matrix}}, \boxed{\begin{matrix} x_1\ y_1 \\ y_2\ z_2 \end{matrix}}, \boxed{\begin{matrix} y_1\ z_1 \\ x_2\ y_2 \end{matrix}} \mid x_1, x_2, z_1, z_2 \in V^*, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \in \begin{bmatrix} V \\ V \end{bmatrix}_\rho^+ \}.$$

Elements in $W_\rho(V)$ are called *bricks*.

[Sticker systems]
A *sticker system* is a 4-tuple $\gamma = (V, \rho, A, D)$, where V is a finite set of symbols, $\rho \subseteq V \times V$ is the complementary relation on V, $A \subseteq LR_\rho(V)$ is a finite set of axioms, and D is a finite set of elements in $W_\rho(V) \times W_\rho(V)$.

For $\gamma = (V, \rho, A, D)$ and $\alpha, \beta \in W_\rho(V)$, we write $\alpha \overset{d_\pi}{\Longrightarrow}_\gamma \beta$ (or simply $\alpha \Longrightarrow \beta$) if and only if $\beta = u\alpha v$, for some $d_\pi : (u, v) \in D$. That is, for example, in a graphical representation, it means

$$d_\pi : \left(\begin{array}{|c|c|}\hline u_3 & u_2 \\ \hline \end{array} \!\!\begin{array}{|c|c|}\hline \bar{u}_2 & u_1 \\ \hline \end{array} \;,\; \begin{array}{|c|c|}\hline v_2 & v_3 \\ \hline \end{array} \!\!\begin{array}{|c|c|}\hline v_1 & \bar{v}_2 \\ \hline \end{array} \right) = (u, v) \quad \text{and}$$

$$\alpha = \begin{array}{|c|c|c|}\hline \bar{u}_1 & \alpha_1 & \bar{v}_1 \\ \hline \end{array} \!\!\begin{array}{|c|}\hline \bar{\alpha}_1 \\ \hline \end{array} \overset{d_\pi}{\Longrightarrow} \begin{array}{|c|c|c|c|c|}\hline u_3 & u_2 & \bar{u}_1 & \alpha_1 & \bar{v}_1 \\ \hline \end{array} \!\!\begin{array}{|c|c|}\hline v_2 & v_3 \\ \hline \end{array} \!\!\begin{array}{|c|c|}\hline \bar{u}_2 & u_1 \\ \hline \end{array} \!\!\begin{array}{|c|}\hline \bar{\alpha}_1 \\ \hline \end{array} \!\!\begin{array}{|c|c|}\hline v_1 & \bar{v}_2 \\ \hline \end{array} = \beta,$$

where u_1, u_2, v_1, v_2 and $\bar{u}_1$, $\bar{u}_2$, $\bar{v}_1$, $\bar{v}_2$ are complementary, respectively.

For any other types of bricks for d_π in γ, we similarly define $\overset{d_\pi}{\Longrightarrow}_\gamma$. We denote by $\Longrightarrow^*$ the reflexive and transitive closure of $\Longrightarrow$.

A set of molecules generated by γ called *molecular language* is defined by

$$LM(\gamma) = \{w \in WK_\rho(V) \mid x_1 \Longrightarrow^* w,\ x_1 \in A\}.$$

Furthermore, a (*string*) *language* $L(\gamma)$ *generated* by γ is a coding image of $LM(\gamma)$, i.e., the set of all *upper* components of the molecular language $LM(\gamma)$. The classes of molecular languages and of string languages generated by γ are denoted by $\mathcal{SL}_m$ and $\mathcal{SL}$, respectively.

[Watson-Crick finite automata]

A *Watson-Crick finite automaton* (abb. WK-automaton) is defined by the tuple

$$M = (V, \rho, Q, q_0, F, \delta).$$

V is an *(input) alphabet*, Q is a finite set of *states*, V and Q are disjoint alphabets. $\rho \subseteq V \times V$ is a symmetric relation. q_0 is the *initial state* in Q. $F \subseteq Q$ is the set of *final states*. $\delta : Q \times \binom{V^*}{V^*} \to \mathcal{P}(Q)$ is a *transition mapping* such that $\delta(q, \binom{x}{y}) \neq \phi$ only for finitely many triples $(s, x, y) \in Q \times V^* \times V^*$, where $\mathcal{P}(Q)$ is the set of all possible subsets of Q.

A transition in a WK-automaton can be defined as follows: For $\binom{x_1}{x_2}, \binom{u_1}{u_2}, \binom{y_1}{y_2} \in \binom{V^*}{V^*}$ with $\begin{bmatrix} x_1u_1y_1 \\ x_2u_2y_2 \end{bmatrix} \in WK_\rho(V)$, and $q_1, q_2 \in Q$, we write

$$\binom{x_1}{x_2} q_1 \binom{u_1}{u_2} \binom{y_1}{y_2} \Longrightarrow_M \binom{x_1}{x_2} \binom{u_1}{u_2} q_2 \binom{y_1}{y_2}$$

if and only if $\delta(q_1, \binom{u_1}{u_2}) \ni q_2$. We denote by $\Longrightarrow^*_M$ the reflexive and transitive closure of the relation $\Longrightarrow_M$. If there is no confusion, we use $\Longrightarrow$ instead of $\Longrightarrow_M$.

