Implementation of Haskell modules for automata and Sticker systems

Perera, K.K.K.R.
Graduate School of Mathematics, Kyushu University

Mizoguchi, Yoshihiro
Faculty of Mathematics, Kyushu University

出版情報：Journal of Math-for-Industry (JMI). 1 (A), pp.51-56, 2009-04-08. 九州大学大学院数理学研究院

Journal of Math-for-industry, Vol.1(2009A-7), pp.51–56

Implementation of Haskell modules for automata and Sticker systems

K.K.K.R.Perera and Yoshihiro Mizoguchi

Received on March 11, 2009 / Revised on March 17, 2009

Abstract.
We realized operations appeared in the theory of automata using Haskell languages. Using the benefits of functions of lazy evaluations in Haskell, we can express a language set which contains infinite elements as concrete functional notations like mathematical notations. Our modules can be used not only for analyzing the properties about automata and their application systems but also for self study materials or a tutorial to learn automata, grammar and language theories. We also implemented the modules for sticker systems. Paun and Rozenberg explained a concrete method to transform an automaton to a sticker system in 1998. We modified their definitions and improved their insufficient results. Using our module functions, we can easily define finite automata and linear grammars and construct sticker systems which have the same power of finite automata and linear grammars.

Keywords. Automata, Language, Sticker System, DNA Computing, Haskell

1. Introduction

The sticker system is a formal model based on sticking operations, which is an abstraction of the Watson-Crick complementarity. We use the term domino to represent double stranded DNA sequences with sticky ends. By using the sticking operator, dominoes can be annealed and formed a complete double stranded sequence. Paun and Rozenberg [3] explained a concrete method to transform automata to sticker systems. In this paper we are trying to introduce simple efficient transformation and implement it using Haskell module functions. We also indicate and improve the insufficient results in [3]. We modify the expression of dominoes and the sticking operator for realizing Haskell functions. We change the definition of a domino (D) from a string of pairs of alphabet to a triple (l, r, x) of two string l, r and an integer x. For example $\begin{pmatrix} \lambda \\ C \end{pmatrix} \begin{bmatrix} AT \\ TA \end{bmatrix} \begin{pmatrix} GC \\ \lambda \end{pmatrix}$ in [3] is represented as $(ATGC, CTA, -1)$. According to this modification, the definition of sticking operator has been reformulated.

One of the benefits of using Haskell language is that it has descriptions for infinite set of strings using lazy evaluation schemes. For example, the infinite set $\{a, b\}^*$ is denoted by finite length of expression `sstar ['a','b']`. We use the `take` function to view contents of an infinite set (e.g. `take 5 (sstar ['a','b'])` is `["","a","b","aa","ba"]`. Further using set theoretical notions in Haskell, we can easily realize the definitions of various kinds of set of dominoes. For example, to make a sticker system which generates the equivalent language of a finite automaton, we need an atom set

$$A_2 = \bigcup_{i=1}^{k+1} \{(xu, x, 0) | \quad x \in \Sigma^*, u \in \Sigma^*, |xu| = k+2, \quad |u| = i, \delta^*(0, xu) = i-1\}.$$

In Haskell notations, we have following function definitions.

```
aA2::Automaton->[Domino]
aA2 m@(q,s,d,q0,f) = concat [(aA2' m i)| i<- [1..(k+1)]]
                       where k = (length q)-1
aA2'::Automaton->Int->[Domino]
aA2' m@(q,s,d,q0,f) i =  [(x++u,x,0)| (x,u) <- xupair,
                          (dstar d 0 (x++u)) ==(i-1)]
where xupair = [(x,u)|x<-(sigman s (k+2-i)),
                      u<-(sigman s i)]
         k = (length q)-1
```

The precise definition of the generated sticker system is described in Section 3. We prove that the generated languages are equal by using our formulations.

The Haskell module can be downloaded from our homepage[1].

2. Automaton Module

Let Σ is an alphabet and Σ^* is the set of all strings over Σ including the empty string λ. For a string w, we denote the length of w by $|w|$.

[1]`http://haskell.math.kyushu-u.ac.jp/`

Definition 1. A finite automaton is a five-tuple of $M = (Q, \Sigma, \delta, q_0, F)$, where Q is the finite set of states, Σ is the alphabet, q_0 is the initial state, F is the set of final states and $\delta : Q \times \Sigma \to Q$ is the transition function.

