TEST TUBE SYSTEMS WITH CUTTING/RECOMBINATION OPERATIONS

Rudolf FREUND
Institut für Computersprachen, Technische Universität Wien
Resselgasse 3, 1040 Wien, Austria
email: rudi@logic.tuwien.ac.at

Erzsébet CSUHAJ-VARJÚ
Computer and Automation Institute, Hungarian Academy of Sciences
Kende u. 13-17, 1111 Budapest, Hungary
email: csuhaj@luna.aszi.sztaki.hu

Franz WACHTLER
Institut für Histologie und Embryologie, Universität Wien
Schwarzspanierstr. 17, 1090 Wien, Austria
email: franz.wachtler@univie.ac.at

We introduce test tube systems[1,2,3,9] based on operations that are closely related to the splicing operation[8], i.e. we consider the operations of *cutting* a string at a specific site into two pieces with marking them at the cut ends and of *recombining* two strings with specificly marked endings[7]. Whereas in the splicing of two strings these strings are cut at specific sites and the cut pieces are recombined immediately in a crosswise way, in CR(cutting/recombination)-schemes cutting can happen independently from recombining the cut pieces. Test tube systems based on these operations of cutting and recombination turn out to have maximal generative power even if only very restricted types of input filters for the test tubes are used for the redistribution of the contents of the test tubes after a period of cuttings and recombinations in the test tubes.

1 Introduction

Test tube systems were introduced as biological computer systems based on DNA molecules[1,2,3,9], the practical solution of NP problems with such systems was described, and the theoretical features of test tube systems based on the splicing operation were investigated[4]. In this paper we are going to explore test tube systems based on the operations of cutting and recombination[7] and with input filters which only allow specific parts of the contents of the other test tubes to pass (such filters, for example, are conceivable for DNA molecules by combining the principles of affinity chromatography and in-situ-hybridization). As we shall show even very restricted kinds of such filters testing for the existence respectively non-existence of specific markings allow for reaching the generative power to generate any recursively enumerable language.

The computational universality of specific variants of H systems [5,10] and for test tube systems based on the splicing operation [4] has been proved recently. In this paper we shall show that universal test tube systems based on cutting and recombination rules exist for different variants of input filters.

In the second section of this paper we define the notions from formal language theory needed in this paper and introduce the formal definitions of cutting/recombination schemes (CR-schemes). In the third section of this paper we introduce test tube systems with cutting/recombination rules (CRTTS) and we prove that CRTTS can generate every recursively enumerable language. This result also implies the existence of universal CRTTS. A short summary of the results obtained in this paper and an overview of future research topics conclude the paper.

2 Definitions and Examples

In this section we only define some notions from formal language theory that we shall need in this paper. Moreover we recall the definitions for CR-schemes [7] and give some explanatory examples.

The free monoid generated by the alphabet V is denoted by V^*, its elements are called *strings* or *words* over V; λ is the empty string, $V^+ = V^* - \{\lambda\}$. The length of a string w in V^* is written as $|w|$.

A *grammar scheme* γ is a triple (V_N, V_T, P), where V_N is a (finite) set of symbols, i.e. the alphabet of *non-terminal symbols;* V_T is a (finite) set of symbols with $V_N \cap V_T = \emptyset$, i.e. the alphabet of *terminal symbols*; P is a (finite) set of *productions* of the form (α, β), where $\alpha \in (V_N \cup V_T)^+$ and $\beta \in (V_N \cup V_T)^*$. For two words $x, y \in (V_N \cup V_T)^*$, the *derivation relation* $\vdash_\gamma$ is defined if and only if $x = u\alpha v$ and $y = u\beta v$ for some production $(\alpha, \beta) \in P$ and two strings $u, v \in (V_N \cup V_T)^*$; we then also write $x \vdash_\gamma y$. The reflexive and transitive closure of the relation $\vdash_\gamma$ is denoted by $\vdash_\gamma^*$.

A *grammar* G is a quadruple (V_N, V_T, P, S), where $\gamma = (V_N, V_T, P)$ is a grammar scheme and $S \in V_N$; in a more general way, we can also take $S \in (V_N \cup V_T)^+$. The λ-free language generated by G is

$$L(G) = \left\{ w \in V_T^+ \mid S \vdash_\gamma^* w \right\}. \tag{1}$$

A subset L of V_T^+ is called *recursively enumerable* if and only if there exists a grammar G that generates L, i.e. $L(G) = L$. Moreover, L is called *recursive* if and only if both L and its complement, $V_T^+ \setminus L$, are recursively enumerable. The family of (λ-free) recursively enumerable languages and the family of (λ-free) recursive languages over the alphabet V_T shall be denoted by $ENUM(V_T)$ and $REC(V_T)$, respectively.

