On the Power of Circular Splicing Systems and DNA Computability

Takashi YOKOMORI Satoshi KOBAYASHI

Dept. of Computer Science and Information Mathematics
University of Electro-Communications
1-5-1 Chofugaoka, Chofu, Tokyo 182, JAPAN
Tel.: +81-424-83-2161 ex.4375, Fax.: +81-424-82-3055
yokomori@cs.uec.ac.jp, satoshi@cs.uec.ac.jp
and
Claudio FERRETTI

Department of Computer Science
University of Milano
via Comelico 39, 20135 Milano, ITALY
ferretti@imiucca.csi.unimi.it

Abstract

From a biological motivation of interactions between linear and circular DNA sequences, we propose a new type of splicing models called circular H systems and show that they have the same computational power as Turing machines. It is also shown that there effectively exists a universal circular H system which can simulate any circular H system with the same terminal alphabet, which strongly suggests a feasible design for a DNA computer based on circular splicing.

1 Introduction

Since Adleman's breath-taking paper on molecular (DNA) computing ([1]), there have already been quite a few papers on this challenging topic : [10] shows how to solve NP-complete problems using DNA, while [3] discusses a design method for simulating a Turing machine by molecular biological techniques and shows how to compute PSPACE, and [4]) gives a methodology for breaking the DES using techniques in genetic engineering.

In response to the rapid stream of experimental research on this new computation paradigm, a series of papers on *theoretical* research on DNA computing has lately been attracting much attention in computer science, which has been originated by Head's work on splicing systems (or H systems) and their languages for modeling DNA recombination ([8]). Those discuss a variety of topics including the regularity of splicing DNA languages, the language-theoretic properties of circular DNA languages ([9, 15, 19]), and the universal computability of extended models of splicing systems ([11, 12, 13, 14]). Among others, [6] shows the existence of a universal extended splicing system which can simulate the behavior of any other extended splicing system.

In this paper, we introduce a new type of splicing operations to propose a new splicing model called *circular H systems*. The schematic definition for this new type of splicing is given in Figure 1 where, besides usual (linear) strings over some alphabet, circular strings are involved in the splicing operations as well.

Suppose that $x = x_1\mathtt{u}_1\mathtt{u}_2x_2$ is a *linear* string and $y = y_1\mathtt{u}_3\mathtt{u}_4y_2$ is a *circular* string over some finite alphabet Σ (see Figure 1). Then, following the notation in [6], as some of the existing papers ([9, 15, 19]) suggest, it seems to be biologically reasonable that applying a splicing rule $r = \mathtt{u}_1\#\mathtt{u}_2\$\mathtt{u}_3\#\mathtt{u}_4$ yields a longer linear string $z = x_1\mathtt{u}_1\mathtt{u}_4y_2y_1\mathtt{u}_3\mathtt{u}_2x_2$ or sometimes two linear strings $z_1 = x_1\mathtt{u}_1\mathtt{u}_4y_2$ and $z_2 = y_1\mathtt{u}_3\mathtt{u}_2x_2$, as a recombination behavior at splicing sites $\mathtt{u}_1\mathtt{u}_2$ and $\mathtt{u}_3\mathtt{u}_4$ of x and y, respectively, where # and \$ are special symbols not in Σ. It is well recognized in molecular biology that some kinds of DNA molecules may form both linear and circular strings, depending upon a biochemical circumstance.

We are mainly interested in the effects caused by interactive behaviors of linear and circular DNA sequences in the splicing mechanism. Therefore, we will consider a splicing operation which, given x and y with r, produces one linear z_1 and one circular z_2, which we call *circular splicing*. This type of splicing might also be taken as a natural (biological) modification (under the existence of sufficient quantity of ligases) of the usual splicing in the case of linear strings investigated so far in the literature. Together with this splicing mode, a *circular H system* is defined as an extended H system (studied in the existing literature) having not