A *molecular language* and a (*string*) *language* over V recognized by M are defined by

$$LM(M) = \{ \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \in WK_\rho(V) \mid q_0 \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \Longrightarrow_M^* \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} q_f,\ q_f \in F\}$$

and $L(M)$ is the set of all *upper* components of the molecular language $LM(M)$.

A WK-automaton $M = (V, \rho, Q, q_0, F, \delta)$ is 1-*limited* if for any transition $\delta(q_1, \binom{x_1}{x_2}) \ni q_2$, $|x_1 x_2| = 1$ holds.

Let $\mathcal{WK}_m$ and $1\mathcal{WK}_m$ be the classes of molecular languages recognized by WK-automata and 1-limited WK-automata, resp. Further, $\mathcal{WK}$ and $1\mathcal{WK}$ denote their corresponding string language classes.

Theorem 1. ([3]) $\mathcal{WK}_m = 1\mathcal{WK}_m$ (and $\mathcal{WK} = 1\mathcal{WK}$).

[External contextual grammars ([2])]
An *external contextual grammar* is a construct $G = (V, A, C)$, where V is an alphabet, A ($\subseteq V^*$) is a finite set of axioms, C is a finite set of elements in $V^* \times V^*$. For $\alpha, \beta \in V^*$, we write $\alpha \Longrightarrow_G \beta$ if $\beta = u\alpha v$, for some $(u, v) \in C$.

An *external contextual language generated by* G is

$$L(G) = \{w \in V^* \mid x_1 \Longrightarrow_G^* w,\ x_1 \in A\}.$$

Let $\mathcal{EC}$ be the class of external contextual languages.

Theorem 2. ([2]) *It holds that* $\mathcal{EC} = \mathcal{MLIN}$ (*the class of minimal linear languages*).

[Twin-shuffle languages and their extensions]
Let V and $\bar{V} = \{\bar{a} \mid a \in V\}$ be alphabets. A *twin-shuffle language* over V is defined as

$$TS(V) = \bigcup_{x \in V^*} x \sqcup\!\sqcup \bar{x}, \quad \text{where}$$

$$x \sqcup\!\sqcup y = \{x_1 y_1 \cdots x_n y_n \mid x = x_1 \cdots x_n,\ y = y_1 \cdots y_n,\ n \geq 1,\ 1 \leq i \leq n,\ x_i, y_i \in V^*\}.$$

Consider alphabets V, $\bar{V}$ and V', where $V \cap V' = \phi$ and $\bar{V} \cap V' = \phi$. We define an *extended twin-shuffle language* over V and V' as follows :

$$ETS(V, V') = \{x_1 y_1 \cdots x_n y_n \mid n \geq 1, \text{ for } 1 \leq i \leq n,\ x_i \in TS(V),\ y_i \in V'^*\}.$$

3 Two Specific Mappings: Linearizer and Doubler

In order to materialize our goal of providing an unified view of WK-automata and sticker systems, we newly introduce two specific mappings : one is a mapping

that linearizes a given molecular language (consisting of elements in $W_\rho(V)$) into its coded form of string language, and the other is the one that, given a string language, transforms into its double stranded version of molecular language.

[Linearizer mapping: f_ℓ]
In this paper, for an alphabet V let $\bar{V} = \{\bar{a} \mid a \in V\}$, and we assume that $\rho \subseteq V \times \bar{V}$ is a complementary symmetric relation and for any $a \in V$, $(a, \bar{a}) \in \rho$ and $\bar{\bar{a}} = a$. We first define a mapping f_ℓ to transform double strands into strings.

Let $\Sigma = V \cup \bar{V}$, then we introduce new notations : for $a \in \Sigma$, $\binom{a}{\epsilon} = \hat{a}$, $\binom{\epsilon}{a} = \check{a}$, $\binom{a}{\bar{a}} = \tilde{a}$. Further, let $\hat{\Sigma} = \{\hat{a} \mid a \in \Sigma\}$, $\check{\Sigma} = \{\check{a} \mid a \in \Sigma\}$, $\tilde{\Sigma} = \{\tilde{a} \mid a \in \Sigma\}$. Now, we define the *linearizer mapping* f_ℓ:

$$f_\ell : W_\rho(V)^* \to (\hat{\Sigma} \cup \check{\Sigma})^* \ \tilde{\Sigma}^+ \ (\hat{\Sigma} \cup \check{\Sigma})^* \ \cup \ (\hat{\Sigma} \cup \check{\Sigma})^*,$$

which transforms double strands over Σ to single strands over $\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma}$.

For example, for double strands $u = \boxed{\begin{matrix} u_1 & u_2 \\ \bar{u}_2 & u_3 \end{matrix}}$ (resp. $u = \boxed{\begin{matrix} u_2 & u_3 \\ u_1 & \bar{u}_2 \end{matrix}}$), our intention is that $f_\ell(u) = \hat{u}_1\tilde{u}_2\check{u}_3$, (resp. $f_\ell(u) = \check{u}_1\tilde{u}_2\hat{u}_3$).

