A Note on Self Cross-Over of Circular Words and Arrays

K.G. SUBRAMANIAN, D.G. THOMAS, M.H. BEGUM, P.H. CHANDRA

Department of Mathematics, Madras Christian College
Tambaram, Chennai (formerly Madras) 600 059, India
E-mail: kgsmtc@kmronline.com

1 Introduction

Head, in his pioneering paper [1], introduced a new class of generative systems, called splicing systems, which have been conceived as a generative formalism with the aim of analyzing the generative capacity of the recombinant behaviour of DNA molecules, in terms of formal languages. Păun et. al. [7], [8] extended the definition of Head and defined extended H systems which are computationally universal. The splicing systems make use of a new operation, called splicing on strings of symbols. Since then theoretical investigations of splicing on strings have been extensively done by different researchers. A comprehensive survey of the work in this area is given in [2], [8] in a general setting.

Recently, Dassow and Mitrana [3] investigated a very simple and natural restriction on the splicing operation, namely the cross-over rule applicable only to identical strings, in trying to capture certain features of the recombination of genes in a chromosome.

DNA exist not only in linear form but also in circular form and hence an extension to circular strings of the splicing operation has been introduced and investigated [2], [8]. Moreover, the splicing operation applied to arrays and graphs has been considered in [5], [6]. It is natural to examine the effect of self cross-over on circular words, arrays and graphs. In this paper we introduce and study self cross-over circular systems and self cross-over array systems. Self cross-over on graphs is examined in [4].

2 Self Cross-Over Circular Languages

Let V be a finite alphabet. $V^*$ is the set of all (linear) strings over V including the empty string $\lambda$. For details of formal language theory we refer to [8].

We recall the notion of a circular word [2], [8]. A circular word (or string) over an alphabet $V$ is a sequence $x_1x_2 \ldots x_n$ of elements of $V$ with the understanding that $x_1$ follows $x_n$. For example, the circular strings $abaab, baaba, aabab, ababa, babaa$ are all regarded as the same circular string. In fact, these five strings can be considered as linearized forms of a single circular string $x$ of symbols $a, b, a, a, b$, in this order, with the first symbol $a$ following the last symbol $b$. We write $x = \sim abaab$. In other words, by introducing an equivalence relation $\sim$ on strings in $V^*$ by requiring $xy \sim yx$, for all strings $x, y$ in $V^*$, we define the circular word $\sim w$ to be the equivalence class of the string $w$. The set of all circular strings over $V$ is denoted by $V^c$. $Cir(L)$ is the set of all circular
strings corresponding to elements of $L$ and is called the circularization of $L$. If $C$ is a circular language in $V^\ast$, any language $L$ for which $\text{Cir}(L) = C$, is called a linearization of $C$. The set $\text{Lin}(C)$ of all strings in $V^\ast$ corresponding to elements of $C$ is called the full linearization of $C$.

**Example 2.1.** $C = \{a^{2m}b \mid m \geq 0\}$ is a circular language over $V = \{a, b\}$. $L_1 = (a^2)^*b$ is a linearization of $C$. $L_2 = \{a^m ba^n \mid m \geq 0\}$ is another linearization. The full linearization is $\text{Lin}(C) = (a^2)^* \{b, aba\} (a^2)^*$.

Regularity of a circular language can be defined as follows [2], [8]: A circular language $C \subseteq V^\ast$ is regular if there exists a regular language $L \subseteq V^\ast$ such that $L$ is a linearization of $C$. In other words, a circular language $C$ is regular if and only if $C$ has a regular linearization. The circular language $C$ given in Example 2.1 is regular.

We now introduce self cross-over circular systems.

**Definition 2.1.** A self cross-over circular system (SCOC) is a triple $S = (V, A, R)$, where $V$ is an alphabet, $A$ is a finite subset of $V^\ast$, and $R$ is a finite commutative relation, $R \subseteq (V^\ast \times V^\ast)^2$. (Sometimes, for $(\alpha, \beta) R(\gamma, \delta)$, we write $(\alpha, \beta; \gamma, \delta) \in R$.)