A *grammar scheme* γ_U with $\gamma_U = (V_N, V_T, P)$ is called *universal (for V_T)* if for every $L \in ENUM(V_T)$ there exists a word A_L such that the grammar G_L with $G_L = (V_N, V_T, P, A_L)$ generates L. One of the important results of formal language theory is that for every V_T such a universal grammar γ_U exists.

Definition 1. A *cutting/recombination scheme* (or a *CR-scheme*) is a quadruple $\sigma = (V, M, C, R)$, where V is a finite alphabet; M is a finite set of *markings;* V and M are disjoint sets; C is a set of *cutting rules* of the form $u\#l\$m\#v$, where $u \in V^* \cup MV^*$, $v \in V^* \cup V^*M$, and $m, l \in M$, and $\#, \$$ are special symbols not in $V \cup M$; $R \subseteq M \times M$ is the recombination relation representing the *recombination rules.*

Cutting and recombination rules are applied to objects from $O(V, M)$, where we define

$$O(V, M) = V^+ \cup MV^* \cup V^*M \cup MV^*M, \tag{2}$$

i.e., as the empty word has no meaningful representation in nature, λ is not considered to be an object we have to deal with.

For $x, y, z \in O(V, M)$ and a cutting rule $c = u\#l\$m\#v$ we define $x \vdash_c (y, z)$ if and only if for some $\alpha \in V^* \cup MV^*$ and $\beta \in V^* \cup V^*M$ we have $x = \alpha uv\beta$ and $y = \alpha ul$, $z = mv\beta$. For $x, y, z \in O(V, M)$ and a recombination rule $r = (l, m)$ from R we define $(x, y) \vdash_r z$ if and only if for some $\alpha \in V^* \cup MV^*$ and $\beta \in V^* \cup V^*M$ we have $x = \alpha l$, $y = m\beta$, and $z = \alpha\beta$. For a CR-scheme $\sigma = (V, M, C, R)$ and any language $L \subseteq O(V, M)$ we write

$$\begin{aligned} \sigma(L) \quad = \quad & \{y \mid x \vdash_c (y, z) \text{ or } x \vdash_c (z, y) \text{ for some } x \in L,\ c \in C\} \cup \\ & \{z \mid (x, y) \vdash_r z \text{ for some } x, y \in L,\ r \in R\}, \end{aligned} \tag{3}$$

and we define $\sigma^*(L) = \bigcup_{i \geq 0} \sigma^i(L)$, where $\sigma^0(L) = L$ and $\sigma^{i+1}(L) = \sigma^i(L) \cup \sigma\left(\sigma^i(L)\right)$ for all $i \geq 0$. □

Thus $\sigma(L)$ contains all objects obtained by applying one cutting or one recombination rule to objects from L; $\sigma^*(L)$ is the smallest subset of $O(V, M)$ that contains L and is closed under the cutting and recombination rules of σ.

There is a close relationship between CR schemes and *splicing schemes (H schemes):* For short, a *splicing rule* $u_1\#v_1\$u_2\#v_2$ being applied to two strings $x_1u_1v_1y_1$ and $x_2u_2v_2y_2$ yields the two strings $x_1u_1v_2y_2$ and $x_2u_2v_1y_1$ which corresponds to cutting the strings $x_1u_1v_1y_1$ and $x_2u_2v_2y_2$ into the strings $x_1u_1[m]^+$, $[m]^-v_1y_1$ and $x_2u_2[m]^+$, $[m]^-v_2y_2$ by using the cutting rules $u_1\#[m]^+\$[m]^-\#v_1$ and $u_2\#[m]^+\$[m]^-\#v_2$ and recombining them immediately by applying the recombination rule $\left([m]^+, [m]^-\right)$ in a crosswise way.

In the following we shall restrict ourselves to cutting rules of the form $u\#\,[m]^+\,\$\,[m]^-\,v$, i.e. the markings generated by the cutting rule are the *positive* $\left([m]^+\right)$ and the *negative* $\left([m]^-\right)$ version of $[m]$. In this case the derivation of the two parts $xu\,[m]^+$, $[m]^-\,vy$ from the object $xuvy$ by the cutting rule $u\#\,[m]^+\,\$\,[m]^-\,\#v$, i.e. the derivation $xuvy \vdash_{u\#[m]^+\$[m]^-\#v} \left(xu\,[m]^+, [m]^-\,vy\right)$ in a more depictive way can be expressed by $xu \underset{[m]}{|} vy \vdash \left(xu\,[m]^+, [m]^-\,vy\right)$.