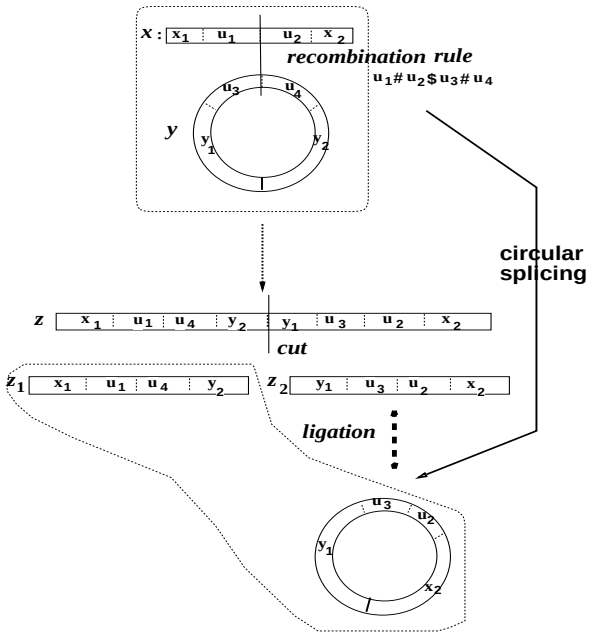

Figure 1. Circular Splicings

only a set of linear strings but also a set of *circular* axioms as well.

The motivation of introducing circular splicing rules and a new model based on the splicing mechanism may be justified at least twofold. First, it is reported by molecular biologist that, for instance, a kind of this type of splicing occurs in a recombination (transposition) mechanism between bacterial chromosomes and F plasmids in *E. Coli* in the real world. Sometimes *transposons*, DNA sequences able to insert themselves at a new location in the genome without any sequence relationship to the target locus, could cause DNA recombinations similar to the circular splicing mentioned above.

Second, from the practical viewpoint of designing "DNA computers" based on this computation mechanism of circular splicing, we believe that the model of circular H systems defined as above has a certain advantage over any other existing type of splicing model, in that a variety of techniques in state-of-the-art molecular biology will make the "circular computation" via circular splicing more feasible one.

The goal of this paper is to show that (i) circular H systems are computationally equivalent to Turing machines, (ii) there is a universal circular H system which can simulate the behavior of any other circular H system, (iii) there is a simple normal form for the class of circular H systems. Surprisingly, all these results are obtained without considering multiplicity constraint, which is in marked contrast to the previous results for linear H systems.

Because of its simplicity of the splicing mechanism, the fact of the existence of the universal circular H system directly implies a feasible implementation procedure for building DNA (test tube) computers in vitro. These practical issues on DNA computer architecture should be discussed in great detail somewhere else, but is beyond the topic covered by this paper.

2 Preliminary

Following the notations and definitions in [7] and [6], V^* is the set of all (finite length) strings over a finite alphabet V. The empty string is denoted by λ, and $V^+ = V^* - \{\lambda\}$. For a string $x \in V^*$, $|x|$ denotes the length of x, i.e., the number of symbols from V comprising x. A language is any set of strings over a finite alphabet. A language is λ-free if it does not contain the empty string λ. *Throughout this paper, we assume that any language is λ-free.* The family of recursively enumerable languages is denoted by $\mathcal{RE}$.

We will introduce Post systems originally proposed in [16] but here we are only concerned with their special forms, because it is sufficient for our purpose.

A *Post Normal system* (PN system) is a quadruple $G = (V, \Sigma, P, A)$, where V and Σ are finite alphabets such that $\Sigma \subseteq V$, and P is a finite set of rules of the form : $uX \to Xw$ (X: unique variable not in V, $u, w \in V^*$). $A(\subset V^+)$ is a finite set of *axioms*. (Note that G has only one variable X, and V is the alphabet of constants in the definition of Post systems.)

Given strings $\alpha, \beta \in V^*$, a binary relation $\Longrightarrow$ is defined as follows:

$$\alpha \Rightarrow \beta \text{ iff } \exists uX \to Xw \in P, \delta \in V^*[\alpha = u\delta, \beta = \delta w].$$

The reflexive and transitive closure of $\Longrightarrow$ is denoted by $\Longrightarrow^*$. For a given G, a language generated by G is defined as

$$L(G) = \{w \in \Sigma^* | \exists u \in A\ [u \Longrightarrow^* w]\}.$$

By $POST$, we denote the family of languages generated by Post Normal systems.

Lemma 1 ([18]) $POST = \mathcal{RE}$.

In order to prove our first result, we need some more preliminary results about Post systems.

A *generalized regular Post system* (GRP system) is a quadruple $G = (V, \Sigma, P, A)$, where V and Σ are finite alphabets such that $\Sigma \subseteq V$, and P is a finite set of rules whose forms are either $uX \to wX$ or $aX \to Xb$ (X: unique variable not in V, $u, w \in V^*$, $a, b \in V$). $A(\subset V^+)$ is a finite set of *axioms*.

