Controlled H Systems of Small Radius

Andrei PĂUN

Faculty of Mathematics, University of Bucharest
Str. Academiei 14, 70109 Bucureşti, Romania

Abstract. Several characterizations of recursively enumerable languages are given, using H systems with permitting contexts having splicing rules of small radius. Representations of context-free languages are also obtained in certain particular cases. These results improve previous related results which were recently published. Some open problems are also pointed out.

1. Extended H systems

We consider here extended H systems in the sense of [11], as generative mechanisms based on the *splicing operation* introduced in [5]. In DNA computing area, such systems having splicing rules of small dimensions are of interest; see, e.g., [6], [8]. The study of *small H systems* (from the point of view of rules size) is the purpose of this paper.

A *splicing rule* (over an alphabet V) is a string $r = u_1\#u_2\$u_3\#u_4$, where $u_1, u_2, u_3, u_4 \in V^*$ and $\#, \$$ are two special symbols not in V. (V^* is the free monoid generated by the alphabet V under the operation of concatenation; the empty string is denoted by λ; the length of $x \in V^*$ is denoted by $|x|$.)

For $x, y, w, z \in V^*$ and r as above we write

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

We say that we splice x, y at the *sites* $u_1u_2,\ u_3u_4$. These sites encode the patterns recognized by restrictions enzymes able to cut the DNA sequences between u_1, u_2, respectively between u_3, u_4. The strings x, y are called the *terms* of the splicing.

The *radius* of a splicing rule $u_1\#u_2\$u_3\#u_4$ is the length of the longest string u_1, u_2, u_3, u_4.

An *extended H system* ([11]) is a quadruple $\gamma = (V, T, A, R)$, where V is an alphabet, $T \subseteq V$, A is a finite subset of V^*, and $R \subseteq V^*\#V^*\$V^*\#V^*$, for $\#, \$$ being special symbols not in V. (V is the total alphabet, T is the terminal alphabet, A is the set of axioms, R is the set of splicing rules.) The pair $\sigma = (V, R)$ is sometimes called the underlying *H scheme* associated to γ.

For any $L \subseteq V^*$ and $\gamma = (V, T, A, R)$ we define

$$\sigma(L) = \{w \mid (x, y) \vdash_r (w, z) \text{ or } (x, y) \vdash_r (z, w), \text{ for } x, y \in L, r \in R\},$$
$$\sigma^*(L) = \bigcup_{i \geq 0} \sigma^i(L), \text{ for}$$
$$\sigma^0(L) = L,$$
$$\sigma^{i+1}(L) = \sigma^i(L) \cup \sigma(\sigma^i(L)), \ i \geq 0.$$

Then, the language generated by γ is

$$L(\gamma) = \sigma^*(A) \cap T^*.$$

(We iteratively splice strings, starting from axioms, and we keep in the generated language only strings consisting of terminal symbols.)

Convention. Two languages L_1, L_2 are considered equal if they differ by at most the empty string, that is $L_1 = L_2$ iff $L_1 - \{\lambda\} = L_2 - \{\lambda\}$.

We denote by *CF, RE* the families of context-free and of recursively enumerable languages, respectively. We refer to [13] for basic elements of formal language theory.

2. Extended H systems with permitting contexts

The extended H systems with finitely many splicing rules generate only regular languages ([3], [12]); when a regular set of splicing rules is used we can characterize the family of recursively enumerable languages ([7]). However, a computing device with infinitely many rules, even constituting a regular language, is not of practical interest. In view of the result in [3], [12], in order to increase the power of finite extended H systems beyond the power of finite automata we have to impose a control on the splicing operation. Several such control mechanisms were considered in [2], [4], [9], [10], etc. We discuss here only one of them.

An *extended H system with permitting contexts* is a quadruple $\gamma = (V, T, A, R)$, where V, T, A are as above, and R is a finite set of triples (we call them rules with permitting contexts) $p : (r = u_1 \# u_2 \$ u_3 \# u_4; C_1, C_2)$, where r is a usual splicing rule and $C_1, C_2 \subseteq V$.

With respect to such a triple p, for $x, y, w, z \in V^*$ we write $(x, y) \vdash_p (w, z)$ iff $(x, y) \vdash_r (w, z)$, all symbols of C_1 appear in x, and all symbols of C_2 appear in y. Then, the language generated by γ is defined as usual.