The following example shows the chemical background of the notations introduced above, i.e. the markings $[m]^+$ and $[m]^-$, respectively, correspond to the positive and negative charges of ions:

Example 1. Consider the salt molecule $NaCl$, which in water dissipates to the ions $Na+$ and $Cl-$. This reaction corresponds to applying the formal cutting rule $Na\#\,[m]^+\,\$\,[m]^-\,\#Cl$ to the molecule string $NaCl$, which yields the formal derivation step $Na \underset{[m]}{|} Cl \vdash \left(Na\,[m]^+, [m]^-\,Cl\right)$. Obviously the two parts $Na\,[m]^+$, $[m]^-\,Cl$ can be recombined to $NaCl$ by the recombination rule $\left([m]^+, [m]^-\right)$, i.e. $\left(Na\,[m]^+, [m]^-\,Cl\right) \vdash NaCl$. □

3 CR Test Tube Systems

In this section we introduce test tube systems[1,2,3,9] that are based on the formal operations of cutting and recombination rules as introduced in the previous section. The idea of test tube systems is to describe computational devices where the computations in each test tube are based on specific operations and any computation is done in a distributed way. As a communication step the resulting contents of the test tubes then is redistributed according to specific constraints, i.e. the contents of each test tube is distributed to all test tubes according to specific input filters again, whereas the rest remains in the test tube. These ideas have already been formalized for the splicing operation[4]; in the following we define test tube systems where the operations that can take place in one test tube are cuttings and recombinations and investigate the generative power of these systems with the operations of cuttings and recombinations. We shall show that every recursively enumerable language can be generated by such a test tube system which only needs a very special restricted kind of input filters. Moreover, this result also guarantees the existence of a universal test tube system with cuttings and recombinations.

Definition 2. A *test tube system with cuttings and recombinations* (a *CRTTS*

for short) σ is a quintuple (V, M, A, n, ρ, I), where

1. V is a (finite) set of *symbols;*
2. M is a (finite) set of *markings;* M and V are disjoint sets;
3. A is a (finite) set of *axioms,* which are elements from $O(V, M)$;
4. n, $n \geq 1$, is the number of test tubes;
5. ρ is a (finite) sequence $(\rho_1, ..., \rho_n)$, where $\rho_i = (C_i, R_i)$ is a finite set of *test tube operations of cuttings and recombinations,* respectively, in the test tube T_i, i.e. C_i is a (finite) set of cutting rules over (V, M) and R_i is a (finite) set of recombination rules over (V, M); $\sigma_i = (V, M, C_i, R_i)$ is the corresponding CR-scheme;
6. $I = (I_1, ..., I_n)$, where $I_i \subseteq O(V, M)$ is the *input filter* for the test tube T_i, $1 \leq i \leq n$.

The computations in the system σ run as follows: At the beginning of the computation the axioms are distributed over the n test tubes T_i according to the corresponding input filters I_i, i.e. T_i starts with $A \cap I_i$. Now let L_i be the contents of T_i at the beginning of a derivation step. Then in each test tube the CR-scheme σ_i operates on L_i, i.e. we obtain $\sigma_i^*(L_i)$. The next substep is the redistribution of $\sigma_i^*(L_i)$ over all test tubes according to the corresponding input filters. From $\sigma_i^*(L_i)$ only the part $\sigma_i^*(L_i) \cap I_j$ that passes the input filter I_j is distributed to the test tubes T_j, $1 \leq j \leq n$, whereas the rest $\sigma_i^*(L_i) \setminus \left(\bigcup_{1 \leq j \leq n} (\sigma_i^*(L_i) \cap I_j)\right)$ remains in T_i. The final result of the computations in σ consists of all strings from V^+ that can be extracted from the *final test tube* T_1.

More formally, an *instantaneous description* (ID for short) of the system σ is an n-tuple $(L_1, ..., L_n)$ with $L_i \subseteq O(V, M)$, $1 \leq i \leq n$, where L_i describes the contents of test tube T_i at the beginning of a derivation step. The initial ID is $(A \cap I_1, ..., A \cap I_n)$, i.e. at time $t = 0$ the test tubes T_i contain the axioms $A \cap I_i$. Let $(L_1(t), ..., L_n(t))$ denote the ID at time t; then one derivation step with the system σ yields the ID $(L_1(t+1), ..., L_n(t+1))$, where

$$\begin{aligned} L_i(t+1) &= \left(\bigcup_{1 \leq j \leq n} (\sigma_j^*(L_j(t)) \cap I_i)\right) \cup \\ &\quad \left(\sigma_i^*(L_i(t)) \setminus \left(\bigcup_{1 \leq j \leq n} (\sigma_i^*(L_i(t)) \cap I_j)\right)\right) \\ &= \left(\left(\bigcup_{1 \leq j \leq n} \sigma_j^*(L_j(t))\right) \cap I_i\right) \cup \\ &\quad \left(\sigma_i^*(L_i(t)) \setminus \left(\sigma_i^*(L_i(t)) \cap \bigcup_{1 \leq j \leq n} I_j\right)\right). \end{aligned} \tag{4}$$

We also write $(L_1(t), ..., L_n(t)) \vdash_\sigma (L_1(t+1), ..., L_n(t+1))$. The language generated by σ, $L(\sigma)$, is defined by $L(\sigma) = \bigcup_{t=0}^{\infty} (L_1(t) \cap V^+)$. □

A minimal requirement on the feasability of the input filters I_i is their recursiveness, i.e. we demand that it is decidable whether a string can pass the filter or not. Yet in order to obtain more interesting results we have to put restrictions on the input filters:

Definition 3. A subset of $O(V, M)$ is called a *simple* $(V, M)_2$*-filter* if it equals

1. V^+ or
2. $\{m\} V^*$ for some $m \in M$ or
3. $V^* \{m\}$ for some $m \in M$ or
4. $\{m\} V^* \{n\}$ for some $m, n \in M$.