Given strings $\alpha, \beta \in V^*$, a binary relation $\Longrightarrow$ is defined as follows:

$\alpha \Longrightarrow \beta$ iff
either $\exists uX \to wX \in P, \delta \in V^*$ $[\alpha = u\delta, \beta = w\delta]$
or $\exists aX \to Xb \in P, \delta \in V^*$ $[\alpha = a\delta, \beta = \delta b]$.

The reflexive and transitive closure of $\Longrightarrow$ is denoted by $\Longrightarrow^*$. For a given G, a language generated by G is defined as

$$L(G) = \{w \in \Sigma^* | \exists u \in A\ [u \Longrightarrow^* w]\}.$$

By $GRPS$, we denote the family of languages generated by generalized regular Post systems. Then, we can easily prove the following theorem.

Lemma 2 *For a given PN system G, there effectively exists a GRP system G' such that $L(G) = L(G')$.*

From Lemma 1 and Turing-Church thesis, we have the following :

Corollary 3 $POST = GRPS = \mathcal{RE}$.

3 Circular H Systems and Their Languages

[Circular Strings]

Following [9] and [19], a *circular string* consisting of $a_1, ..., a_m$ in this order is denoted by

$$\hat{}a_1 \cdots a_m$$

and one of its linearized froms is : $a_1 \cdots a_m$. (The symbol $\hat{}$ is reserved to indicate circularity.) When we want to indicate that a string x is circular, we denote it by $\hat{}x$. Note that in this notation, for example, $\hat{}$`atcg`, $\hat{}$`tcga`, $\hat{}$`cgat`, and $\hat{}$`gatc` are all notations for the same circular string. The set of all circular strings over Σ is denoted by $\Sigma\hat{}$.

3.1 Circular H Systems

We now introduce a new type of circular splicing models based on the notion of a circular splicing operation with modes.

[Circular Splicing with Modes]

For two linear strings $x, z \in V^*$ and two circular strings $\hat{}y$, $\hat{}w \in V\hat{}$, and a rule $r = u_1\#u_2\$u_3\#u_4$, we first define the *circular splicing operation* by

$$(x, \hat{}y) \models_r (z, \hat{}w) \text{ iff } \begin{cases} x = x_1u_1u_2x_2, \ \hat{}y = \hat{}y_1u_3u_4y_2 \\ z = x_1u_1u_4y_2, \ \hat{}w = \hat{}y_1u_3u_2x_2, \quad ... \quad (*) \\ \text{for some } x_1, x_2, y_1, y_2 \in V^* \end{cases}$$

(Note that in the circular splicing, an *ordered* pair of linear and circular strings yields another *ordered* pair of linear and circular strings with the help of one application of a splicing rule.)

In the definition (*) above, we note that since $\hat{}y$ is a circular string, for any $\hat{}y$ and $r = u_1\#u_2\$u_3\#u_4$, there is more than one possible representations of the form : $\hat{}y = y_1u_3u_4y_2$, where $y_1y_2 \neq \lambda$. This implies that the splicing result $(z, \hat{}w)$ in (*) can be different, depending on the manner of representing an identical circular string $\hat{}y$.

One of the ways of fixing this ambiguity of the definition (*) is to consider the splicing modes as follows. We say that u_3u_4 appears in y in the mode:

$$\begin{array}{lll} \textit{prefix} & \text{iff} & y = u_3u_4y', \text{ for some } y' \in V^*, \\ \textit{suffix} & \text{iff} & y = y'u_3u_4, \text{ for some } y' \in V^*. \end{array}$$

In fact, these notions have already been introduced in [14] in the context of splicing linear strings. Following [14], we denote these modes by p, s and let $M = \{p, s\}$.

We now define the circular splicing operation *with mode* as follows : For $g \in M$ and a rule $r = u_1\#u_2\$u_3\#u_4$,

$$(x, \hat{}y) \models_r^g (z, \hat{}w) \text{ iff } \begin{array}{l} u_3u_4 \text{ appears in } y \text{ in the mode } g, \text{ and} \\ (x, \hat{}y) \models_r (z, \hat{}w). \end{array}$$

A *circular H system* (CH system) is a quadruple $H = (V, \Sigma, A, R)$, where V is a finite alphabet, $\Sigma \subseteq V$, $A \subseteq V^* \cup V\hat{}$, and $R \subseteq (V^*\#V^*\$V^*\#V^*) \times M$ ($M = \{p, s\}$, and $\#, \$$ are special symbols not in V). $A = A_\ell \cup A_c$ is the set of *axioms*, where A_ℓ and A_c consisting of *linear* and *circular* strings over V, respectively, R is the set of *splicing rules with mode*.