Given a self cross-over circular system $(V, A, R)$, we define, for $x^\ast, y^\ast \in V^\ast$, $x \vdash^* y$ if and only if

1. $x = x_1 \alpha \beta x_2 = x_3 \gamma \delta x_4$,
2. $y = x_1 \alpha \delta x_4$,
3. $(\alpha, \beta) R(\gamma, \delta)$.

As usual, $\vdash^*$ is the reflexive, transitive closure of the relation $\vdash$. The circular language generated by a self cross-over circular system, called a self cross-over circular language, is given by

$$L(S) = \{x^\ast \in V^\ast \mid x \vdash^* x, x \in A\}.$$ 

**Example 2.2** Let $V = \{a, b\}$, $A = \{\ast b a b\}$, $R = \{(a, b; b, a)\}$, $S = (V, A, R)$

Then $L(S) = \{b^2 a, \ast b^4 a^2, \ast b^8 a^4, \ldots \} = \{b^{2^n} a^{2^{n-1}} \mid n \geq 1\}$.

**Theorem 2.1** Every self cross-over circular language $L$ over $\{a\}$ is of the form $F$ or $F \cup \bigcup_k \{\ast a^{k \cdot 2^n} \mid n \geq 0, \text{ for some } \ast a^k \in F\}$, where $F$ is a finite subset of $\{a\}^\ast$.

**Proof.** Let $S = (\{a\}, A, R)$ be a self cross-over circular system. If $R$ is empty, then $F = A$. Otherwise, since $L(S) = \bigcup_{x \in A} L(S_x)$, where $S_x = (\{a\}, \ast x, R)$, it suffices to show that we have

$$L(S_x) = \{\ast a^{k \cdot 2^n} \mid n \geq 0, \text{ for some positive integer } k \text{ with } |x| = k,
\text{ } k \geq |\alpha| + |\beta| \text{ and } k \geq |\gamma| + |\delta|, \text{ for some } (\alpha, \beta; \gamma, \delta) \in R\}.$$ 

Let $(\alpha, \beta; \gamma, \delta)$ be a rule in $R$. Let $x \in A$ such that $x = \ast a^k$ for some $k \geq |\alpha| + |\beta|$ and $k \geq |\gamma| + |\delta|$. We have:

$$x = \ast a^{m \cdot 2^n} a^{x|\alpha| + |\beta|} a^n = \ast a^m \alpha \beta a^n, \text{ where } m + |\alpha| + |\beta| + n = k, \text{ and}$$

$$x = \ast a^{p \cdot 2^n} a^{x|\alpha| + |\beta|} a^n = \ast a^p \gamma \delta a^q, \text{ where } p + |\gamma| + |\delta| + q = k.$$
Then \( a^n a^m a^n a^p \beta a^n = a^{n+|a|+|\beta|+|a|+|\beta|+n} = a^{k_2}. \)

Likewise it can be seen that \( a^n a^{k_2} \) for every \( n \geq 0 \). Consequently, \( L(S_x) = \{ a^{k_2} | n \geq 0, \text{ for some positive integer } k \} \), and thus \( L(S) = F \cup \bigcup_{k} \{ a^{k_2} | n \geq 0 \}. \)

**Notation:** \( \mathcal{L}(SCO) \) [3] is the family of self cross-overs languages of linear words and \( \mathcal{L}(SCOC) \) is the family of self cross-over circular languages.

**Proposition 2.1** \( \mathcal{L}(SCOC) \neq Cir(\mathcal{L}(SCO)) \) and \( \mathcal{L}(SCO) \neq Lin(\mathcal{L}(SCOC)) \).