We denote by $EH(p[m])$ the family of languages generated by extended H systems with permitting contexts, having rules with the maximal radius equal to $m, m \geq 1$. In [2], [4] it is proved that $EH(p[m]) = RE, \ m \geq 3$. The result is improved in [8]:

Theorem 1. $EH(p[m]) = RE, \ m \geq 2$.

That is to say, systems of radius two suffice. Stronger versions of this result will be proved in the subsequent section.

3. The width of an H system

The aim of this paper is to consider a more precise estimation of the size of the splicing rules in an extended H system. Namely, for $\gamma = (V, T, A, R)$ we define

$$width(\gamma) = (n_1, n_2, n_3, n_4),$$

where

$$n_i = \max\{|u_i| \mid u_1\#u_2\$u_3\#u_4 \in R\},\ 1 \leq i \leq 4.$$

The family of languages generated by extended H system with permitting contexts having the width less than or equal to $(n_1, n_2, n_3, n_4), n_i \geq 0, 1 \leq i \leq 4$, is denoted by $EH(p[n_1, n_2, n_3, n_4])$ (the vector ordering is the natural componentwise one).

The construction in [8] in the proof of Theorem 1 provides the equality $RE = EH(p[2, 2, 2, 2])$. However, there are $3^4 = 81$ families $EH(p[n_1, n_2, n_3, n_4])$ such that $(n_1, n_2, n_3, n_4) \leq (2, 2, 2, 2)$. We shall see below that many of them are equal to RE.

The following relations are direct consequences of the definitions.

Lemma 1. *$EH(p[n_1, n_2, n_3, n_4]) \subseteq EH(p[m_1, m_2, m_3, m_4])$, for all $0 \leq n_i \leq m_i$, $1 \leq i \leq 4$.*

From the Turing-Church thesis or by a straightforward construction, we get

Lemma 2. *$EH(p[n_1, n_2, n_3, n_4]) \subseteq RE$, for all $n_i \geq 0, 1 \leq i \leq 4$.*

Lemma 3. *$EH(p[n_1, n_2, n_3, n_4]) = EH(p[n_3, n_4, n_1, n_2])$, for all $n_i \geq 0, 1 \leq i \leq 4$.*

Proof. Consider an extended H system $\gamma = (V, T, A, R)$ and construct the system $\gamma' = (V, T, A, R')$ with

$$R' = \{(u_3\#u_4\$u_1\#u_2; C_2, C_1) \mid (u_1\#u_2\$u_3\#u_4; C_1, C_2) \in R\}.$$

Because $(x, y) \vdash_p (w, z)$ by $p = (u_1\#u_2\$u_3\#u_4; C_1, C_2)$ if and only if $(y, x) \vdash_{p'} (z, w)$ by $p' = (u_3\#u_4\$u_1\#u_2; C_2, C_1)$, we obtain $L(\gamma) = L(\gamma')$. Clearly, if $width(\gamma) = (n_1, n_2, n_3, n_4)$, then $width(\gamma') = (n_3, n_4, n_1, n_2)$. □

Lemma 4. *$RE \subseteq EH(p[0, 2, 1, 0])$.*

Proof. Consider a type-0 grammar $G = (N, T, S, P)$ containing rules of the forms $C \to x, CD \to EF$, for $C, D, E, F \in N$ and $x \in (N \cup T)^*, |x| \leq 2$ (for instance, we can take G in Kuroda normal form). Denote by P_1 the set of context-free rules in P and by P_2 the set of non-context-free rules in P. The rules in P are supposed labelled in a one-to-one manner.