Note that [9] has already suggested this type of splicing models, and similar types of splicing systems are also proposed and investigated under the terminology of "mixed splicing" systems but only in the context of non-extended models (i.e., without introducing terminal alphabet Σ) in [15, 19].

In what follows, we are concerned with CH system where A and R are both *non-empty finite sets*, simply because it will turn out that finite sets are already sufficient for proving the computational universality of CH systems considered in this paper.

For a CH system $H = (V, \Sigma, A, R)$, a language $L \subset V^* \cup V\hat{}$, we define

$$\delta(L) = \{z \in V^*, \ \hat{}w \in V\hat{} \mid (x, \hat{}y) \models_r^g (z, \hat{}w), \text{ for some } x, \hat{}y \in L, (r, g) \in R\},$$

and define

$$\delta^*(L) = \bigcup_{i \geq 0} \delta^i(L), \quad \begin{cases} \delta^0(L) = L \\ \delta^{i+1}(L) = \delta^i(L) \cup \delta(\delta^i(L)) \\ \qquad \text{for } i \geq 0. \end{cases}$$

The language $L_\ell(H)$ of *linear* strings over Σ is defined as follows :

$$L_\ell(H) = \delta^*(A) \cap \Sigma^*.$$

3.2 Primitive Operations by Circular Splicings

We consider what kinds of primitive operations on strings can be performed by a variety of circular splicing mechanism with modes. We analyse and select the following three as a minimal set of primitive operations, realizable in one step by the circular splicing with modes, where a symbol B in each operation works as a *block symbol*, and | indicates a position to be separated, u, w, x are in Σ^*, B and @ are in $V - \Sigma$.

1. **[Replacement]** (`REP`) : (B|@`w`, $\hat{}x$|@`u`) $\models_r^s$ (B@`u`, $\hat{}x$@`w`) for $r =$ B#@`w`$#@`u`.

   In the circular string $\hat{}$@`u`x(=$\hat{}x$@`u`), the substring `u` is replaced with `w`, producing the circular string $\hat{}$@`w`x(=$\hat{}x$@`w`).

2. **[Rotation]** (`ROT`) : (B|`w`@, $\hat{}x$|@`u`) $\models_r^s$ (B@`u`, $\hat{}x$`w`@) for $r =$ B#`w`@$#@`u`.

   In the circular string $\hat{}$@`u`x(= $\hat{}x$@`u`), the substring @`u` is replaced with `w`@, producing the circular string $\hat{}$@x`w`(= $\hat{}x$`w`@).

3. **[Cut Linearization]** (`CL`) : $(|\mathrm{B}, \hat{}@|x) \models^p_r (x, \hat{}@\mathrm{B})$ for $r = \#\mathrm{B}\$@\#$.

The circular string $\hat{}@x$ is linearized into x.

In fact, it will soon turn out that these operations are powerful enough for achieving the universal computability. Our first main result is as follows :

Lemma 4 *For any recursively enumerable language L, there effectively exists a CH system H such that $L_\ell(H) = L$.*

Proof. By Corollary 3, for any recursively enumerable language L, let $G = (V, \Sigma, P, A)$ be a GRP system such that $L = L(G)$, where we may assume that P does not have a rule of the form $uX \to X$. (Any rule of the form $uX \to X$ can be replaced with a set of rules $\{uaX \to aX | a \in V\}$. Recall that as stated in Section 2, we deal with only λ-free languages in this paper.)

Now, construct a CH system $H = (V', \Sigma, A', R)$ as follows:

$$\begin{aligned}
V' &= V \cup \{@, @', \mathrm{B}, \mathrm{B}'\}, \\
A' &= A_\ell \cup A_c \\
&= \{\mathrm{B}@\overline{w} \mid uX \to \overline{w}X \in P\} \\
&\quad \cup \{\mathrm{B}b@ \mid \overline{b}X \to Xb \in P\} \\
&\quad \cup \{@', \mathrm{B}'\} \cup \{\hat{}@x \mid x \in A\}, \text{ and} \\
R &= \{(\mathrm{B}\#@\overline{w}\$\#@u, s) \mid uX \to \overline{w}X \in P\} \\
&\qquad \text{(corresponding to } \texttt{REP}) \\
&\quad \cup \{(\mathrm{B}\#b@\$\#@\overline{b}, s) \mid \overline{b}X \to Xb \in P\} \\
&\qquad \text{(corresponding to } \texttt{ROT}) \\
&\quad \cup \{(\#\mathrm{B}'\$@\#, p)\} \text{ (corresponding to } \texttt{CL}) \\
&\quad \cup \{(@'\#\$\#@, s)\}.
\end{aligned}$$

(Symbols @ (@′) and B (B′) are special ones not in Σ to used, respectively, as a marker indicating a splicing location and as a 'block symbol' in H.)