A simple $(V, M)_2$-filter is called a *simple* $(V, M)_1$*-filter*, if it is not of the form $\{m\} V^* \{n\}$. Any finite union of simple $(V, M)_i$-filters, $i \in \{1, 2\}$, is called a $(V, M)_i$*-filter.* □

In the following example we show how the language $\left\{a^{2^n} \mid n \geq 1\right\}$ can be generated by a CRTTS with $(V, M)_1$-filters:

Example 2. Let $\sigma = (V, M, A, 8, \rho, I)$ be the CRTTS with

$V = \{a, B, F, X, Y\}$, $M = \left\{[x]^+, [x]^- \mid x \in \{b, c, d, e, f, l, n, r, s, t\}\right\}$,

$A = \{XaaBY, XaaY, XBY, F\}$,

$\rho = (\rho_1, \rho_2, \rho_3, \rho_4, \rho_5, \rho_6, \rho_7, \rho_8)$, $I = (I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8)$,

$\rho_1 = \left(\emptyset, \left\{\left([s]^+, [e]^-\right)\right\}\right)$,

$\rho_2 = \left(\left\{X\#[e]^+ \$ [e]^- \#aa, \#[s]^+ \$ [s]^- \#F\right\}, \left\{\left([f]^+, [t]^-\right)\right\}\right)$,

$\rho_3 = \left(\left\{aa\#[f]^+ \$ [f]^- \#BY, F\#[t]^+ \$ [t]^- \#\right\}, \emptyset\right)$,

$\rho_4 = \left(\left\{aa\#[b]^+ \$ [b]^- \#BY, XB\#[c]^+ \$ [c]^- \#Y\right\}, \emptyset\right)$,

$\rho_5 = \left(\begin{array}{l} \left\{X\#[d]^+ \$ [d]^- \#aaa, X\#[d]^+ \$ [d]^- \#aaY\right\}, \\ \left\{\left([b]^+, [c]^-\right), \left([c]^+, [d]^-\right)\right\} \end{array}\right)$,

$\rho_6 = \left(\begin{array}{l} \left\{w\#[r]^+ \$ [r]^- \#aY \mid w \in \{aa, aB, Ba\}\right\} \cup \\ \left\{Xaa\#[n]^+ \$ [n]^- \#Y\right\}, \emptyset \end{array}\right)$,

$\rho_7 = \left(\left\{X\#[l]^+ \$ [l]^- \#v \mid v \in \{aaa, aaB, BaY\}\right\}, \left\{\left([r]^+, [n]^-\right)\right\}\right)$,

$\rho_8 = \left(\emptyset, \left\{\left([n]^+, [l]^-\right)\right\}\right)$;

$I_1 = V^* \left\{[s]^+\right\} \cup \left\{[e]^-\right\} V^*, I_2 = V^* \left\{[f]^+\right\} \cup \left\{[t]^-\right\} V^* \cup V^* \left\{[t]^+\right\}$,

$I_3 = I_4 = I_6 = V^+$, $I_5 = V^* \left\{[b]^+\right\} \cup V^* \left\{[c]^+\right\} \cup \left\{[c]^-\right\} V^*$,

$I_7 = V^* \left\{[r]^+\right\} \cup \left\{[n]^-\right\} V^*$, $I_8 = V^* \left\{[n]^+\right\} \cup \left\{[l]^-\right\} V^*$.

The generation of the words a^{2^n} in this CR test tube system briefly can be described in the following way:

In general, let us assume we have already obtained the word $Xa^{2^n}BY$ for some $n \geq 1$ in test tube T_4 (originally we start with the axiom $XaaBY$), where the following cuttings can take place: $Xa^{2^n} \underset{[b]}{|} BY \vdash \left(Xa^{2^n}[b]^+, [b]^- BY\right)$ and $XB \underset{[c]}{|} Y \vdash \left(XB[c]^+, [c]^- Y\right)$. In T_5 we then obtain $\left(Xa^{2^n}[b]^+, [c]^- Y\right) \vdash Xa^{2^n}Y$ by the recombination rule $\left([b]^+, [c]^-\right)$, and by cutting with one of the rules $X\#[d]^+\$[d]^-\#aaa, X\#[d]^+\$[d]^-\#aaY$ we get $X \underset{[d]}{|} a^{2^n}Y \vdash \left(X[d]^+, [d]^- a^{2^n}Y\right)$. The recombination rule $\left([c]^+, [d]^-\right)$ then yields $\left(XB[c]^+, [d]^- a^{2^n}Y\right) \vdash XBa^{2^n}Y$. In sum, we have rotated the symbol B from the end to the beginning of the block of symbols a.