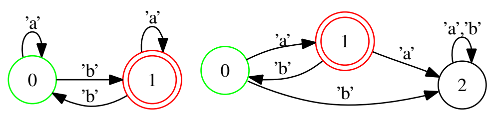
A transition function $\delta : Q \times \Sigma \to Q$ is generally extended to a function $\delta^* : Q \times \Sigma^* \to Q$ by $\delta^*(q, \lambda) = q$ and $\delta^*(q, xw) = \delta^*(\delta(q, x), w)$ for $q \in Q$, $x \in \Sigma$ and $w \in \Sigma^*$.

Definition 2. For a finite automaton $\mathtt{M}{=}(Q, \Sigma, \delta, q_0, F)$, we define the language $L(M)$ accepted by M by $L(M) = \{w \in \Sigma^* | \delta^*(q_0, w) \in Q_F\}$.

Example 1. Automata M_1 and M_2 is defined as follows. $M_1 = (\{0, 1\}, \{a, b\}, \delta_1, 0, \{1\})$ and $M_2 = (\{0, 1, 2\}, \{a, b\}, \delta_2, 0, \{1\})$, where $\delta_1(0, a) = 0$, $\delta_1(0, b) = 1$, $\delta_1(1, a) = 1$, $\delta_1(1, b) = 0$, $\delta_2(0, a) = 1$, $\delta_2(0, b) = 2$, $\delta_2(1, a) = 2$, $\delta_2(1, b) = 0$, $\delta_2(2, a) = 2$, $\delta_2(2, b) = 2$. Figure 1 is the transition diagram for M_1 and M_2. The examples are expressed as follows using our Haskell Modules.

```
m1::Automaton
m1 = ([0,1], ['a','b'], d1, 0, [1])
    where d1 0 'a' = 0
          d1 0 'b' = 1
          d1 1 'a' = 1
          d1 1 'b' = 0

m2::Automaton
m2 = ([0,1,2], ['a','b'], d2, 0, [1])
    where d2 0 'a' = 1
          d2 0 'b' = 2
          d2 1 'a' = 2
          d2 1 'b' = 0
          d2 2 'a' = 2
          d2 2 'b' = 2
```



(a) M_1 (b) M_2

Figure 1: Example of finite automata

We note that $L(M_1) = \{w \in \Sigma^* \,|\, |w|_b = 1 (\bmod\ 2)\}$, and $L(M_2) = \{a(ba)^n \,|\, n = 0, 1, ...\}$. In our module the function `Automaton.language` returns the accepted language. To compute the accepted language generated by M_1, we use `Automaton.language m1`, where `m1` is the automaton described in Haskell.

Following is a code for finding accepted language and their executions.

```
accepts::Automaton->[String]->[String]
accepts m ss = [w | w <- ss, (dstar d s0 w) `elem` f]
                 where (q, s, d, s0, f)=m

language::Automaton ->[String]
language m = accepts m (sstar s)
               where (q, s, d, s0, f)=m
```

```
*AutomatonEx>take 10 $ Automaton.language m1
["b","ba","ab","baa","aba","aab","bbb"
,"baaa","abaa","aaba"]
*AutomatonEx>take 5 $ Automaton.language m2
["a","aba","ababa","abababa","ababababa"]
```

3. Sticker Module

Definition 3. Let Σ be a set of alphabet and Z a set of integers and $\rho \subseteq \Sigma \times \Sigma$. An element (l, r, n) of $\Sigma^* \times \Sigma^* \times \mathbf{Z}$ is a **domino** over (Σ, ρ), if the following conditions holds:

- If $n \geq 0$ then $(l[i + n], r[i]) \in \rho$, for $1 \leq i \leq min(|l| - n, |r|)$
- If $n < 0$ then $(l[i], r[i - n]) \in \rho$, for $1 \leq i \leq min(|r| + n, |l|)$

We denote the set of all dominoes over (Σ, ρ) by D.

The possible shapes of the dominoes are illustrated as follows: $\boxed{\begin{array}{ll} \text{x} & \text{u} \\ \text{x'} & \end{array}}$, $\boxed{\begin{array}{l} \text{x} \\ \text{x'} \end{array}}$, $\boxed{\begin{array}{ll} & \text{x} \\ \text{v} & \text{x'} \end{array}}$, $\boxed{\begin{array}{lll} & \text{x} & \text{u} \\ \text{v} & \text{x'} & \end{array}}$, where $x = x_1x_2 \cdots x_n$, $x' = x'_1x'_2 \cdots x'_n$, u, v, $\in \Sigma^*$ and $(x_i, x'_i) \in \rho$ for $1 \leq i \leq n$. Sticky ends can be placed in the upper strand or lower strand.