We construct the extended H system with permitting contexts $\gamma = (V, T, A, R)$, where

$$\begin{aligned} V &= N \cup T \cup \{B, X, X', Z_X, Z_Y, Z_\lambda, Z'_\lambda, Y\} \\ &\cup \{Z_r \mid r \in P\} \\ &\cup \{Y_r, Z'_r \mid r \in P_2\} \\ &\cup \{Z_\alpha, Y_\alpha, Z'_\alpha \mid \alpha \in N \cup T \cup \{B\}\}, \end{aligned}$$

$$\begin{aligned} A &= \{XSBY, Z_Y Y, XZ_X, Z_\lambda, Z'_\lambda\} \\ &\cup \{Z_r xY \mid r : C \to x \in P_1\} \\ &\cup \{Z_r Y_r, Z'_r EFY \mid r : CD \to EF \in P_2\} \\ &\cup \{X'\alpha Z'_\alpha, Z_\alpha Y_\alpha \mid \alpha \in N \cup T \cup \{B\}\}, \end{aligned}$$

and R contains the following rules:

$$\begin{aligned} \textit{Simulating}: \quad & 1.\ (\#CY\$Z_r\#; \{X\}, \emptyset), \text{ for } r : C \to x \in P_1, \\ & 2.\ (\#DY\$Z_r\#; \{X\}, \emptyset), \\ & 3.\ (\#CY_r\$Z'_r\#; \{X\}, \emptyset), \text{ for } r : CD \to EF \in P_2, \\ \textit{Rotating}: \quad & 4.\ (\#\alpha Y\$Z_\alpha\#; \{X\}, \emptyset), \\ & 5.\ (\#Z'_\alpha\$X\#; \emptyset, \{Y_\alpha\}), \\ & 6.\ (\#Y_\alpha\$Z_Y\#; \{X'\}, \emptyset), \\ & 7.\ (\#Z_X\$X'\#; \emptyset, \{Y\}), \text{ for } \alpha \in N \cup T \cup \{B\}, \\ \textit{Finishing}: \quad & 8.\ (\#BY\$Z_\lambda\#; \{X\}, \emptyset), \\ & 9.\ (\#Z'_\lambda\$X\#; \emptyset, \emptyset). \end{aligned}$$

The rules of type 1 simulate the rules in P_1, the rules of types 2, 3 simulate the rules in P_2. For instance, consider a string $XwCY$ (initially we have the axiom $XBSY$) and a rule $r : C \to x \in P_1$. We get

$$(Xw|CY, Z_r|xY) \vdash_1 (XwxY, Z_rCY).$$

The string $XwxY$ is of the same form as $XwCY$, the string Z_rCY can enter only splicings using the rule $\#CY\$Z_r\#$, namely as a second term of the splicing (due to the permitting condition),

$$(Xz|CY, Z_r|CY) \vdash_1 (XzCY, Z_rCY),$$

hence nothing new is produced.

If we start from a string $XwCDY$ and a rule $r : CD \to EF \in P_2$, then we get

$$\begin{aligned} &(XwC|DY, Z_r|Y_r) \vdash_2 (XwCY_r, Z_rDY), \\ &(Xw|CY_r, Z'_r|EFY) \vdash_3 (XwEFY, Z'_rCY_r). \end{aligned}$$

Note the fact that the symbols Z_r, Z'_r precisely identify the axiom to be used as the second term of these splicings and that the "by-product" strings (those not of the form XzY, maybe with primed or subscripted versions of X and Y) will never produce new strings by splicing. (For instance, Z_rDY can only be the second term of a splicing using the rule $\#DY\$Z_r\#$, hence the suffixes DY of the splicing terms are interchanged.)

The rules of types 4, 5, 6, 7 are used for "rotating" the string: a symbol α is removed from the right hand end by a rule 4 and introduced in the left hand end by a rule 5; the markers Y_α, X' are removed by rules of types 6, 7, respectively.

The work is finished by removing the auxiliary symbols X, Y by rules 8, 9: Y can be removed only in the presence of B, which always indicates the real beginning of the

sentential form of G processed in a circular permutation by γ. Clearly, if B is placed in the left hand end of the string or in the right hand end of the string, then the string produced by γ – minus the markers X, Y, B – is a nonpermuted sentential form of G.

We leave the details of the proof to the reader. The simulate-and-rotate procedure used above is the same as that in [8] and in other places, with a different encoding of rules and axioms.

We obtain $L(G) = L(\gamma)$. Because $width(\gamma) = (0, 2, 1, 0)$, we have the inclusion $RE \subseteq EH(p[2])$. □

Combining the previous lemmas, we get the following characterizations of RE.