The rules in the test tubes T_6, T_7 and T_8 have the effect that a symbol a at the end of the word to the left of the symbol Y is eliminated, yet instead two symbols a are added at the beginning, i.e. from Xa^kBa^mY we obtain $Xa^{k+2}Ba^{m-1}Y$. A full cycle of rotations therefore doubles the number of symbols a, i.e. from $XBa^{2^n}Y$ we obtain $Xa^{2^{n+1}}BY$.

The test tubes T_3, T_2, and T_1 finally allow us to obtain the terminal strings a^{2^n} : In T_3 we get $Xa^{2^n} \underset{[f]}{|} BY \vdash \left(Xa^{2^n}[f]^+, [f]^- BY\right)$ and $F \underset{[t]}{|} F[t]^+, [t]^-$. The objects $Xa^{2^n}[f]^+$, $F[t]^+$, and $[t]^-$ are passed to T_2, where we have $\left(Xa^{2^n}[f]^+, [t]^-\right) \vdash Xa^{2^n}$, $X \underset{[e]}{|} a^{2^n} \vdash \left(X[e]^+, [e]^- a^{2^n}\right)$, as well as $\underset{[s]}{|} F[t]^+ \vdash \left([s]^+, [s]^- F[t]^+\right)$. Passing the objects $[e]^- a^{2^n}$ and $[s]^+$ to T_1, we finally obtain the terminal word a^{2^n} by $\left([s]^+, [e]^- a^{2^n}\right) \vdash a^{2^n}$. □

Remark 1. In general, the formal definitions allow an infinite number of objects to be generated in one derivation step, which is an unnatural situation for practical implementations. A more practical assumption would be that

instead of $\sigma_i^*(L_i)$ any arbitrary (finite) subset of $\sigma_i^*(L_i)$ can evolve in the test tube T_i during a computation period. Then only this subset is distributed to all test tubes according to the input filters. In fact, for all examples and all constructions in the proofs of this paper such an interpretation of the computations in the test tubes still would allow us to generate all desired objects, although it would never be clear, when these objects would evolve. In a practical environment the number and the size of objects that can be generated also depends on the amount of original material of axioms we take at the beginning. Moreover, if parts of (the subset of) $\sigma_i^*(L_i)$ are to be redistributed over different test tubes it is only necessary to assume that any allowed distribution of the whole material will possibly happen; in practical implementations of test tube systems an intermediate amplification [2,9] of the material may already guarantee that enough material is distributed to all the possible test tubes.

Similar ideas as for the construction of the CRTTS in the preceding example can be used for proving the general result established in the following theorem:

Theorem 1. *For every recursively enumerable language L, $L \subseteq V_T^+$, we can construct a CRTTS σ_L with $(V,M)_1$-filters which generates L.*

Proof. Let L be given by a grammar $G'_L = (V'_N, V_T, P', S')$, i.e. $L(G'_L) = L$. Without loss of generality we can consider the language $L\{h\}$ instead of L, where $h \notin (V'_N \cup V_T)$ is a new symbol; let $G_L = (V_N, V_T, P, S)$ be a grammar such that $L(G_L) = L\{h\}$ and moreover for each derivation of any word $wh \in L\{h\}$ the symbol h is generated in the last step of this derivation and does not occur in another sentential form of this derivation. The effective construction of G_L from G'_L is obvious by using common proof techniques from the theory of formal languages and therefore omitted. Moreover, for each $(\alpha,\beta) \in P$ without loss of generality we can assume $1 \leq |\alpha| \leq 2$ and $0 \leq |\beta| \leq 2$. Now let $P_0 = P \cup \{(U,U) \mid U \in (V_N \cup V_T)\} = \{(\alpha_i,\beta_i) \mid 1 \leq i \leq m\}$ and σ_L be the CRTTS $\sigma_L = (V, M, A, 3m+5, \rho, I)$ with

$$V = V_N \cup V_T \cup \{h\} \cup \{B,F,X,Y\},$$

$$M = \left\{ [x]^+, [x]^- \mid x \in \{b,c,d,e,f,h,s,t\} \cup \{i_l, i_n, i_r \mid 1 \leq i \leq m\} \right\},$$

$$A = \{XSBY, F, XBY\} \cup \{X\beta_i Y \mid 1 \leq i \leq m\},$$

$$\rho = (\rho_1, \rho_2, \rho_3, \rho_4, \rho_5, \rho_6, \rho_7, \ldots, \rho_{3m+4}, \rho_{3m+5}),$$

$$\rho_1 = \left(\emptyset, \left\{ \left([s]^+, [e]^- \right) \right\}\right),$$