In fact, when we consider the one letter case, if \( S = \{ \{ a \} \}, \{ \{ a \} \}, \{ \{ \lambda, \lambda, \lambda \} \} \), then \( L(S) = \{ a^{2n} | n \geq 0 \} \) and \( Lin(L(S)) = \{ a^{2n} | n \geq 0 \} \notin \mathcal{L}(SCO) \).

If \( S' = \{ \{ a \} \}, \{ \{ a \} \}, \{ \{ \lambda, \lambda, \lambda \} \} \), then \( L(S') = a^* \) and \( Cir(L(S')) = a^* \notin \mathcal{L}(SCOC) \).

**Proposition 2.2** The family \( \mathcal{L}(SCOC) \) is not closed under union.

The languages \( L_1 = \{ b^{2n} a^{2n-1} | n \geq 1 \} \) and \( L_2 = \{ a^{2n} | n \geq 0 \} \) are self cross-over circular languages, but it is clear that their union is not a self cross-over circular language.

**Proposition 2.3** The family \( \mathcal{L}(SCOC) \) is not closed under morphisms.

This assertion follows by considering \( S = (V, A, R) \), where \( V = \{ a, b \}, A = \{ \text{bab} \} \) and \( R = \{ \{ \lambda, \lambda, \lambda \} \} \), so that \( L = L(S) = \{ \text{bab}^n | n \geq 1 \} \cup B \), where \( B \) is some subset of \( V^* \).

By considering the morphism \( h : \{ a, b \}^* \rightarrow \{ a \}^* \) defined by \( h(a) = a, \ h(b) = \lambda \) we have \( h(L) = \{ a^n | n \geq 1 \} \) which is not in \( \mathcal{L}(SCOC) \).

**Proposition 2.4** The family \( \mathcal{L}(SCOC) \) is not closed under complement.

This follows by considering \( S = (V, A, R) \) where \( V = \{ a \}, A = \{ \text{a} \} \), and \( R = \{ \{ \lambda, \lambda, \lambda \} \} \) so that \( L = L(S) = \{ \text{a}^{2n} | n \geq 0 \} \).

The complement of \( L \) is \( V^* - L = \bigcup_{k, \text{odd}, k \geq 3} \{ a^{k_2} | n \geq 0 \} \).

The union being over infinite odd \( k \), \( V^* - L \notin \mathcal{L}(SCOC) \) by Theorem 2.1.

**Proposition 2.5** The family \( \mathcal{L}(SCOC) \) is not closed under intersection with regular circular languages.

The language \( L \) considered in Proposition 2.3 is such that the regular circular language \( \{ \text{bab}^n | n \geq 1 \} = L \cap \{ \text{bab}^n | n \geq 1 \} \) is not in \( \mathcal{L}(SCOC) \).

### 3 Array Splicing Systems

The splicing operation on arrays using domino splicing rules has been introduced in [10].

We recall these notions for completeness.

**Definition 3.1** [9] Let \( \Sigma \) be a finite alphabet. An image or a picture over \( \Sigma \) is a rectangular array of elements of \( \Sigma \). The set of all images is denoted by \( \Sigma^{**} \). An image or a picture of size \( m \times n \) is an array of the form
\[ a_{11} \quad a_{12} \quad a_{13} \quad \ldots \quad a_{1n} \\
 a_{21} \quad a_{22} \quad a_{23} \quad \ldots \quad a_{2n} \\
 \vdots \quad \vdots \quad \vdots \quad \quad \ldots \quad \vdots \\
 a_{m1} \quad a_{m2} \quad a_{m3} \quad \ldots \quad a_{mn} \]

or, in short, \([a_{ij}]_{m \times n}\). A picture language or a two-dimensional language over \(\Sigma\) is a subset of \(\Sigma^{**}\).

**Definition 3.2** [10] Let \(V\) be an alphabet and \#, \$ two special symbols, not in \(V\). A domino over \(V\) is of the form \[ \frac{a}{b} \] or \[ \frac{a}{b} \], \(a, b \in V\).