Formally, for a double strand $\binom{x_1}{x_2}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\binom{z_1}{z_2}$, we define $f_\ell(\binom{x_1}{x_2}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\binom{z_1}{z_2}) = x\tilde{y_1}z$, where $x = \begin{cases} \hat{x}_1 & \text{if } x_2 = \epsilon, \\ \check{x}_2 & \text{if } x_1 = \epsilon, \end{cases}$ $z = \begin{cases} \hat{z}_1 & \text{if } z_2 = \epsilon, \\ \check{z}_2 & \text{if } z_1 = \epsilon. \end{cases}$

For a double strand $\binom{a}{\epsilon} = \hat{a}$ with $a \in V$, a double strand $\binom{\epsilon}{\bar{a}} = \check{\bar{a}}$ is complementary, in the sense that $\binom{a}{\epsilon}\binom{\epsilon}{\bar{a}} = \binom{a}{\bar{a}}$. Therefore, for $\hat{a}$ in $\hat{\Sigma}$ and $\check{a}$ in $\check{\Sigma}$, we consider a complementary relation defined by ψ as follows: $\psi(\hat{a}) = \check{b}$, $\psi(\check{a}) = \hat{b}$, where $(a, b) \in \rho$, i.e., $b = \bar{a}$. In $\hat{\Sigma}$ and $\check{\Sigma}$, we consider this complementary relation defined by ψ.

Thus, a twin-shuffle language $TS(\hat{\Sigma})$ is defined as follows:

$$TS(\hat{\Sigma}) = \bigcup_{x \in \hat{\Sigma}^*} x \sqcup\!\sqcup \psi(x).$$

[Doubler mapping: f_d]
Conversely, we want to reconstruct a double strand from a string over $\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma}$.

Consider a string $y = y_1 a y_2 \in (\hat{\Sigma} \cup \check{\Sigma})^*$ with length n. For an alphabet $\hat{\Sigma}$, we say that a symbol a is *$\hat{\Sigma}$-occurrence at position i of y* with $1 \leq i \leq n$ if $|y_1|_{\hat{\Sigma}} = i-1$ and a is in $\hat{\Sigma}$, where $|x|_V$ is the number of symbols in V in the string x.

Let y be a string in $TS(\hat{\Sigma})$ with length $2m \geq 2$. Consider a complete double strand of length m which satisfies the following conditions:

- if $a \in \Sigma$ is the i-th symbol with $1 \leq i \leq m$ in the upper strand, then $\hat{a}$ is $\hat{\Sigma}$-occurrence at position i of y.

– if $a \in \Sigma$ is the i-th symbol with $1 \leq i \leq m$ in the lower strand, then $\check{a}$ is $\check{\Sigma}$-occurrence at position i of y.

$$f_d(y) = \begin{array}{|c|c|c|c|c|}\hline a_1 & \cdots & \bar{a}_i & \cdots & a_m \\ \hline \bar{a}_1 & \cdots & a_i & \cdots & \bar{a}_m \\ \hline \end{array} \quad y = \begin{array}{|c|c|c|c|c|c|c|c|c|}\hline \check{\bar{a}}_1 & \hat{a}_1 & \cdots & \check{a}_i & \cdots\cdots & \hat{\bar{a}}_i & \cdots & \hat{a}_m & \check{\bar{a}}_m \\ \hline \end{array}$$

The double strand thus obtained is called the *doubler of* y and denoted by $f_d(y)$. In particular, for ϵ we define $f_d(\epsilon) = \begin{bmatrix} \epsilon \\ \epsilon \end{bmatrix}$. Note that for a string $y \notin TS(\hat{\Sigma})$, $f_d(y)$ is undefined. Then, $f_d(x) = \begin{bmatrix} \epsilon \\ \epsilon \end{bmatrix}$ implies that $x = \epsilon$.

For example, for strings $\hat{a}\hat{\bar{b}}\hat{c}\check{a}\check{b}\check{\bar{c}}$, $\hat{a}\check{a}\hat{\bar{b}}\hat{c}\check{b}\check{\bar{c}}$, $\check{a}\hat{a}\hat{\bar{b}}\check{b}\check{\bar{c}}\hat{c}$, in $(\hat{\Sigma} \cup \check{\Sigma})^*$,

$$f_d(\hat{a}\hat{\bar{b}}\hat{c}\check{a}\check{b}\check{\bar{c}}) = f_d(\hat{a}\check{a}\hat{\bar{b}}\hat{c}\check{b}\check{\bar{c}}) = f_d(\check{a}\hat{a}\hat{\bar{b}}\check{b}\check{\bar{c}}\hat{c}) = \begin{bmatrix} a\bar{b}c \\ \bar{a}b\bar{c} \end{bmatrix}.$$

Lemma 1. *For a string w in $(\hat{\Sigma} \cup \check{\Sigma})^*$, $f_d(w)$ is a complete double strand if and only if w is in $TS(\hat{\Sigma})$.*

We now want to extend the mapping f_d so as to apply to strings in $ETS(\hat{\Sigma}, \tilde{\Sigma})$.

Let $y = x_1 x_2 \cdots x_{2n}$ be a string in $ETS(\hat{\Sigma}, \tilde{\Sigma})$, where $n \geq 1$, and for $1 \leq i \leq n$, $x_{2i-1} \in TS(\hat{\Sigma})$, $x_{2i} \in \tilde{\Sigma}^*$. Then, a doubler mapping f_d is extended in the following manner.

– For a string $x_{2i} = \tilde{u}_{2i}$ in $\tilde{\Sigma}^*$, $f_d(x_{2i})$ is a complete double strand $\begin{bmatrix} u_{2i} \\ \bar{u}_{2i} \end{bmatrix}$.

– For a string x_{2i-1} in $TS(\hat{\Sigma})$, $f_d(x_{2i-1})$ is the same one as already defined.