Example 2. We can represent a double stranded sequence $\begin{pmatrix} \lambda \\ C \end{pmatrix} \begin{bmatrix} AT \\ TA \end{bmatrix} \begin{pmatrix} GC \\ CG \end{pmatrix}$ in [3] by $(ATGC, CTACG, -1)$ in our module. Similarly, $\begin{pmatrix} G \\ \lambda \end{pmatrix} \begin{bmatrix} AT \\ TA \end{bmatrix}$ can be represented by $(GAT, TA, 1)$.

Definition 4. The sticking operator $\mu : D \times D \to D \cup \{\perp\}$ is defined as follows:

$$\mu((l_1, r_1, n_1), (l_2, r_2, n_2)) = \begin{cases} (l_1l_2, r_1r_2, n_1) & (\text{if } (*)) \\ \perp \quad (\text{otherwise}) \end{cases}$$

(*) $(l_1l_2, r_1r_2, n_1) \in D$ and $n_1 + |r_1| - |l_1| = n_2$

Definition 5. Sticker System γ is a four tuple $\gamma = (\Sigma, \rho, A, R)$ of an alphabet set Σ, $\rho \subseteq \Sigma \times \Sigma$, a finite set of axioms $A (\subseteq D)$ and a finite set of pairs of dominoes $R \subseteq D \times D$.

Let $Q = \{q_0, q_1, ..., q_k\}$ be a finite set, which consists of $k{+}1$ elements. For a finite automaton $M = (Q, \Sigma, \delta, q_0, F_M)$, the sticker system $\gamma_M = (\Sigma, \rho, A, R)$ is defined as follows:

$$\begin{aligned} \rho &= \{(a, a) | a \in \Sigma\} \\ A &= A_1 \cup A_2 \\ A_1 &= \{(x, x, 0) \,|\, x \in L(M), |x| \leq k + 2\} \end{aligned}$$

$$
\begin{aligned}
A_2 &= \{(xu, x, 0) \mid |xu| = k+2,\ |u| = i, \\
&\qquad \delta^*(q_0, xu) = q_{i-1}, 1 \le i \le k+1\} \\
R &= R_1 \cup F \\
R_1 &= \{((\lambda, \lambda, 0), (xu, vx, -|v|)) \mid \\
&\qquad |xu| = k+2,\ |u| = i,\ |v| = j,\ \delta^*(q_{j-1}, xu) = q_{i-1}, \\
&\qquad 1 \le i \le k+1,\ 1 \le j \le k+1\} \\
F &= \{((\lambda, \lambda, 0), (x, vx, -|v|)) \\
&\qquad \mid |v| = i,\ |x| = j,\ \delta(q_{i-1}, x) \in F_M, \\
&\qquad 1 \le i \le k+1, 1 \le j \le k+2\}
\end{aligned}
$$

For a sticker system $\gamma = (\Sigma, \rho, A, R)$, we define a relation $\Rightarrow_\gamma$ on D as follows.

$$x \Rightarrow_\gamma y \ \stackrel{def}{\Longleftrightarrow}\ y = \mu(\alpha, \mu(x, \beta)) \text{ for some } (\alpha, \beta) \in R,$$

Let $\Rightarrow_\gamma^*$ be the reflective and transitive closure of $\Rightarrow_\gamma$.

Definition 6. The set of dominoes $LM(\gamma)$ generated by γ is defined by $LM(\gamma) = \{(l, r, 0) | a \Rightarrow^* (l, r, 0), a \in A, |l| = |r|\}$. The language $L(\gamma)$ generated by γ is defined by $L(\gamma) = \{l \in \Sigma^* | (l, r, 0) \in LM(\gamma)\}$.