A domino column splicing rule over \(V\) is of the form \(r = \alpha_1 \# \alpha_2 \$ \alpha_3 \# \alpha_4\) where \(\alpha_i = \frac{a}{b}\) for some \(a, b \in V\), or \(\alpha_i = \frac{\lambda}{\lambda}\) where \(\lambda\) is the empty word.

A domino row splicing rule over \(V\) is of the form \(r = \beta_1 \# \beta_2 \$ \beta_3 \# \beta_4\), where each \(\beta_i = \frac{c}{d}\) for some \(c, d \in V\), or \(\beta_i = \frac{\lambda}{\lambda}\).

Given two arrays \(X\) and \(Y\) of sizes \(m \times p\) and \(m \times q\), respectively,

\[
X =
\begin{array}{cccc}
 a_{11} & \ldots & a_{1j} & \ldots & a_{1p} \\
 \vdots & \ddots & \vdots & \ddots & \vdots \\
 a_{m1} & \ldots & a_{mj} & \ldots & a_{mp}
\end{array}
\]

\[
Y =
\begin{array}{cccc}
 b_{11} & \ldots & b_{1k} & \ldots & b_{1q} \\
 \vdots & \ddots & \vdots & \ddots & \vdots \\
 b_{m1} & \ldots & b_{mk} & \ldots & b_{mq}
\end{array}
\]

we write \((X,Y) \Phi Z\) if there exist the column splicing rules \(r_1, r_2, r_3, \ldots, r_{m-1}\) (not necessarily all different) such that

\[
r_i = \frac{a_{ij}}{a_{i+1,j}} \quad \# \quad \frac{a_{i,j+1}}{a_{i+1,j+1}} \quad \$ \quad \frac{b_{ik}}{b_{i+1,k}} \quad \# \quad \frac{b_{i,k+1}}{b_{i+1,k+1}}
\]

for all \(1 \leq i \leq m - 1\), for some \(1 \leq j \leq p, 1 \leq k \leq q\), and

\[
Z =
\begin{array}{cccc}
 a_{11} & \ldots & a_{1j} & b_{1,k+1} & \ldots & b_{1q} \\
 \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
 a_{m1} & \ldots & a_{mj} & b_{m,k+1} & \ldots & b_{mq}
\end{array}
\]

In particular, if any of the symbols \(a_{ij}\) is \(\lambda\), then for all \(1 \leq i \leq m\) we have \(a_{ij} = \lambda\). Likewise, for \(a_{i,j+1}, b_{ik}, b_{i,k+1}, 1 \leq i \leq m\). We now say that \(Z\) is obtained from \(X\) and \(Y\) by domino column splicing in parallel.

262
We can similarly define row splicing operation of two arrays $U$ and $V$ of sizes $p \times n$ and $q \times n$ using row splicing rules to yield an array $W$.

\[
U = \begin{bmatrix}
  a_{11} & a_{12} & \cdots & a_{1n} \\
  \vdots & \vdots & \ddots & \vdots \\
  a_{i1} & a_{i2} & \cdots & a_{in} \\
  a_{i+1,1} & a_{i+1,2} & \cdots & a_{i+1,n} \\
  \vdots & \vdots & \ddots & \vdots \\
  a_{p1} & a_{p2} & \cdots & a_{pn}
\end{bmatrix}
\]

\[
V = \begin{bmatrix}
  b_{11} & b_{12} & \cdots & b_{1n} \\
  \vdots & \vdots & \ddots & \vdots \\
  b_{k1} & b_{k2} & \cdots & b_{kn} \\
  b_{k+1,1} & b_{k+1,2} & \cdots & b_{k+1,n} \\
  \vdots & \vdots & \ddots & \vdots \\
  b_{q1} & b_{q2} & \cdots & b_{qn}
\end{bmatrix}
\]