In a graphical representation, this means the following :

$$f_d(y) = f_d(x_1)f_d(\tilde{u}_2)\cdots f_d(x_{2n-1})f_d(\tilde{u}_{2n}) = \begin{array}{|c|c|c|c|c|}\hline f_d(x_1) & \dfrac{u_2}{\bar{u}_2} & \cdots & f_d(x_{2n-1}) & \dfrac{u_{2n}}{\bar{u}_{2n}} \\ \hline \end{array}$$

Note that for a string $y \notin ETS(\hat{\Sigma}, \tilde{\Sigma})$, $f_d(y)$ is undefined.

For example, for strings $\hat{a}\hat{\bar{b}}\check{a}\check{b}\tilde{c}\hat{d}\check{d}$, $\hat{a}\check{a}\hat{\bar{b}}\hat{c}\check{b}\check{\tilde{c}}\tilde{d}$, $\tilde{a}\hat{\tilde{b}}\tilde{c}\tilde{d}$ in $(\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma})^*$,

$$f_d(\hat{a}\hat{\bar{b}}\check{a}\check{b}\tilde{c}\hat{d}\check{d}) = f_d(\hat{a}\check{a}\hat{\bar{b}}\hat{c}\check{b}\check{\tilde{c}}\tilde{d}) = f_d(\tilde{a}\hat{\tilde{b}}\tilde{c}\tilde{d}) = \begin{bmatrix} a\bar{b}cd \\ \bar{a}b\bar{c}\bar{d} \end{bmatrix}.$$

Note 1. f_d is different from ℓ_p (in [5]) in that ℓ_p has no $\tilde{\Sigma}$ for its alphabet, and f_d has more flexibility of y than ℓ_p to build up a double strand $f_d(y)$.

Lemma 2. *For a string w in $(\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma})^*$, $f_d(w)$ is a complete double strand if and only if w is in $ETS(\hat{\Sigma}, \tilde{\Sigma})$.*

4 Characterization Results in Terms of Doubler

In this section, by using the doubler mapping f_d, we characterize languages recognized by a Watson-Crick finite automaton and generated by a sticker system.

4.1 WK Molecular Languages Are f_d(Regular Languages)

Lemma 3. *For a Watson-Crick finite automaton M_W, there exists a finite automaton M such that $LM(M_W) = f_d(L(M)) = \{f_d(w) \mid w \in L(M)\}$.*

Proof. We may consider a 1-limited WK-automaton $M_W = (\Sigma, \rho, Q, q_0, F, \delta_W)$. Then, construct a finite automaton $M = (\hat{\Sigma} \cup \check{\Sigma}, Q, q_0, F, \delta)$ *derived from* M_W as follows: For $\delta_W(q_i, x) \ni q_j$ in M_W, construct $\delta(q_i, f_\ell(x)) \ni q_j$ in M.

It suffices to show that $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ is in $LM(M_W)$ if and only if there exists a string $w \in L(M)$ such that $f_d(w) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$.

Assume that $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ is in $LM(M_W)$ and there exists a transition,

$$\binom{u_1 \cdots u_i}{v_1 \cdots v_i} q_i \binom{u_{i+1}}{v_{i+1}} \binom{u_{i+2} \cdots u_n}{v_{i+2} \cdots v_n} \Longrightarrow_{M_W} \binom{u_1 \cdots u_i}{v_1 \cdots v_i} \binom{u_{i+1}}{v_{i+1}} q_{i+1} \binom{u_{i+2} \cdots u_n}{v_{i+2} \cdots v_n},$$

where $n \geq 1$, $\begin{bmatrix} u_1 \cdots u_n \\ v_1 \cdots v_n \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$, $0 \leq i \leq n$, and $\binom{u_j}{v_j} \in \binom{\Sigma}{\epsilon} \cup \binom{\epsilon}{\Sigma}$, for $1 \leq j \leq n$, and $q_n \in F$.

From the way of constructing δ, for each $0 \leq i \leq n$, there exists a transition $\delta(q_i, b_{i+1}) \ni q_{i+1}$ in M, where $b_{i+1} = f_\ell(\binom{u_{i+1}}{v_{i+1}})$. Then, there exists a transition $\delta(q_0, b_1 \cdots b_n) \ni q_n$ in M.

Since $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ is the complete double strand, the i-th symbol in the upper strand and the i-th symbol in the lower strand are complementary. Therefore, for the string $w = b_1 \cdots b_n$, $\hat{\Sigma}$-occurrence at position i of w and $\check{\Sigma}$-occurrence at position i of w are complementary. Then, it holds that $b_1 \cdots b_n \in \hat{u}_1 \cdots \hat{u}_n \amalg \check{v}_1 \cdots \check{v}_n$, which leads to that $f_d(b_1 \cdots b_n) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$.

Conversely, assume that a string w is in $L(M)$ such that $f_d(w) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$.

Let $w = b_1 \cdots b_{2n}$, where $n \geq 1$, for $1 \leq i \leq 2n$, $b_i \in \hat{\Sigma} \cup \check{\Sigma}$, then from Lemma 1, w is in $TS(\hat{\Sigma})$. Let $w \in \hat{u}_1 \cdots \hat{u}_n \amalg \check{v}_1 \cdots \check{v}_n$.