(1) Suppose that, for $u, v, \overline{w} \in V^*$, it holds that $uv \Longrightarrow \overline{w}v$ by a rule $uX \to \overline{w}X$ in G. Then, we see that

(i) $(\mathrm{B}|@\overline{w}, \hat{}v|@u) \models^s_r (\mathrm{B}@u, \hat{}v@\overline{w})$
for $r = \mathrm{B}\#@\overline{w}\$\#@u$ and $\mathrm{B}@\overline{w} \in A_\ell$,

(ii) $(|\mathrm{B}', \hat{}@|\overline{w}v) \models^p_{r'} (\overline{w}v, \hat{}@\mathrm{B}')$
for $r' = \#\mathrm{B}'\$@\#$ and $\mathrm{B}' \in A_\ell$.

(Note that $\hat{}v@\overline{w} = \hat{}@\overline{w}v$.) Thus, we have a circular string $\hat{}@\overline{w}v$ in (i) and its linearized string $\overline{w}v$ in (ii).

(2) Suppose that, for $b \in V$, $v \in V^*$, it holds that $\overline{b}v \Longrightarrow vb$ by a rule $\overline{b}X \to Xb$ in G. Then, we see that

(i) $(\mathrm{B}|b@, \hat{}v|@\overline{b}) \models^s_r (\mathrm{B}@\overline{b}, \hat{}vb@)$
for $r = \mathrm{B}\#b@\$\#@\overline{b}$ and $\mathrm{B}b@ \in A_\ell$,

(ii) $(|\mathrm{B}', \hat{}@|vb) \models^p_{r'} (vb, \hat{}@\mathrm{B}')$
for $r' = \#\mathrm{B}'\$@\#$ and $\mathrm{B}' \in A_\ell$.

Thus, we have a circular string $\hat{}vb@(=\hat{}@vb)$ in (i) and its linearized string vb in (ii). (Note that v may contain some barred symbols $\overline{b}$.)

(3) Suppose that we have a circular string $\hat{}z@$ such that $x \Longrightarrow^* z$ for some $x \in A$ in G. Then, we see that

(iii) $(@'|, \hat{}z|@) \models^s_{r''} (@'@, \hat{}z)$
for $r'' = @'\#\$\#@$ and $@' \in A_\ell$.

Thus, we have a circular string $\hat{}z$ such that $x \Longrightarrow^* z$ for some $x \in A$ in G.

Observe that once the symbol B (B′) has been incorporated into a linear (circular) string by a splicing (i) $((ii))$, there is no way to remove it from the string. Also, once a circular string has lost the marker @ by a splicing (ii) or (iii), the resulting string has no chance to be spliced further more.

Now, let us first calculate the set of all circular strings generated by H. Then, we see that

$$\begin{aligned}
&\delta^*(A') \cap V'^{\hat{}} = \\
&\{\hat{}z@, \hat{}z | \exists x \in A\ [x \Longrightarrow^* z] \text{ in } G\ \} \cup \{\hat{}@\mathrm{B}', \hat{}\mathrm{B}'\} \\
&\cup \{\hat{}\mathrm{B}'@w_{i1} \cdots w_{ip},\ \hat{}\mathrm{B}'w_{i1} \cdots w_{ip} \mid p \geq 1, \\
&\quad w_{ij} \text{ in } \{\ \overline{w} \,|X \to \overline{w}X \in P\}\}.
\end{aligned}$$

After these observation, we now consider the set of all linear strings in $\delta^*(A')$. That is, from the manner of constructing R of H, we calculate

$$\begin{aligned}
&\delta^*(A') \cap V'^* = \\
&A_\ell \cup \{z | \exists x \in A[x \Longrightarrow^* z] \text{ in } G\} \\
&\cup \{\mathrm{B}@u \mid uX \to \overline{w}X \in P\} \\
&\cup \{\mathrm{B}@\overline{b} \mid \overline{b}X \to Xb \in P\} \\
&\cup \{w_{i1} \cdots w_{ip}\mathrm{B}' \mid p \geq 1,\ w_{ij} \in \{\ \overline{w} \,|X \to \overline{w}X \in P\}\ \} \\
&\cup \{\ @'@\ \}.
\end{aligned}$$

Hence, it holds that

$$L_\ell(H) = \delta^*(A') \cap \Sigma^* = L. \square$$

Corollary 5 *A language is recursively enumerable iff it is generated by a CH system whose rule set comprises types of* `REP`, `ROT` *and* `CL`.