Example 3. Consider the sticker system γ_{M_1} generated by the automaton M_1 in Example 1. Since $\delta_1^*(0, bbb) = 1$ then the domino $(bbb, b, 0) \in A$. Also we have $((\lambda, \lambda, 0), (bab, bbbab, -2)) \in F$ by $\delta^*(1, bab) \in F_{M_1}$. The domino $(bbb, b, 0)$ is figured as $\boxed{\begin{array}{ccc} \mathrm{b} & \mathrm{b} & \mathrm{b} \\ \mathrm{b} & & \end{array}}$ and $(bab, bbbab, -2)$ is figured as $\boxed{\begin{array}{ccccc} & & \mathrm{b} & \mathrm{a} & \mathrm{b} \\ \mathrm{b} & \mathrm{b} & \mathrm{b} & \mathrm{a} & \mathrm{b} \end{array}}$.

$$\mu((bbb, b, 0), (bab, bbbab, -2)) \quad = \quad (bbbbab, bbbbab, 0))$$

Since $(bbb, b, 0) \in A$ and $(bbb, b, 0) \Rightarrow_\gamma^* (bbbbab, bbbbab, 0)$, we have $bbbbab \in L(\gamma_{M_1})$.

For $i = 1, ..., k+1$, we define X_i, Y_i and F_i as follows:

$$
\begin{aligned}
X_i &= \{(xu, x, 0) \in A \mid |xu| = k+2,\ |u| = i,\ u, x \in \Sigma^*\}, \\
Y_i &= \{(xu, x, 0) \mid a \Rightarrow_\gamma^* (xu, x, 0),\ a \in A,\ |u| = i, \\
&\qquad\qquad u, x \in \Sigma^*\}, \\
Z_i &= \{((\lambda, \lambda, 0), (x, vx, -i)) \in F \mid |v| = i,\ v, x \in \Sigma^*\}
\end{aligned}
$$

Lemma 1. *Define the sticker system $\gamma = \gamma_M$ for a finite automaton $M = (Q, \Sigma, \delta, q_0, F_M)$. For $i = 1, 2, ..., k+1$ the followings hold.*

(i) *For $a \in A$, If $a \Rightarrow_\gamma^* (l, r, n)$, then $n = 0$ and $|r| \le |l| \le |r| + k + 1$.*

(ii) *If $(l, r, 0) \in LM(\gamma)$, then $(l, r, 0) \in A$ or there exist $x, u, x' \in \Sigma^*$ such that $|u| = i$, $1 \le |x|$, $1 \le |x'|$, $l = x'ux$ and $((\lambda, \lambda, 0), (x', ux', -i)) \in F_i$. i.e. $\mu((xu, x, 0), (x', ux', -i)) = (l, r, 0)$ and $(xu, x, 0) \in Y_i$.*

(iii) *$X_i = \{(xu, x, 0) \mid |u| = i,\ |xu| = k+2,\ u, x \in \Sigma^*,\ \delta^*(q_0, xu) = q_{i-1}\}$.*

(iv) *If $(xu, x, 0) \in Y_i$ and $|xu| \le k+2$ then $(xu, x, 0) \in X_i$.*

(v) *If $(xu, x, 0) \in Y_i$ and $|xu| > k+2$, then there exist $x'', u', x' \in \Sigma^*$ such that $|x'u| = k+2$, $1 \le |u'| \le k+1$, $x''u'x' = x$ and $((\lambda, \lambda, 0),\ (x'u, u'x, -|u'|)) \in R_1$. i.e. $\mu((x''u', x'', 0), (x'u, u'x', -|u'|)) = (xu, x, 0)$ and $(x''u',\ x'',\ 0) \in Y_{|u'|}$.*
(cf. $\boxed{\begin{array}{cccc} x'' & u' & x' & u \\ x'' & u' & x' & \end{array}} = \boxed{\begin{array}{cc} x & u \\ x & \end{array}}$ *)*

(vi) $Y_i = \{(xu, x, 0) \mid |u| = i,\ \delta^*(q_0, xu) = q_{i-1},\ (k+2) \mid |xu|,\ u, x \in \Sigma^*\}$.