Theorem 2. *$RE = EH(p[n_1, n_2, n_3, n_4])$, for all $(n_1, n_2, n_3, n_4) \geq (0, 2, 1, 0)$ or $(n_1, n_2, n_3, n_4) \geq (1, 0, 0, 2)$.*

In the proof of Lemma 4, the application of rules of the grammar G was simulated in the equivalent extended H system in the right hand end of the strings. It is easy to see that we can perform this simulation also in the left hand end of the strings. Also the rotation can be done in the reverse direction to that in the proof of Lemma 4: cut a symbol from the left end of the string and add it in the right hand end, repeatedly. A counterpart of the result in Lemma 4 can be obtained in this way.

Lemma 5. *$RE \subseteq EH(p[2, 0, 0, 1])$.*

Proof. We simply repeat the construction in the proof of Lemma 4. We present here only the splicing rules of the obtained system, γ, together with the axioms used by each of them; the "main axiom" is $XSBY$.

Simulating :	1.	$(XC\#\$\#Z_r; \{Y\}, \emptyset)$,	for $r : C \to x \in P_1$,	axiom : XxZ_r,
	2.	$(XC\#\$\#Z_r; \{Y\}, \emptyset)$,		axiom : X_rZ_r,
	3.	$(X_rD\#\$\#Z'_r; \{Y\}, \emptyset)$,	for $r : CD \to EF \in P_2$,	axiom : $XEFZ'_r$,
Rotating :	4.	$(X\alpha\#\$\#Z_\alpha; \{Y\}, \emptyset)$,		axiom : $X_\alpha Z_\alpha$,
	5.	$(Z'_\alpha\#\$\#Y; \emptyset, \{X_\alpha\})$,		axiom : $Z'_\alpha \alpha Y'$,
	6.	$(X_\alpha\#\$\#Z_X; \{Y'\}, \emptyset)$,		axiom : XZ_X,
	7.	$(Z_Y\#\$\#Y'; \emptyset, \{X\})$,	for $\alpha \in N \cup T \cup \{B\}$,	axiom : Z_YY,
Finishing :	8.	$(XB\#\$\#Z_\lambda; \{Y\}, \emptyset)$,		axiom : Z_λ,
	9.	$(Z'_\lambda\#\$\#Y; \emptyset, \emptyset)$		axiom : Z'_λ.

As in the proof of Lemma 4 we obtain $L(G) = L(\gamma)$. Clearly, $width(\gamma) = (2, 0, 0, 1)$. □

Theorem 3. *$RE = EH(p[n_1, n_2, n_3, n_4])$, for all $(n_1, n_2, n_3, n_4) \geq (2, 0, 0, 1)$ or $(n_1, n_2, n_3, n_4) \geq (0, 1, 2, 0)$.*

Lemma 6. *$RE \subseteq EH(p[1, 2, 0, 1])$.*

Proof. Consider a type-0 grammar $G = (N, T, S, P)$ as in the proof of Lemma 4. We assume the rules in P labelled in a one-to-one manner.

We construct the extended H system with permitting contexts $\gamma = (V, T, A, R)$, where

$$V = N \cup T \cup \{B, X, X', Z_X, Z_Y, Z_\lambda, Z'_\lambda, Y, Y'_B\}$$

$$
\begin{aligned}
&\cup \ \{Z_r \mid r \in P\} \\
&\cup \ \{Y_r, Z'_r \mid r : CD \to EF \in P\} \\
&\cup \ \{Z_\alpha, Y_\alpha, Z'_\alpha \mid \alpha \in N \cup T \cup \{B\}\}, \\
A \ &= \ \{XSBY, Z_Y Y, XZ_X, Z_\lambda, Z'_\lambda, Y'_B\} \\
&\cup \ \{Z_r xY \mid r : C \to x \in P\} \\
&\cup \ \{Z_r Y_r, Z'_r EFY \mid r : CD \to EF \in P\} \\
&\cup \ \{X'\alpha Z'_\alpha, Z_\alpha Y_\alpha \mid \alpha \in N \cup T \cup \{B\}\},
\end{aligned}
$$

and R contains the following rules:

$$
\begin{aligned}
\textit{Simulating}: \quad & 1. \quad (\#CY\$\#Y; \{X\}, \{Z_r\}), \text{ for } r : C \to \lambda \in P, \\
& 1'. \quad (\#CY\$\#\alpha; \{X\}, \{Z_r\}), \text{ for } r : C \to \alpha \in P, \alpha \in N \cup T, \\
& 1''. \quad (\#CY\$\#\alpha; \{X\}, \{Z_r\}), \text{ for } r : C \to \alpha\beta \in P, \alpha, \beta \in N \cup T, \\
& 2. \quad (\#DY\$\#Y_r; \{X\}, \{Z_r\}), \\
& 3. \quad (\#CY_r\$\#E; \{X\}, \{Z'_r\}), \text{ for } r : CD \to EF \in P, \\
\textit{Rotating}: \quad & 4. \quad (\#\alpha Y\$\#Y_\alpha; \{X\}, \{Z_\alpha\}), \\
& 5. \quad (X\#\$\#Z'_\alpha; \{Y_\alpha\}, \emptyset), \\
& 6. \quad (\#Y_\alpha\$\#Y; \{X'\}, \{Z_Y\}), \\
& 7. \quad (X'\#\$\#Z_X; \{Y\}, \emptyset), \text{ for } \alpha \in N \cup T \cup \{B\}, \\
\textit{Finishing}: \quad & 8. \quad (\#BY\$\#Y'_B; \{X\}, \emptyset), \\
& 9. \quad (Z_\lambda\#\$\#Y'_B; \emptyset, \{X\}), \\
& 10. \quad (X\#\$\#Z'_\lambda; \emptyset, \emptyset).
\end{aligned}
$$

The proof of the equality $L(G) = L(\gamma)$ is similar to the proofs of the corresponding equalities in the proofs of Lemmas 4 and 5. The main difference consists of the way of ensuring the correct use of the splicing rules (for instance, the fact that one of the terms of the splicing is an axiom or an "axiom-like" string produced at a previous splicing): this is done both by the symbols appearing in the splicing rules and in their permitting contexts. For example, using a rule of type 1 we get

$$(Xw|CY, Z_r|Y) \vdash_1 (XwY, Z_r CY).$$

The use of the axiom $Z_r Y$ is forced by the context Z_r of this rule. The string XwY is of the same form as $XwCY$, the "by-product" string $Z_r CY$ can enter only splicings using the same rule $\#CY\$\#$:

$$(Xz|CY, Z_r C|Y) \vdash_1 (XzY, Z_r C^2 Y).$$

One sees that the process can be iterated but no "illegal" string is obtained, always the rule $C \to \lambda$ is simulated in the right hand end of a string of the form XzY.

Using a rule of type $1'$ we get

$$(Xw|CY, Z_r|\alpha Y) \vdash_{1'} (Xw\alpha Y, Z_r CY).$$

Now, $Z_r CY$ can enter a new splicing only if $C = \alpha$, hence the terms of the splicing are reproduced.

The rotation phase is the same as in the proof of Lemma 4. A difference appear in the terminating phase: by the rule of type 8 we first replace BY by Y'_B (this ensures the fact that the string is in the correct permutation; note that $Y'_B \neq Y_B$, where Y_B is the symbol used in the rotating phase), and then Y'_B is removed by rule 9. □

Simulating the rules of G in the left hand end of the strings of γ we can obtain the following counterpart of Lemma 6 (in the same way as Lemma 5 corresponds to Lemma 4); the details are left to the reader.

Lemma 7. *$RE \subseteq EH(p[2,1,1,0])$.*

Synthesizing these results, we obtain further characterizations of RE.

Theorem 4. *$RE = EH(p[n_1,n_2,n_3,n_4])$, for all $(n_1,n_2,n_3,n_4) \geq (1,2,0,1)$, or $(n_1,n_2,n_3,n_4) \geq (0,1,1,2)$, or $(n_1,n_2,n_3,n_4) \geq (2,1,1,0)$, or $(n_1,n_2,n_3,n_4) \geq (1,0,2,1)$.*

4. Generating context-free languages

Theorems 2, 3, 4 give a precise characterization of 49 families of the form $EH(p[n_1,n_2,n_3,n_4])$; thus, out of the 81 families of this type, 32 remain to be further investigated. We *conjecture* that all these remaining families are strictly included in RE.

This conjecture is related to the *open problem* whether or not systems of radius 1 can characterize the family RE. In [8] one formulates the conjecture that this is not the case. More specifically, in [8] it is conjectured that $EH(p[1]) = CF$. The inclusion $CF \subseteq EH(p[1])$ is confirmed in [1], the converse inclusion is still open.