Note that in fact only `REP` and `ROT` realized by using only the *suffix mode* are playing an essential role in the computation.

By $\mathcal{CH}_\ell$ denote the family of all languages $L_\ell(H)$ generated by CH systems H. Then, Lemma 4 and Turing-Church thesis lead to the following.

Theorem 6 $\mathcal{RE} = \mathcal{CH}_\ell$.

4 Universal Circular System

From the practical point of view of "programmable DNA computers", it is obviously important to have a *universal CH system* $H_u = (V_u, \Sigma, A_u, R_u)$ in the following sense : given an alphabet Σ and for every CH system $H = (V, \Sigma, A, R)$, there effectively exists a finite set A_H such that $L_\ell(H) = L_\ell(H'_u)$, where $H'_u = (V_u, \Sigma, A_u \cup A_H, R_u)$, A_u is a finite set of $V_u^* \cup V_u\hat{}$ and R_u is a finite set of $(V_u^*\#V_u^*\$V_u^*\#V_u^*) \times M$.

We are going to show that there effectively exists a universal CH system H_u for the class of CH systems. The proof is as a whole similar to that of the existence of a universal H system in [6], however we will take more direct path to the goal by way of Post Systems rather than type-0 grammars or Turing machines.

Theorem 7 *Given any alphabet Σ, there effectively exists a universal CH system H'_u for the class of CH systems over Σ.*

Proof. Starting with any CH system H with terminal alphabet Σ, from Lemma 1 and Theorem 6, there effectively exists a PN system $G = (V, \Sigma, P, A)$ such that $L(G) = L_\ell(H)$.

For the class of PN systems with the terminal alphabet Σ, we first consider a kind of a universal Post system for the class of PN systems, that is, a Post system $G_u = (V_{v_u}, V_{c_u}, P_u, -)$ such that for any PN system $G = (V, \Sigma, P, A)$ there is a string $w(G)$ in $V_{c_u}^*$ such that $L(G) = L(G'_u)$, where $G'_u = (V_{v_u}, V_{c_u}, P_u, \{w(G)\})$ and without loss of generality we may assume that G has a unique axiom $A = \{S\}$. In fact, this string $w(G)$ is constructed as follows:

$$w(G) = \mathtt{d}h(u_1)\mathtt{c}h(w_1)\mathtt{d}\cdots\mathtt{d}h(u_k)\mathtt{c}h(w_k)\mathtt{d}\%h(S)$$
$$\text{if } P = \{u_1X \to Xw_1, ..., u_kX \to Xw_k\},$$

where h is a mapping defined by $h(Z_i) = \mathtt{c}_1\mathtt{c}_2^i\mathtt{c}_1$ for each $Z_i \in V - \Sigma = \{Z_1, ..., Z_n\}$ and $h(a) = a (a \in \Sigma)$. Actually, construct G_u as follows:

$V_{v_u} = \{X_1, X_2, X_3, X, Y, Z\}$,
$V_{c_u} = \{\mathtt{c}_1, \mathtt{c}_2, \mathtt{c}, \mathtt{d}, \%\} \cup \Sigma$, and
$P_u = \{X_1\mathtt{d}Y\mathtt{c}Z\mathtt{d}X_2\%YX_3 \to X_1\mathtt{d}Y\mathtt{c}Z\mathtt{d}X_2\%X_3Z\}$
$\cup\{Y\%X \to X\}$.

Thus, we have obtained a Post system $G_u = (V_{v_u}, V_{c_u}, P_u, -)$ with the terminal alphabet Σ such that $L(G) = L(G'_u)$, where $G'_u = (V_{v_u}, V_{c_u}, P_u, \{w(G)\})$. Then, from the main theorem in [16], one can convert G_u into a Post Normal system $\tilde{G}_u = (\tilde{V}_u, \Sigma, \tilde{P}_u, -)$ such that $L(G'_u) = L(\tilde{G}'_u)$, where $\tilde{G}'_u = (\tilde{V}_u, \Sigma, \tilde{P}_u, \{w'(G)\})$ and $w'(G)$ is a modification of $w(G)$. Further, using Lemma 2, one can also effectively find a GRP system $\tilde{G}''_u$ such that $L(\tilde{G}'_u) = L(\tilde{G}''_u)$.