There exists a transition $\delta(q_0, b_1 \cdots b_{2n}) \ni q_{2n}$ with $q_{2n} \in F$. From the way of constructing δ, for a transition $\delta(q_i, b_{i+1}) \ni q_{i+1}$ in M, there exists a transition $\delta_W(q_i, b'_{i+1}) \ni q_{i+1}$, where $0 \leq i \leq 2n-1$, $b_{i+1} = f_\ell(b'_{i+1})$.

Then, there exists a transition in M_W, $q_0 \binom{u_1 \cdots u_n}{v_1 \cdots v_n} \Longrightarrow^*_{M_W} \binom{u_1 \cdots u_n}{v_1 \cdots v_n} q_f$, where q_f is in F. Since w is in $TS(\hat{\Sigma})$, for each $1 \leq i \leq n$, u_i and v_i are complementary, which means that $\begin{bmatrix} u_1 \cdots u_n \\ v_1 \cdots v_n \end{bmatrix} \in WK_\rho(V)$. □

Lemma 4. *For a finite automaton $M = (\hat{\Sigma} \cup \check{\Sigma}, Q, q_0, F, \delta)$, there exists a Watson-Crick finite automaton $M_W = (\Sigma, \rho, Q, q_0, F, \delta_W)$ such that $LM(M_W) = f_d(L(M)) = \{f_d(w) \mid w \in L(M)\}$.*

Proof (*sketch*). For a finite automaton $M = (\hat{\Sigma} \cup \check{\Sigma}, Q, q_0, F, \delta)$, construct a WK-automaton $M_W = (\Sigma, \rho, Q, q_0, F, \delta_W)$ as follows:

For a transition $\delta(q_i, \hat{a}) \ni q_j$ in M, construct $\delta_W(q_i, \begin{pmatrix} a \\ \epsilon \end{pmatrix}) \ni q_j$ in M_W.

For a transition $\delta(q_i, \check{a}) \ni q_j$ in M, construct $\delta_W(q_i, \begin{pmatrix} \epsilon \\ a \end{pmatrix}) \ni q_j$ in M_W.

It suffices to show that a complete double strand $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ is in $LM(M_W)$ if and only if there exists a string $w \in L(M)$ such that $f_d(w) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$, which is proved in a manner similar to the above lemma. Thus, we can prove the equation $LM(M_W) = \{f_d(w) \mid w \in L(M)\}$. □

From Lemmas 3 and 4, we have the following theorem.

Theorem 3. *A molecular language L is in $\mathcal{WK}_m$ if and only if there exists a regular language R such that $L = f_d(R)$.*

4.2 Sticker Molecular Languages Are f_d(Minimal Linear Languages)

We slightly extend f_d to f'_d as follows : For $w = xyz$ in $(\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma})^*$ such that only $f_d(y)$ is well-defined and x, z are in $\hat{\Sigma}^* \cup \check{\Sigma}^*$, define $f'_d(w) = x' f_d(y) z'$, where x' (z') represents that x (z) forms an "upper stand" if x (z) is in $\hat{\Sigma}^*$ or "lower one" otherwise.

Lemma 5. *For a sticker system γ_W, there exists an external contextual grammar G such that $LM(\gamma_W) = f_d(L(G)) = \{f_d(w) \mid w \in L(G)\}$.*

Proof. For a sticker system $\gamma_W = (\Sigma, \rho, A_W, D_W)$, we define an external contextual grammar $G = (\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma}, A, C)$ *derived from* γ_W as follows:

For (u, v) in D_W, construct $(f_\ell(u), f_\ell(v))$ in C. Let $A = \{f_\ell(\alpha) \mid \alpha \in A_W\}$.

It suffices to show that for any $\alpha' = \begin{pmatrix} \alpha'_1 \\ \alpha'_2 \end{pmatrix}$ in $LR_\rho(\Sigma)$, there exists a computation $\begin{pmatrix} \alpha'_1 \\ \alpha'_2 \end{pmatrix} \Longrightarrow^n_{\gamma_W} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ if and only if there exists a computation $\alpha'' \Longrightarrow^n_G z$, where $f'_d(\alpha'') = \alpha'$ and $f_d(z) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$.

We will prove this by the induction on n.

Base step : ($n = 0$) It trivially holds. *Induction step* : There exists a computation

$$\begin{pmatrix} \alpha'_1 \\ \alpha'_2 \end{pmatrix} \Longrightarrow_{\gamma_W} \begin{pmatrix} u \\ u' \end{pmatrix} \begin{pmatrix} \alpha'_1 \\ \alpha'_2 \end{pmatrix} \begin{pmatrix} v \\ v' \end{pmatrix} \Longrightarrow^n_{\gamma_W} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

iff (by inductive hypothesis and the way of constructing C) there uniquely exist $(f_\ell(\begin{pmatrix} u \\ u' \end{pmatrix}), f_\ell(\begin{pmatrix} v \\ v' \end{pmatrix}))$ in C such that

$$f_\ell(\alpha') \Longrightarrow_G f_\ell(\begin{pmatrix} u \\ u' \end{pmatrix}) f_\ell(\alpha') f_\ell(\begin{pmatrix} v \\ v' \end{pmatrix}) = w \quad \text{and} \quad w \Longrightarrow_G^n z,$$

where $f'_d(w) = \begin{pmatrix} u \\ u' \end{pmatrix} \begin{pmatrix} \alpha'_1 \\ \alpha'_2 \end{pmatrix} \begin{pmatrix} v \\ v' \end{pmatrix}$ and $f_d(z) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$

iff there exists

$$f_\ell(\alpha') \Longrightarrow_G f_\ell(\begin{pmatrix} u \\ u' \end{pmatrix}) f_\ell(\alpha') f_\ell(\begin{pmatrix} v \\ v' \end{pmatrix}) \Longrightarrow_G^n z, \text{ and } f_d(z) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}.$$