$$\rho_2 = \left(\begin{array}{l} \left\{ X\#[e]^+ \$ [e]^- \#a \mid a \in V \right\} \cup \left\{ \#[s]^+ \$ [s]^- \#F \right\}, \\ \left\{ \left([f]^+, [t]^- \right) \right\} \end{array} \right),$$

$$\rho_3 = \left(\left\{ a\#[f]^+ \$ [f]^- \#hBY \mid a \in V \right\} \cup \left\{ F\#[t]^+ \$ [t]^- \# \right\}, \emptyset \right),$$

$$\rho_4 = \left(\begin{array}{l} \left\{ w\#[b]^+ \,\$[b]^- \,\#BY \mid w \in V^2 \cup \{X\}\, V \right\} \cup \\ \left\{ XB\#\,[c]^+ \,\$\,[c]^- \,\#Y \right\}, \emptyset \end{array} \right),$$

$$\rho_5 = \left(\begin{array}{l} \left\{ X\#\,[d]^+ \,\$\,[d]^- \,\#v \mid v \in V^3 \cup V^2\,\{Y\} \cup V\,\{Y\} \right\}, \\ \left\{ \left([b]^+ , [c]^- \right), \left([c]^+ , [d]^- \right) \right\} \end{array} \right),$$

$I_1 = V^* \left\{ [s]^+ \right\} \cup \left\{ [e]^- \right\} V^*$, $I_2 = V^* \left\{ [f]^+ \right\} \cup \left\{ [t]^- \right\} V^* \cup V^* \left\{ [t]^+ \right\}$,

$I_3 = I_4 = V^+$, $I_5 = V^* \left\{ [b]^+ \right\} \cup V^* \left\{ [c]^+ \right\} \cup \left\{ [c]^- \right\} V^*$,

and for all i with $1 \leq i \leq m$,

$$\rho_{3i+3} = \left(\begin{array}{l} \left\{ w\#\,[i_r]^+ \,\$\,[i_r]^- \,\#\alpha_i Y \mid w \in \{XB\} \cup V^2 \cup \right. \\ \left. \{uB, Bu \mid u \in V\} \right\} \cup \left\{ X\beta_i\#\,[i_n]^+ \,\$\,[i_n]^- \,\#Y \right\}, \emptyset, \end{array} \right)$$

$$\rho_{3i+4} = \left(\begin{array}{l} \left\{ \begin{array}{l} X\#\,[i_l]^+ \,\$\,[i_l]^- \,\#v \mid v \in V^3 \cup V^2\,\{B\} \cup V\,\{B\}\, V \cup \\ \{B\}\, V^2 \cup V\,\{BY\} \cup \{B\}\, V\,\{Y\} \cup \{BY\} \end{array} \right\}, \\ \left\{ \left([i_r]^+ , [i_n]^- \right) \right\} \end{array} \right),$$

$$\rho_{3i+5} = \left(\emptyset, \left\{ \left([i_n]^+ , [i_l]^- \right) \right\} \right),$$

$I_{3i+3} = V^+$, $I_{3i+4} = V^* \left\{ [i_r]^+ \right\} \cup \left\{ [i_n]^- \right\} V^*$,

$I_{3i+5} = V^* \left\{ [i_n]^+ \right\} \cup \left\{ [i_l]^- \right\} V^*$.

Any sentential form w occuring in a derivation in G_L is represented by a rotated version of the form $XvBuY$, where $w = uv$, in σ_L; the symbol B marking the beginning of the word w in its rotated version in $XvBuY$ can be rotated in the test tubes T_4 and T_5 as it was already explained in the previous example. The final extraction of the terminal words in Lh is done in the test tubes T_3, T_2, and T_1 in a similar way as in the previous example. In the test tubes T_{3i+3}, T_{3i+4}, and T_{3i+5} the application of a production (α_i, β_i) is simulated in that way that α_i is eliminated at the right end and β_i is inserted at the left end, i.e. from $XvBu\alpha_i Y$ we obtain $X\beta_i vBuY$: By the cutting rules in T_{3i+3} we obtain $Xz \underset{[i_r]}{\mid} \alpha_i Y \vdash \left(Xz\,[i_r]^+ , [i_r]^- \,\alpha_i Y \right)$ and $X\beta_i \underset{[i_n]}{\mid} Y \vdash \left(X\beta_i\,[i_n]^+ , [i_n]^- \,Y \right)$, in T_{3i+4} the cutting and recombination rules yield $\left(Xz\,[i_r]^+ , [i_n]^- \,Y \right) \vdash XzY$, $X \underset{[i_l]}{\mid} zY \vdash \left(X\,[i_l]^+ , [i_l]^- \,zY \right)$, and in T_{3i+5} we obtain $\left(X\beta_i\,[i_n]^+ , [i_l]^- \,zY \right) \vdash X\beta_i zY$. By simulating the additional unit productions (U, U), every symbol $U \in (V_N \cup V_T)$ can be rotated, i.e. from