Now, applying the proof procedure of Lemma 4 to $\tilde{G}''_u$, we eventually obtain a CH system $H'_u = (V'_u, \Sigma, A'_u \cup A_H, R'_u)$ satisfying that

$$L_\ell(H'_u) = L(\tilde{G}''_u) = L(\tilde{G}'_u) = L(G'_u) = L(G) = L_\ell(H),$$

where A_H is obtained from $w(G)$ which depends on H. Thus, $H_u = (V'_u, \Sigma, A'_u, R'_u)$ is a universal CH system. □

5 A Normal Form for CH Systems

We consider in this section the following problem : given a CH system H, find an equivalent CH system H' which is as simple as possible under a certain criterion.

To this end, following [11] we consider some special types of CH systems. Given a positive integer $k \geq 1$, a CH system $H = (V, \Sigma, A, R)$ is said to be *k-limited* iff for every $u_1\#u_2\$u_3\#u_4 \in R$, it holds that $|u_i| \leq k$ (for $i = 1, 2, 3, 4$).

Now we present the following result whose proof is omitted here due to the space limitation. (See [21] for the proof.)

Theorem 8 *Given any CH system H, one can effectively find an equivalent 3-limited CH system H'.*

Thus, this result provides us with a kind of normal form theorem for the class of CH systems. As far as we know, this would be the simplest splicing systems with the universal computability.

It should be noted that this 3-limitedness property of CH systems has a very significant meaning in the biological sense, in particular from the viewpoint of designing DNA computer as briefly mentioned below.

6 Discussions

Since Tom Head's pioneering paper [8] on his splicing systems for modeling DNA recombination behaviors, Paun and his group have made a series of intensive studies on an extended version of splicing systems where most of the problems considered can be formulated in the following: Starting with a language A from a family $\mathcal{F}_1$ and a language R from a family $\mathcal{F}_2$, investigate properties of the family of languages $\sigma^*(A)$ or $\sigma^*(A) \cap \Sigma^*$ thus obtained, where σ is a splicing operation realized by a rule of the form $u_1\#u_2\$u_3\#u_4$ in R with or without multiplicity, where $\mathcal{F}_1$ and $\mathcal{F}_2$ range within Chomsky's hierarchy. ([11, 12, 13]). In [6] Freund et al show the existence of universal splicing systems for several types of extended splicing systems, suggesting the possible design for programmable DNA computer based on the splicing model.

In contrast, a relatively few works on circular splicing systems and their languages are known. Tom Head, the initiator of this research area of formal splicing systems, reviews previous work on splicing systems and addresses the significance of studying this new type of "mixed splicing" systems as well as that of multiplicity notions in the standard splicing model ([9]). Siromoney et al propose several splicing operations involving circular strings as well and show some non-regualrity circular example generated by a mixed splicing system with finite axioms and rules ([19]). A paper by Pixton ([15]) provides us with intensive studies on the regularity properties of linear, circular and mixed splicing languages. All these work are, however, concerned with only the original (non-extended) model of splicing systems, while we are dealing with extended models of mixed splicing systems in this article. In a separate paper ([5]), we have discussed some new developments on the computational power of splicing systems in which, by establishing interesting relationships between splicing systems and Post systems, new characterizations of families of regular and recursively enumerable languages are presented in the framework of extended splicing systems.

A molecular biological observation reports that *lambda phage* DNA can easily change its form from linear to circular and vice versa. When *lambda phage* DNA is transposed in a bacterial DNA like *E.Coli* DNA, it forms a circular then is incorporated into a host DNA by splicing (crossing over), resulting that the original genes A and B in *lambda phage* can be reordered as B and A in the recombinant DNA.

These may support the appropriateness of the notion of circular H systems and give some justification of our splicing models. It should be noted that the circularity plays a critical role in achieving the universal computability of Turing machine in our model.