The construction in [1] proves the relation $CF \subseteq EH(p[1,1,1,1])$. Using similar techniques as in the previous section, we can improve this result.

Theorem 5. *$CF \subseteq EH(p[0,1,1,0])$.*

Proof. Consider a context-free grammar $G = (N,T,S,P)$ in the strong Chomsky normal form (see, for instance, [14]), that is with the rules in P of the forms $X \to a$, $X \to YZ$, for $X,Y,Z \in N, a \in T$, and with the additional restrictions

1. if $X \to YZ$ is in P, then $Y \neq Z$,
2. if $X \to YZ$ is in P, then for each rule $X \to Y'Z'$ in P we have $Z' \neq Y$ and $Y' \neq Z$.

We construct the permitting context H system $\gamma = (V,T,A,R)$, where

$$\begin{aligned} V &= T \cup \{B_1, B_2, B'_2, Z_{B_1}, Z_{B_2}, Z_{B'_2} \mid B \in N\} \cup \{Z, Z'\} \\ A &= \{B_1 a B_2 \mid B \to a \in P\} \\ &\cup \{B_1 Z_{B_1}, Z_{B_2} B_2, Z_{B'_2} B'_2 \mid B \in N\} \\ &\cup \{Z, Z'\}, \end{aligned}$$

and R consists of the following splicing rules:

$$\begin{array}{lll} 1) & (\#C_2\$D_1\#;\{C_1\},\{D_2\}), & \text{for } B \to CD \in P, \\ 2) & (\#D_2\$Z_{B_2'}\#;\{C_1\},\{B_2'\}), & \text{for } B \to CD \in P, \\ 3) & (\#Z_{B_1}\$C_1\#;\{B_1\},\{B_2'\}), & \text{for } B \to CD \in P, \\ 4) & (\#B_2'\$Z_{B_2}\#;\{B_1\},\{B_2\}), & \text{for } B \to CD \in P, \\ 5) & (\#S_2\$Z\#;\{S_1\},\emptyset), & \\ 6) & (\#Z'\$S_1\#;\emptyset,\emptyset), & . \end{array}$$

The basic idea of this construction is the same as in [1]: a string B_1wB_2, with $w \in T^+, B \in N$, is generated in γ if and only if $B \Longrightarrow^* w$ in the grammar G. When $B = S$, then also w can be generated in γ.

Consider two strings $C_1w_1C_2, D_1w_2D_2$ produced by γ; the axioms in A corresponding to terminal rules in P are of this form, all other axioms are of a different form. If there is a rule $B \to CD$ in P, then the associated splicing rules of types 1, 2, 3, 4 exist in R, hence we can perform

$$\begin{array}{l} (C_1w_1|C_2, D_1|w_2D_2) \vdash_1 (C_1w_1w_2D_2, D_1C_2), \\ (C_1w_1w_2|D_2, Z_{B_2'}|B_2') \vdash_2 (C_1w_1w_2B_2', Z_{B_2'}D_2), \\ (B_1|Z_{B_1}, C_1|w_1w_2B_2') \vdash_3 (B_1w_1w_2B_2', C_1Z_{B_1}), \\ (B_1w_1w_2|B_2', Z_{B_2}|B_2) \vdash_4 (B_1w_1w_2B_2, Z_{B_2}B_2'). \end{array}$$

The use of the rule $B \to CD$ in P has been simulated by the passing from $C_1w_1C_2, D_1w_2D_2$ to $B_1w_1w_2B_2$.

Due to the form of the splicing rules (especially, their permitting context conditions), the strings $Z_{B_2'}D_2, Z_{B_2}B_2'$ cannot enter new splicings. The string $C_1Z_{B_1}$ can enter a splicing using a rule of type 3 only when $C = B$, hence nothing new is produced.

The string D_1C_2 can enter a splicing using a rule of type 1 only when $D = C$, and this contradicts condition 1 on the form of rules in P. A rule of type 2 can involve this string, leading to a splicing of the form

$$(D_1|C_2, Z_{B_2'}|B_2') \vdash_2 (D_1B_2', Z_{B_2'}C_2),$$

only if $B \to DC \in P$; this possibility is ruled out by condition 2 on the form of rules in P.