We write $(U, V) \xrightarrow{\theta} W$ if there exist some row splicing rules $r_1, r_2, r_3, \ldots, r_{n-1}$ such that

\[
\begin{align*}
  r_i &= \left[ a_{ij} \mid a_{i,j+1} \right] \# \left[ a_{i+1,j} \mid a_{i+1,j+1} \right] \$ \left[ b_{kj} \mid b_{k,j+1} \right] \# \left[ b_{k+1,j} \mid b_{k+1,j+1} \right]
\end{align*}
\]

for all $1 \leq j \leq n-1$ and for some $1 \leq i \leq p$ and $1 \leq k \leq q$, and

\[
W = \begin{bmatrix}
  a_{11} & a_{12} & \cdots & a_{1n} \\
  \vdots & \vdots & \ddots & \vdots \\
  a_{i1} & a_{i2} & \cdots & a_{in} \\
  a_{i+1,1} & a_{i+1,2} & \cdots & a_{i+1,n} \\
  \vdots & \vdots & \ddots & \vdots \\
  b_{k1} & b_{k2} & \cdots & b_{kn} \\
  b_{k+1,1} & b_{k+1,2} & \cdots & b_{k+1,n} \\
  \vdots & \vdots & \ddots & \vdots \\
  b_{q1} & b_{q2} & \cdots & b_{qn}
\end{bmatrix}
\]

We say that $W$ is obtained from $U$ and $V$ by domino row splicing in parallel.

**Definition 3.2** [10] An H array splicing scheme is a triple $\Gamma = (V, R_c, R_r)$, where $V$ is an alphabet, $R_c$ is a finite set of domino column splicing rules, and $R_r$ is a finite set of domino row splicing rules.

For a given H array splicing scheme $\Gamma = (V, R_c, R_r)$ and a language $L \subseteq V^\ast$, we define

\[
\Gamma(L) = \{ Z \in V^\ast \mid (X,Y) \xrightarrow{\theta} Z \text{ or } (X,Y) \xrightarrow{\theta} Z \text{ for some } X,Y \in L, \ p_i \in R_c \text{ and } q_j \in R_r, 1 \leq i \leq m-1, 1 \leq j \leq n-1 \}.
\]

Iteratively, we define

\[
\begin{align*}
  \Gamma^0(L) &= L, \\
  \Gamma^{i+1}(L) &= \Gamma(L) \cup \Gamma(\Gamma^i(L)), i \geq 0, \\
  \Gamma^*(L) &= \bigcup_{i \geq 0} \Gamma^i(L).
\end{align*}
\]

An H array splicing system is defined by $S = (\Gamma, I)$, where $\Gamma = (V, R_c, R_r)$ and $I$ is a finite subset of $V^\ast$. The language of $S$ is defined by $L(S) = \Gamma^*(I)$ and we call it a splicing array language.
We illustrate this definition with an example.

**Example 3.1** Let \( V = \{a, b\} \) and

\[
I = \begin{pmatrix}
a & b \\
b & a
\end{pmatrix},
\]

\[
R_e = \{ p_1 : \begin{pmatrix}
a & b \\
b & a
\end{pmatrix} \},
\]

\[
p_2 : \begin{pmatrix}
\lambda & \lambda \\
\lambda & a
\end{pmatrix} \}
\]

\[
R_e = \{ q_1 : \begin{pmatrix}
a & b \\
b & a
\end{pmatrix} \},
\]

\[
q_2 : \begin{pmatrix}
\lambda & \lambda \\
\lambda & \lambda
\end{pmatrix} \}
\]

By a column splicing in parallel using \( p_2 \), the arrays \( \begin{pmatrix} a & b \\ b & a \end{pmatrix}, \begin{pmatrix} a & b \\ b & a \end{pmatrix} \) yield

\[
\begin{pmatrix}
a & b \\
b & a
\end{pmatrix} \begin{pmatrix}
\lambda & \lambda \\
\lambda & a
\end{pmatrix} \Hrightarrow \begin{pmatrix}
a & b & a \\
b & a & a
\end{pmatrix}.
\]

We have shown the empty column \( \lambda \) to indicate the place where splicing is done.