$XvBuUY$ we obtain $XUvBuY$. In this way we can rotate the sentential form $u\alpha v$ represented by a string of the form as $Xv_2Bu\alpha v_1Y$, $v = v_1v_2$, until the position where we want to apply a production (α, β) is just at the left of the symbol Y, i.e. until we have obtained $XvBu\alpha Y$. □

When using $(V, M)_2$-filters, all the test tubes T_{3i+3}, T_{3i+4} and T_{3i+5}, $1 \leq i \leq m$, as well as T_4 and T_5 constructed in the proof of Theorem 1 can be merged into only two test tubes T_2' and T_3', respectively:

Theorem 2. *For every recursively enumerable language L, $L \subseteq V_T^+$, we can construct a CRTTS σ_L' with $(V, M)_2$-filters and only three test tubes which generates L.*

Proof. Let L be given by the grammar G_L and let P_0 be defined as in the proof of Theorem 1.

We now consider the CRTTS $\sigma_L' = (V, M, A, 3, (\rho_1', \rho_2', \rho_3'), (I_1', I_2', I_3'))$, where M, A are defined as in the proof of Theorem 1 and

$$\rho_1' = \left(\emptyset, \left\{\left([s]^+, [e]^-\right), \left([f]^+, [t]^-\right)\right\}\right),$$

$$\begin{aligned}\rho_2' = \Big(&\Big\{X\#[i_l]^+\,\$\,[i_l]^-\,\#v \mid v \in V^3 \cup V^2\{B\} \cup V\{B\}V \cup \{B\}V^2 \cup \\ &V\{BY\} \cup \{B\}V\{Y\}, 1 \leq i \leq m\Big\} \cup \Big\{w\#[i_r]^+\,\$\,[i_r]^-\,\#\alpha_i Y \mid \\ &\{w \in \{B\} \cup \{Bu \mid u \in V\} \cup V^2, 1 \leq i \leq m\}\Big\} \cup \\ &\Big\{X\beta_i\#[i_n]^+\,\$\,[i_n]^-\,\#Y \mid 1 \leq i \leq m\Big\} \cup \Big\{\#[s]^+\,\$\,[s]^-\,\#F\Big\} \cup \\ &\Big\{X\#[d]^+\,\$\,[d]^-\,\#v \mid v \in V^3 \cup V^2\{B\} \cup V\{BY\}\Big\} \cup \\ &\Big\{w\#[b]^+\,\$\,[b]^-\,\#BY \mid w \in V\Big\} \cup \Big\{XB\#[c]^+\,\$\,[c]^-\,\#Y\Big\} \cup \\ &\Big\{X\#[e]^+\,\$\,[e]^-\,\#w \mid w \in V^3 \cup V^2\{h\} \cup V\{hB\}\Big\} \cup \\ &\Big\{a\#[f]^+\,\$\,[f]^-\,\#hBY \mid a \in V\Big\} \cup \Big\{F\#[t]^+\,\$\,[t]^-\,\#\Big\}, \emptyset\Big),\end{aligned}$$

$$\rho_3' = \left(\emptyset, \left\{\left([i_n]^+, [i_l]^-\right), \left([i_r]^+, [i_n]^-\right), \left([c]^+, [d]^-\right), \left([b]^+, [c]^-\right)\right\}\right);$$

$$I_1' = \left\{[e]^-\right\}V^*\left\{[f]^+\right\} \cup V^*\left\{[s]^+\right\} \cup \left\{[t]^-\right\}V^*,\ I_2' = V^+,$$

$$\begin{aligned}I_3' = &\left\{[d]^-\right\}V^*\left\{[b]^+\right\} \cup V^*\left\{[c]^+\right\} \cup \left\{[c]^-\right\}V^* \cup \\ &\bigcup_{1 \leq i \leq m}\left(\left\{[i_l]^-\right\}V^*\left\{[i_r]^+\right\} \cup V^*\left\{[i_n]^+\right\} \cup \left\{[i_n]^-\right\}V^*\right).\end{aligned}$$

By the cutting rules in T_2' we obtain $X \underset{[i_l]}{|} uBv\alpha_i Y \vdash \left(X[i_l]^+, [i_l]^- uBv\alpha_i Y\right)$ and $[i_l]^- uBv \underset{[i_r]}{|} \alpha_i Y \vdash \left([i_l]^- uBv[i_r]^+, [i_r]^-\alpha_i Y\right)$, as well as in addition