(vii) $F = \displaystyle\bigcup_{i=1}^{k+1} Z_i$

(Proof) **(i)**,**(iii)**,**(iv)** and **(vii)** are trivial.
(ii) Let $(l, r, 0)$ be a domino and $((\lambda, \lambda, 0), (xu, vx, -|v|)) \in R_1$. If $\mu((l, r, 0), (xu, vx, -|v|)) \ne \perp$ then $\mu((l, r, 0), (xu, vx, -i)) = (lxu, rvx, 0)$ and $0 + |r| - |l| = -|v|$. $|lxu| - |rvx| = |l| + |x| + |u| - |r| - |v| - |x| = |u| \ne 0$. So $\mu((l, r, 0), (xu, vx, -i)) \notin LM(\gamma)$.
Let $(xu, x, 0)$ be a domino with $x, u \in \Sigma^*$, $1 \le x$ and $1 \le |u| \le k+1$. If $\mu((xu, x, 0), (l', r', n')) \ne \perp$ and $(xul', xr', 0) \in LM(\gamma)$, then (l', r', n') is $(x', ux', -|u|)$ for some $x' \in \Sigma^*$ and $1 \le x'$. So there exist $((\lambda, \lambda, 0),\ (x', ux', -|u|)) \in F$ and $a \Rightarrow_\gamma^* (xu, x, 0) \Rightarrow_\gamma (l, r, 0)$.
(v) Since $|xu| > k+2$, there exists a domino $(x''u', x'', 0)$ such that $a \Rightarrow_\gamma^* (x''u', x'', 0) \Rightarrow_\gamma (xu, u, 0)$. This means there exists $((\lambda, \lambda, 0),\ (x'u, u'x', -|u'|)) \in R_1$ such that $\mu((x''u', x'', 0),\ (x'u, u'x', -|u'|)) = (x''u'x'u, x''u'x', 0) = (xu, x, 0)$. So we have $x = x''u'x'$, $|x'u| = k+2$ and $1 \le |u'| \le k+1$.
(vi) ($\subset$) Let $(xu, x, 0) \in Y_i$. If $|xu| \le k+2$ then $\delta^*(q_0, xu) = q_{i-1}$ by (iii) and (iv). If $|xu| > k+2$ then there exists a domino $(x''u', x'', 0) \in Y_{|u'|}$ and $((\lambda, \lambda, 0), (x'u, u'x', -|u'|)) \in R_1$ such that $x = x''u'x$ by (v). Since $(x''u', x'', 0) \in Y_{|u'|}$ we have $\delta^*(q_0, x''u') = q_{|u'|-1}$. Since $((\lambda, \lambda, 0),\ (x'u,\ u'x',\ -|u'|)) \in R_1$, we have $\delta^*(q_{|u'|-1},\ x'u) = q_{|u|-1}$. So we have $\delta^*(q_0, xu) = \delta^*(q_0, x''u'x'u) = q_{|u|-1} = q_{i-1}$.
($\supset$) Let $(xu, x, 0)$ be an element of the right-hand set. That is $\delta^*(q_0, xu) = q_{i-1}$, $|u| = i$ and $(k+2) \mid |xu|$. We prove $(xu, x, 0) \in Y_i$ using induction on n where $|xu| = n(k+2)$. If $|xu| = k+2$ then $(xu, x, 0) \in X_i$ by (iii), $(xu, x, 0) \in A$ and we have $(xu, x, 0) \in Y_i$.
Assume $(xu, x, 0) \in Y_i$ for any $xu \in \Sigma^*$ with $|xu| = n(k+2)$. Let $(xu, x, 0)$ be a domino and $|xu| = (n+1)(k+2)$. We put $x = x'u'x''$ where $|x''u| = k+2$, $1 \le |u'| \le k+1$ and $\delta^*(q_0, x'u') = q_{|u'|-1}$. Since $|x'u'| = |xu| - |x''u| = n(k+2)$, we have $(x'u', x', 0) \in Y_i$ by the hypothesis of the induction. Since $\delta^*(q_{|u'|-1}, x''u) = \delta^*(\delta^*(q_0, x'u'), x''u) = \delta^*(q_0, x'u'x''u) = q_{i-1}$, we have $((\lambda, \lambda, 0),\ (x''u, u'x'',\ -|u'|)) \in R_1$. Since $\mu((x'u', x', 0),\ (x''u, u'x'',\ -|u'|)) = (x'u'x''u,\ x'u'x'',\ 0) = (xu, x, 0)$, we have $(x'u', x', 0) \Rightarrow_\gamma (xu, x, 0)$ and $(xu, x, 0) \in Y_i$. $\square$

The idea of the proof of the next theorem is originally introduced by Paun and Rozenberg([3]) in 1998. It lacked several conditions and formal proofs in their paper. We modified and improved them and proved it using our formulations.