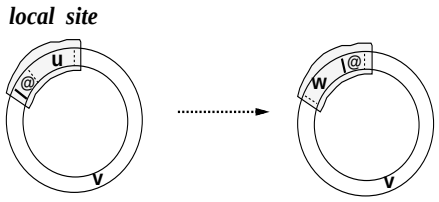

Figure 2. Basic Operation for DNA Computing

In more detailed discussion, because of the circularity of a DNA sequence, performing a rewriting rule $uX \rightarrow Xw$ with $w \neq \lambda$ (in Post Normal system) by circular splicing can easily be carried out so that only a *local splicing site of fixed length* should be preserved during the splicing process (see Figure 2). In fact, Theorem 8 suggests that it suffices to keep the length 6 window for each splicing rule. This local feature is extremely important when we design an architecture for programmable DNA computers and may provide feasible procedures to realize it. This observation also suggests the high feasibility of a universal CH system.

Finally, as for the DNA implementation of *rotation operation*, one can find some detailed discussion in [2], while [17] discusses the DNA and restriction enzyme implementation of Turing machines.

Acknowledgements

This work was supported in part by Grant No. JSPS-RFTF 96I00101 from Research-for-the-Future Program of the Japan Society for the Promotion of Science.

References

[1] L. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, 1994.

[2] M. Arita and M. Hagiya. Joining and rotating data with molecules. In *Proc. of IEEE Intern. Conf. Evol. Comput.*(ICEC'97), (this volume) 1997.

[3] D. Beaver. A universal molecular computer. In *DNA Based Computers* (Lipton and Baum, eds.), DIMACS series, Vol.27, *Proc. of a DIMAC Workshop*, AMS, pages 29–36, 1996.

[4] D. Boneh, C. Dunworth, and R. Lipton. Breaking DES using a molecular computer. In *DNA Based Computers* (Lipton and Baum, eds.), DIMACS series, Vol.27, *Proc. of a DIMAC Workshop*, AMS, pages 37–65, 1996.

[5] C. Ferretti and S. Kobayashi. DNA splicing systems and Post systems. In *Proc. of Pacific Symposium on Biocomputing*, Hawaii, World Scientific Publisher, pp.288-299. 1996.

[6] R. Freund, L. Kari, and Gh. Paun. DNA computing based on splicing : The existence of universal computers. *J. ACM*, to appear.

[7] M.A. Harrison. *Introduction to Formal Language Theory.* Addison-Wesley, Reading, MA, 1978.

[8] T. Head. Formal language theory and DNA : An analysis of the generative capacity of specific recombinant behaviors. *Bulletin of Mathematical Biology*, 49:737–759, 1987.

[9] T. Head. Splicing schemes and DNA. In *Lindenmayer Systems* (G.Rozenberg and A.Salomma, eds.), pages 371–383. Springer-Verlag, 1992.

[10] R. Lipton. Using DNA to solve NP-complete problems. *Science*, 268: 542–545, 1995.

[11] Gh. Paun. On the splicing operation. *Discrete Applied Mathematics*, 70: 57–79, 1996.

[12] Gh. Paun. On the power of the splicing operation. *Intern. Journal Comput. Math.*, to appear, 1996.

[13] Gh. Paun. Regular extended H systems are computationally universal. *J. Automata, Languages and Combinatorics*, 1: 27–36, 1996.

[14] Gh. Paun, G. Rozenberg, and A. Salomaa. Computing by splicing. *Theoretical Computer Science*, to appear, 1996.

[15] D. Pixton. Regularity of splicing languages. *Discrete Applied Mathematics*, 69: 101–124, 1996.

[16] E. L. Post. Formal reductions of the general combinatorial decision problem. *American Journal of Mathematics*, 65:197–215, 1943.

[17] P.W.K. Rothemund. A DNA and restriction enzyme implementation of Turing machines. In *DNA Based Computers* (Lipton and Baum, eds.), DIMACS series, Vol.27, *Proc. of a DIMAC Workshop*, AMS, pages 75–119, 1996.

[18] A. Salomaa. *Computation and Automata.* Cambridge University Press, 1985.

[19] R. Siromoney, K.G. Subramanian, and V.R. Dare. Circular DNA and splicing systems. In *Proc. of Intern. Conference on Parallel Image Analysis,* Lecture Notes in Computer Science 654, Springer-Verlag, pages 260–273, 1992.

[20] T. Yokomori and S. Kobayashi. DNA evolutionary linguistics and RNA structure modeling: A computational approach. In *Proc. of IEEE Symp. on Intelligence in Neural and Biological Systems*, pages 38–45, 1995.

[21] T. Yokomori, S. Kobayashi and C. Ferretti. DNA On the power of circular splicing systems and DNA computability, Technical report, Dept. of Comput. Sci. and Inform. Math., Univ. of Electro-Commun., CSIM 95-01, 1995.