$X\beta_i \underset{[i_n]}{\mid} Y \vdash \left(X\beta_i\,[i_n]^+\,,[i_n]^-\,Y\right)$; by the recombination rules in T_3' we get $\left(X\beta_i\,[i_n]^+\,,[i_l]^-\,uBv\,[i_r]^+\right) \vdash X\beta_i uBv\,[i_r]^+$ and $\left(X\beta_i uBv\,[i_r]^+\,,[i_n]^-\,Y\right) \vdash X\beta_i uBvY$, i.e. in sum from $XuBv\alpha_iY$ we derive $X\beta_iuBvY$ thus simulating the application of the production (α_i,β_i). Rotating the symbol B works in a similar way, thus yielding $XBwY$ from $XwBY$, and terminal strings $w \in V_T^+$ are obtained in T_1' by passing the objects $[e]^-\,w\,[f]^+$ as well as $[s]^+$ and $[t]^-$ from T_2' to T_1'. The use of the $(V,M)_2$-filters $\left\{[e]^-\right\} V^* \left\{[f]^+\right\}$ in I_1' as well as of the $(V,M)_2$-filters $\left\{[d]^-\right\} V^* \left\{[b]^+\right\}$ and $\left\{[i_l]^-\right\} V^* \left\{[i_r]^+\right\}$, $1 \leq i \leq m$, respectively, which are not $(V,M)_1$ -filters, guarantees that only objects with corresponding markings on the left-hand side and on the right-hand side can pass from T_2' to T_1' and T_3', whereas objects with markings that do not fit together remain in T_2'. □

The existence of universal grammar schemes now implies the existence of universal CRTTS with $(V,M)_i$-filters, $i \in \{1,2\}$:

Corollary 1. *For every alphabet V_T and every universal grammar scheme γ_U for V_T, $\gamma_U = (V_U, V_T, P_U)$, we can effectively construct a universal CRTTS $\sigma_{U,i}$, $\sigma_{U,i} = (V_i, M_i, A_i, n_i, \rho_i, I_i)$, with $(V,M)_i$-filters, $i \in \{1,2\}$, such that any $L \in ENUM(V_T)$ is generated by the CRTTS $\sigma_{L,i}$ with $(V,M)_i$-filters with $\sigma_{L,i} = (V_i, M_i, A_i \cup \{XA_LBY\}, n_i, \rho_i, I_i)$, where A_L is the axiom for which the grammar $G_L = (V_N, V_T, P, A_L)$ generates L; in the case $i = 2$, $n_2 = 3$, i.e. ρ_2 consists of only three components.*

Proof. We just have to apply the constructions in the proofs of Theorem 1 and Theorem 2, respectively, to the universal grammar scheme γ_U for V_T with the only exception that we do not take $XSBY$ as an axiom in A. Instead of this, in every special CRTTS $\sigma_{L,i}$ with $(V,M)_i$-filters the starting axiom A_L is taken. □

4 Summary and Future Research

The construction of molecular computers based on test tubes has been considered by using different operations on the test tubes [1,2,3,9]. Test tube systems based on the splicing operation were shown to allow the construction of universal mechanisms [4]. In the preceding section we have shown the (theoretical) possibility how to obtain universal biological molecular computers based on test tube systems with cutting and recombination rules. Our definitions can easily be extended in order to cover a large variety of such test tube sytems [6]; hence our results should also be true for various possible practical implementations of

such systems. On the other hand, we only provide a kind of programming language for molecular computers; feasible solutions for specific problems should take advantage of the specific features of CRTTS without only relying on the general methods used in the proofs of the results given in this paper.

Acknowledgements

The first two authors want to gratefully thank Gheorghe Păun for days (and nights) of intensive discussions on the topics related with splicing and test tube systems and they also appreciate fruitful discussions with Lila Kari on some of the topics considered in this paper. The work of the second author was supported by Hungarian Scientific Research Fund OTKA No. T017105.

References

1. L. M. Adleman, Molecular computation of solutions to combinatorial problems, *Science* **226**, 1021 – 1024 (1994).
2. L. M. Adleman, On constructing a molecular computer, *manuscript* (1995).
3. D. Boneh, C. Dunworth, R.J. Lipton, and J. Sgall, On the computational power of DNA, *to appear* (1996).
4. E. Csuhaj-Varjú, L. Kari, and Gh. Păun, Test tube distributed systems based on splicing, *Computers and Artificial Intelligence,* Vol. **15** (2), 211-232 (1996).
5. R. Freund, L. Kari, and Gh. Păun, DNA computing based on splicing: The existence of universal computers, Techn. Report 185-2/FR-2/95, TU Wien, Institute for Computer Languages (1995).
6. R. Freund and F. Freund, Test tube systems or How to bake a DNA cake, Proc. Workshop Grammar Systems: Recent Results and Perspectives, *to appear* (1996).
7. R. Freund and F. Wachtler, Universal systems with operations related to splicing, *Computers and Artificial Intelligence,* Vol. **15** (4), 273-294 (1996).
8. T. Head, Formal language theory and DNA: An analysis of the generative capacity of specific recombinant behaviors, *Bull. Math. Biology* **49**, 737 – 759 (1987).
9. R. J. Lipton, Speeding up computations via molecular biology, *manuscript (*1994).
10. Gh. Păun, Regular extended H systems are computationally universal, *J. Automata, Languages and Combinatorics*, Vol. 1 (1), 27 - 36 (1996).