Considering $\alpha' = \alpha \in A_W$, we have that

$$\alpha \Longrightarrow_{\gamma_W}^n \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \text{ if and only if } f_\ell(\alpha) \Longrightarrow_G^n z, \text{ where } f_d(z) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}. \qquad \square$$

An external contextual grammar $G = (\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma}, A, C)$ is said to be *restricted* if (1) for any (u, v) in C, u and v are in $(\hat{\Sigma} \cup \check{\Sigma})^* \ \tilde{\Sigma}^+ \ (\hat{\Sigma} \cup \check{\Sigma})^* \ \cup \ (\hat{\Sigma} \cup \check{\Sigma})^*$, and (2) $A \subset (\hat{\Sigma} \cup \check{\Sigma})^* \ \tilde{\Sigma}^+ \ (\hat{\Sigma} \cup \check{\Sigma})^*$.

Let $r\text{-}\mathcal{EC}$ be the class of languages generated by restricted external contextual grammars.

Lemma 6. *For a given restricted external contextual grammar* $G = (\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma}, A, C)$, *there exists a sticker system* $\gamma_W = (\Sigma, \rho, A_W, D_W)$ *such that* $LM(\gamma_W) = \{f_d(w) \mid w \in L(G)\}$.

Proof (*sketch*). For a given G above, we construct a sticker system $\gamma_W = (\Sigma, \rho, A_W, D_W)$ as follows : For (x, y) in C, construct $(f'_d(x), f'_d(y))$ in D_W. Let $A_W = \{f'_d(\alpha) \mid \alpha \in A\}$.

We can prove that $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ is in $LM(\gamma_W)$ if and only if there exists a string $w \in L(G)$ such that $f_d(w) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$, in a manner similar to the above lemma, which implies the equation $LM(\gamma_W) = \{f_d(w) \mid w \in L(G)\}$. $\square$

From Lemmas 5 and 6, we have the following theorem.

Theorem 4. *A molecular language* L *is in* $\mathcal{SL}_m$ *if and only if there exists an external contextual language* R *in* $r\text{-}\mathcal{EC}$ *such that* $L = f_d(R)$.

4.3 Characterizing Recursively Enumerable Languages by f_d

Based on the doubler mapping f_d and a projection, we first introduce a mapping f_{pr}. For the projection $pr_T : V_2^* \to T^*$, we define $f_{pr} : (\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma})^* \to T^*$ as follows: $f_{pr}(w) = pr_T(x)$, where $f_d(w) = \begin{bmatrix} x \\ x' \end{bmatrix}$ for some $x' \in \Sigma^*$. Then, using the class $\mathcal{EC}$ and f_{pr}, we have the following characterization of recursively enumerable languages.

Theorem 5. *For a recursively enumerable language L, there exists an external contextual grammar G such that $f_{pr}(L(G)) = L$.*

Proof. It is well known that for any recursively enumerable language $L \subseteq T^*$, there exist two ϵ-free morphisms h_1, h_2, a regular language R, and a projection pr_T such that $L = pr_T(h_1(EQ(h_1, h_2)) \cap R)$. Let $M = (Q, \Gamma_2, \delta, q_0, F)$ be a finite automaton such that $L(M) = R$. Let $h_1, h_2 : V_1^* \to V_2^*$.

We construct an external contextual grammar $G = (\hat{\Sigma} \cup \check{\Sigma} \cup \tilde{\Sigma}, A, C)$:

1. Let $(w_a, u_a v_a)$ be in C, where $u_a = \hat{b}_1 \cdots \hat{b}_n$, with $h_1(a) = b_1 \cdots b_n$, $v_a = \check{c}_1 \cdots \check{c}_m$, with $h_2(a) = c_1 \cdots c_m$,
   for any q_1 in Q, $w_a = \hat{q}_{n+1}\tilde{\#}\check{q}_n\ \hat{q}_n\tilde{\#}\check{q}_{n-1} \cdots \hat{q}_3\tilde{\#}\check{q}_2\ \hat{q}_2\tilde{\#}\check{q}_1$,
   if $\delta(q_i, b_i) \ni q_{i+1}$ for each $1 \le i \le n$, $q_{n+1} \notin F$,
   $w_a = \tilde{q}_{n+1}\tilde{\#}\check{q}_n\ \hat{q}_n\tilde{\#}\check{q}_{n-1} \cdots \hat{q}_3\tilde{\#}\check{q}_2\ \hat{q}_2\tilde{\#}\check{q}_1$,
   if $\delta(q_i, b_i) \ni q_{i+1}$ for each $1 \le i \le n$, $q_{n+1} \in F$.

   The strings u_a, v_a are used to check the equality for the homomorphisms h_1 and h_2. The string w_a corresponds to the brick $\begin{array}{|l} \hline q_{n+1}\ \#q_n\#\quad \cdots q_3\#q_2\# \\ \hline \quad \#\bar{q}_n\#\bar{q}_{n-1} \cdots \#\bar{q}_2\#\ \bar{q}_1 \\ \hline \end{array}$ which is used to check whether a string is in $R = L(M)$.