Therefore, no "illegal" splicing is possible.

When we obtain a string of the form S_1wS_2, we can remove S_1 and S_2 by means of rules 5 and 6:

$$\begin{array}{l} (S_1w|S_2, Z|) \vdash_5 (S_1w, ZS_2), \\ (|Z', S_1|w) \vdash_6 (w, S_1Z'). \end{array}$$

Consequently, $L(G) = L(\gamma)$; as $width(\gamma) = (0, 1, 1, 0)$, the proof is complete. □

4. Final remarks

Another type of a control mechanism, leading to results similar to those obtained for extended H systems with permitting contexts, is that using *forbidding* contexts. A splicing rule with forbidding context conditions is a triple $p = (r; D_1, D_2)$, where r is a usual splicing rule and D_1, D_2 are sets of symbols. The rule p is used for splicing two strings x and y only if no symbol from D_1 appears in x and no symbol from D_2 appears in y. The definition of an extended H system with forbidding context conditions is natural, similar to that of a system with permitting contexts.

As one can see in the previous proof, we always check the occurrence of the symbol X or of variants of it (primed, subscripted, etc) in the leftmost position of a term of the splicing and the occurrence of Y or of variants of it in the rightmost position of a term of the splicing. Always, an element Z from a precise set Q is checked. This is clearly equivalent with checking the non-occurrence of all symbols in $Q - \{Z\}$ (the form of axioms guarantees that all strings leading to a terminal string start with X and end with Y, or with variants of them, respectively – of course, excepting the last two steps of a computation, when these symbols are removed). Thus, all results obtained above for H systems with permitting contexts hold true for H systems with forbidding contexts, too.

It remains to be investigated the possibility of characterizing RE using H systems of small radius working with multisets. In [2], [4] one uses H systems of this type of radius 5, but this bound can probably be improved.

References

[1] V. T. Chakaravarthy, K. Krithivasan, A note on extended H systems with permitting/forbidding contexts of radius one, *Bulletin of the EATCS*, 1997 (in press).

[2] E. Csuhaj-Varju, L. Freund, L. Kari, Gh. Păun, DNA computing based on splicing: universality results, *First Annual Pacific Symp. on Biocomputing*, Hawaii, Jan. 1996.

[3] K. Culik II, T. Harju, Splicing semigroups of dominoes and DNA, *Discrete Appl. Math.*, 31 (1991), 261 – 277.

[4] R. Freund, L. Kari, Gh. Păun, DNA computing based on splicing: The existence of universal computers, Technical Report 185-2/FR-2/95, Technical Univ. Wien, 1995.

[5] T. Head, Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors, *Bull. Math. Biology*, 49 (1987), 737 – 759.

[6] T. Head, Gh. Păun, D. Pixton, Language theory and molecular genetics. Generative mechanisms suggested by DNA recombination, in *Handbook of Formal Languages* (G. Rozenberg, A. Salomaa, eds.), Springer-Verlag, Berlin, Heidelberg, 1997.

[7] Gh. Păun, Regular extended H systems are computationally universal, *J. Automata, Languages and Combinatorics*, 1, 1 (1996), 27 – 36.

[8] Gh. Păun, Computing by splicing: How simple rules ?, *Bulletin of the EATCS*, 60 (1996), 145 – 150.

[9] Gh. Păun, Splicing systems with targets are computationally complete, *Inform. Processing Letters*, 59 (1996), 129 – 133.

[10] Gh. Păun, G. Rozenberg, A. Salomaa, Restricted use of the splicing operation, *Intern. J. Computer Math.*, 60 (1996), 17 – 32.

[11] Gh. Păun, G. Rozenberg, A. Salomaa, Computing by splicing, *Theoretical Computer Science*, 168, 2 (1996), 321 – 336.

[12] D. Pixton, Regularity of splicing languages, *Discrete Appl. Math.*, 69 (1996), 101 – 124.

[13] G. Rozenberg, A. Salomaa (eds.), *Handbook of Formal Languages*, 3 volumes, Springer-Verlag, Berlin, Heidelberg, 1997.

[14] D. Wood, *Theory of Computation*, Harper and Row Publishers, New York, 1987.