Likewise, a row splicing in parallel using \( q_1, q_2 \), gives

\[
\begin{pmatrix}
\lambda & \lambda & \lambda \\
\lambda & \lambda & \lambda \\
\lambda & \lambda & \lambda
\end{pmatrix} \begin{pmatrix}
a & b & a & b \\
b & a & b & a \\
b & a & b & a
\end{pmatrix} \Hrightarrow \begin{pmatrix}
a & b & a & b \\
b & a & b & a \\
a & b & a & b \\
b & a & b & a
\end{pmatrix}.
\]

\( L \) is the language consisting of all “chessboards” with even side-length (Figure 1).

\[
\begin{array}{cccc}
a & b & a & b \\
b & a & b & a
\end{array}
\begin{array}{cccc}
a & b & a & b \\
b & a & b & a
\end{array}
\begin{array}{cccc}
a & b & a & b \\
b & a & b & a
\end{array}
\begin{array}{cccc}
a & b & a & b \\
b & a & b & a
\end{array}
\]

Figure 1

The picture language generating power of the H array splicing systems has been examined in [10].
4 Self Cross-Over Array Languages

We now introduce self cross-over array systems.

**Definition 4.1** A self cross-over array system is defined by $S = (\Gamma, I)$, where $\Gamma = (V, R_c, R_r)$ is an H array splicing scheme and $I$ is a finite subset of $V^*$. A set of domino splicing rules is applied to two identical copies of the same array. A self cross-over array language is defined as in the case of linear strings.

**Example 4.1** The self cross-over array system $S = (V, R_c, R_r, I)$, where $V = \{0,1\}$, and

$$R_c = \{ p_1 : \begin{array}{l}
0 & \lambda & 1 & \lambda \\
1 & 1 & 0 & 1 \\
end{array} \}$

$$p_2 : \begin{array}{l}
0 & \lambda & 1 & 0 \\
0 & 1 & 0 & 0 \\
end{array} \},$$

$$R_r = \{ q_1 : \begin{array}{l}
1 & 0 & 1 & \lambda \\
0 & 1 & 1 & \lambda \\
end{array} \}$

$$q_2 : \begin{array}{l}
0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 \\
end{array} \},$$

$$I = \begin{Bmatrix}
1 & 0 \\
1 & 1 \\
\end{Bmatrix},$$

generates the picture language $M$ consisting of all $2^n \times 2^n$, $n \geq 0$, arrays of 0’s and 1’s, describing L arrangements of 1’s (Figure 2).

$$\begin{array}{cccccccccccc}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\end{array}$$

**Figure 2**

We note that this picture language cannot be generated by any regular or even context-free two-dimensional matrix grammar [9].

**Proposition 4.1** The language $L = \{(a^p)_m (b^q)_m (a^r)_m (b^s)_m \mid m \geq 1, p, q, r, s \geq 0\}$, where $(x^n)_m$ is an array consisting of $m$ rows with each row containing $x^n$ for a given $n$, cannot be generated by any self cross-over array system.

This is a consequence of Lemma 3.1 of [3].
**Remark:** As a corollary of Proposition 4.1, the family of self cross-over array languages is incomparable with the families of regular and context-free two-dimensional matrix languages [9].

We conclude this section by stating some closure properties of the family of self cross-over array languages [9].

**Proposition 4.2** The family of self cross-over array languages (i) is not closed under union and column (row) catenation, but (ii) is closed under reflections on the base and right leg and rotations by 90°, 180° and 270°.

The nonclosure result can be seen by constructing picture languages analogous to the linear string case [3]. The closure under the geometric operations in (ii) can be seen as in the case of splicing array languages [10].

**Conclusion**

In this note a beginning is made for a theoretical study of the notion of self cross-over of circular words and arrays. It remains to be seen whether other properties obtained in the linear string case [3] carry over here.

**References**