2. Let $\Sigma = \Gamma_Q \cup \Gamma_2 \cup \Gamma_\#$ for $\Gamma_Q = Q \cup \bar{Q}$, $\Gamma_2 = V_2 \cup \bar{V}_2$, $\Gamma_\# = \{\#, \bar{\#}\}$ and let $A = \{\hat{q}_0\tilde{\#}\}$.

We will show the equality $f_{pr}(L(G)) = L$. Assume that w is in $f_{pr}(L(G))$, then there exists a string w' in $L(G)$ such that $f_{pr}(w') = w$. From the definition of C, $w' = w_1\hat{q}_0\tilde{\#}w_2$, where $w_1 \in (\hat{\Gamma}_Q \cup \check{\Gamma}_Q \cup \{\tilde{\#}\})^*$, $w_2 \in (\hat{\Gamma}_2 \cup \check{\Gamma}_2)^*$. Since $f_{pr}(w')$ is defined, w' is in $ETS(\hat{\Sigma}, \tilde{\Sigma})$. Further, $w_1 \in ETS(\hat{\Gamma}_Q, \tilde{\Gamma}_Q \cup \{\tilde{\#}\})$, $w_2 \in TS(\hat{\Gamma}_2)$. Then, there must exist a string $z = a_1 \cdots a_m \in \Gamma_1^*$ which satisfies the following two conditions : for $h_1(z) = b_1 \cdots b_{m'}$ with $m' \ge 1$,

- $w_1 = \tilde{q}_{m'}\tilde{\#}\check{q}_{m'-1}\ \hat{q}_{m'-1}\tilde{\#}\check{q}_{m'-2} \cdots \hat{q}_2\tilde{\#}\check{q}_1\ \hat{q}_1\tilde{\#}\check{q}_0$, where for $0 \le i \le m' - 1$, $\delta(q_i, b_{i+1}) \ni q_{i+1}$, $q_{m'} \in F$. This implies that $b_1 \cdots b_{m'}$ is in R.
- $w_2 \in \hat{b}_1 \cdots \hat{b}_{m'} \sqcup\!\sqcup \check{b}_1 \cdots \check{b}_{m'}$ This implies that $h_1(z) = h_2(z)$.

Therefore, $b_1 \cdots b_{m'}$ is in $h_1(EQ(h_1, h_2)) \cap R$, then from the definition of f_{pr}, $f_{pr}(w') = pr_T(b_1 \cdots b_{m'}) \in L$.

Conversely, assume that w is in L. Then, there exists a string z such that $z \in h_1(EQ(h_1, h_2))$, $z \in R$ and $pr_T(z) = w$. Then, there exists a string $z' = a_1 \cdots a_m$ such that $z' \in EQ(h_1, h_2)$ and $h_1(z') = h_1(a_1) \cdots h_1(a_m) = b_1 \cdots b_{m'} = h_2(a_1) \cdots h_2(a_m)$ for $m' \ge 1$.

For z', there exists a derivation $\hat{q}_0\tilde{\#} \Longrightarrow_G^* w_1\hat{q}_0\tilde{\#}w_2$ in G such that $w_2 = \hat{h}_1(a_1)\check{h}_2(a_1) \cdots \hat{h}_1(a_m)\check{h}_2(a_m)$, where $\hat{h}_1(a_i) = \hat{b}_{i1} \cdots \hat{b}_{ik}$ for $h_1(a_i) = b_{i1} \cdots b_{ik}$, $\check{h}_2(a_i) = \check{c}_{i1} \cdots \check{c}_{i\ell}$ for $h_2(a_i) = c_{i1} \cdots c_{i\ell}$. Then, $w_2 \in \hat{b}_1 \cdots \hat{b}_{m'} \sqcup\!\sqcup \check{b}_1 \cdots \check{b}_{m'}$.

At the same time, since $\delta(q_0, z) \ni q_f$ with $q_f \in F$, from the way of constructing C, $w_1 = \tilde{q}_{m'}\tilde{\#}\check{q}_{m'-1}\ \hat{q}_{m'-1}\tilde{\#}\check{q}_{m'-2}\ \cdots\ \hat{q}_2\tilde{\#}\check{q}_1\ \hat{q}_1\tilde{\#}\check{q}_0$, where for $0 \le i \le m' - 1$, $\delta(q_i, b_{i+1}) \ni q_{i+1}$, $q_{m'} \in F$. Therefore, $w_1\hat{q}_0\tilde{\#}w_2$ is in $ETS(\hat{\Sigma}, \tilde{\Sigma})$. Then, from the definition of f_{pr}, we have $f_{pr}(w_1\hat{q}_0\tilde{\#}w_2) = w$. □

4.4 WK Molecular Languages Are Proj(Sticker Molecular Languages)

Let us define a *double-strand projection* (abb. d-projection) *d-pr* on double strands as follows : For z in $\begin{bmatrix} V_1 \cup V_2 \\ V_1 \cup V_2 \end{bmatrix}$, $d\text{-}pr(z) = z$ if z is in $\begin{bmatrix} V_1 \\ V_1 \end{bmatrix}$ and $d\text{-}pr(z) = \begin{bmatrix} \epsilon \\ \epsilon \end{bmatrix}$ otherwise.

Lemma 7. *For any Watson-Crick finite automaton M, there exists a sticker system γ such that $LM(M) = d\text{-}pr(LM(\gamma))$.*

Proof. (*sketch*) From Theorem 1, we may consider a 1-limited WK-automaton $M = (\Sigma, \rho, Q, q_0, F, \delta)$. Based on M, construct a sticker system $\gamma = (\Sigma \cup \Gamma_Q \cup \Gamma_\#, \rho_Q, A, D)$ as follows : $\Gamma_Q = Q \cup \bar{Q}$, $\Gamma_\# = \{\#, \bar{\#}\}$. For a transition $\delta(q_i, \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}) \ni q_j$ in M with $q_j \notin F$, construct $(\begin{pmatrix} q_j \\ \epsilon \end{pmatrix} \begin{bmatrix} \# \\ \bar{\#} \end{bmatrix} \begin{pmatrix} \epsilon \\ \bar{q}_i \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \end{pmatrix})$ in D. For a transition $\delta(q_i, \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}) \ni q_f$ with $q_f \in F$, construct $(\begin{bmatrix} q_f \\ \bar{q}_f \end{bmatrix} \begin{bmatrix} \# \\ \bar{\#} \end{bmatrix} \begin{pmatrix} \epsilon \\ \bar{q}_i \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \end{pmatrix})$ in D. Finally, let $A = \{\begin{pmatrix} q_0 \# \\ \epsilon \ \bar{\#} \end{pmatrix}\}$. Consider a d-projection *d-pr* on the alphabet $\begin{bmatrix} \Sigma \\ \bar{\Sigma} \end{bmatrix}$. From the way of constructing γ, by the induction on the length of a computation, we can prove that for $q_f \in F$, $q_0 \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \Longrightarrow_M^* \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} q_f$ if and only if there exists a computation $\begin{pmatrix} q_0 \# \\ \epsilon \ \bar{\#} \end{pmatrix} \Longrightarrow_\gamma^* \begin{bmatrix} q_f \\ \bar{q}_f \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \begin{bmatrix} \# \\ \bar{\#} \end{bmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$, where $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \in \begin{bmatrix} \#Q \\ \bar{\#}\bar{Q} \end{bmatrix}^+$. Finally, from the definition of *d-pr*, it holds that a complete double strand $\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ is in $LM(M)$ if and only if there exists a complete double strand $\begin{bmatrix} q_f \\ \bar{q}_f \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \begin{bmatrix} \# \\ \bar{\#} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \in LM(\gamma)$ such that $d\text{-}pr(\begin{bmatrix} q_f \\ \bar{q}_f \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \begin{bmatrix} \# \\ \bar{\#} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}) = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$. □

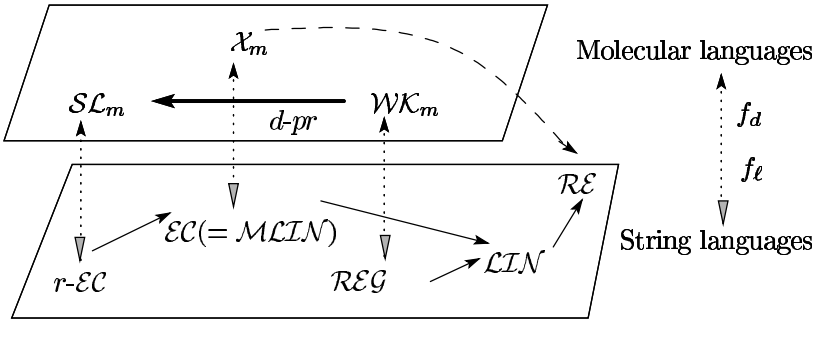

Fig. 1. Landscape of Double-Decker Families of Languages

5 Conclusion

By introducing two specific mappings called "doubler f_d" and "linearizer f_ℓ", we have given new characterization results for the families of sticker languages and of Watson-Crick languages which lead to not only an unified view of the two families of languages but also a clarified view of the computational capability of the DNA complementarity. From Theorems 1 and 5, we have the result $\mathcal{RE} = f_{pr}(\mathcal{MLIN})$, which seems shed some new insights into computations in comparison to the existing ones such as $\mathcal{RE} = dgsm(\mathcal{SL})$ or $\mathcal{RE} = coding(\mathcal{WK})$ (in [3]).

Acknowledgements

This work is supported in part by Grant-in-Aid for Scientific Research on Priority Area no.14085205, Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

1. Hoogeboom, H.J. and Vugt, N.V. : Fair sticker languages, *Acta Informatica*, **37**, 213–225 (2000).
2. Păun, Gh. : *Marcus contextual grammar*. Kluwer Academic Publishers (1997).
3. Păun, Gh., Rozenberg, G. and Salomaa, A. : *DNA Computing. New Computing Paradigms.*, Springer (1998).
4. Rozenberg, G. and Salomaa, A. (Eds.) : *Handbook of Formal Languages*, Springer (1997).
5. Salomaa, A. : *Turing, Watson-Crick and Lindenmayer : Aspects of DNA Complementarity*, In *Unconventional Models of Computation*, Auckland, Springer, 94–107 (1998).
6. Sakakibara, Y. and Kobayashi, S. : *Sticker systems with complex structures. Soft Computing*, **5**, 114–120 (2001).
7. Vliet, R. van, Hoogeboon, H.J. and Rozenberg, G. : *Combinatorial Aspects of Minimal DNA Expressions*, Pre-proc. In *Tenth International Meeting on DNA Computing*, Univ. of Milano-Bicocca, Italy, 84–96 (2004).