Theorem 1. *Define the sticker system $\gamma = \gamma_M$ for a finite automaton $M = (Q, \Sigma, \delta, q_0, F_M)$. Then $L(\gamma) = L(M)$.*

(Proof) ($\subset$) Let $w \in L(\gamma_M)$. Then we have $a \Rightarrow^*_\gamma (w, w, 0)$ for some $a \in A$. If $(w, w, 0) \in A$ then $w \in L(M)$ by definition. If $(w, w, 0) \notin A$ then there exist $(xu, x, 0)$ and $((\lambda, \lambda, 0), (x', ux', -|u|)) \in F$ such that $a \Rightarrow^*_\gamma (xu, x, 0)$ and $\mu((xu, x, 0), (x', ux', -|u|)) = (w, w, 0)$.
Since $a \Rightarrow^*_\gamma (xu, x, 0)$, we have $\delta^*(q_0, xu) = q_{|u|-1}$ from Lemma 1(vi). Since $((\lambda, \lambda, 0), (x', ux', -|u|)) \in F$, we have $\delta^*(q_{|u|-1}, x') \in F_M$. Since $\delta^*(q_0, w) = \delta^*(q_0, xux')$ $= \delta^*(q_{|u|-1}, x') \in F_M$, we have $w \in L(M)$.
($\supset$) Let $w \in L(M)$. If $|w| \le k + 2$ then $(w, w, 0) \in A$ and $w \in L(\gamma_M)$.
If $k+2 \le |w| \le 2(k+2)$ then we put $w = w'x'$ where $|w'| = k + 2$. If $\delta^*(q_0, w') = q_{i-1}$ then $(w''u, w'', 0) \in A$ where $w' = w''u$ and $|u| = i$. Since $\delta^*(q_{i-1}, x') = \delta^*(\delta^*(q_0, w'), x')$ $= \delta^*(q_0, w) \in F_M$, we have $((\lambda, \lambda, 0), (x', ux', -i)) \in F$. Since $\mu((w''u, w'', 0), (x', ux', -1)) = (w''ux', w''ux', 0) = (w, w, 0)$, we have $(w''u, w'', 0) \Rightarrow_\gamma (w, w, 0)$ and $w \in L(\gamma_M)$.
If $|w| > 2(k + 2)$, let $w = w'x'$ where $(k + 2) \mid |w'|$ and $|x'| \le k + 2$. If $\delta^*(q_0, w') = q_{i-1}$ then $(w''u, w'', 0) \in Y_i$ where $w' = w''u$ and $|u| = i$ by Lemma 1(vi). Since $\delta^*(q_{i-1}, x') = \delta^*(\delta^*(q_0, w'), x') = \delta(q_0, w) \in F_M$, we have $((\lambda, \lambda, 0), (x', ux', -i)) \in F$. Since $\mu((w''u, w'', 0),$ $(x', ux', -i)) = (w''ux', w''ux', 0) = (w, w, 0)$, we have $(w, w, 0) \in L(\gamma_M)$. $\square$

Note: We correct the limit length of x in F from k to $k + 2$ in [3]. Consider the sticker system γ_{M_1} generated by the automaton M_1 in Example 1 again. Since $(abb, ab,$ $0) \in A$, $((\lambda, \lambda, 0), (aba, baba, -1)) \in F$ and $\mu((abb, ab, 0),$ $(aba, baba, -1)) = (abbaba, abbaba, 0)$, we have $abbaba \in L(\gamma_{M_1})$. In the definition of F in [3], the limit of length $|x|$ for $(x, vx, -|v|) \in F$ is $k = 1$. Since $|aba| > 1$, we do not have $((\lambda, \lambda, 0), (aba, baba, -1))$ in F by the definition in [3]. So even $abbaba \in L(M_1)$, $abbaba \notin L(\gamma_{M_1})$ according to the definition of sticker system described in [3].

4. Grammar Module

Definition 7. A grammar is a four tuple $G = (T, N, R, S)$ of terminal symbols T, non-terminal symbols N, transformation rules R and a start symbol S.

Definition 8. The language $L(G)$ generated by grammar $G = (\Sigma, N, R, S)$ is defined as $L(G) = \{w \in \Sigma^* | S \Rightarrow^*_G w\}$. For a grammar `g`$= G$, `(Grammar.language g)` computes the $L(G)$.

Example 4. The grammars $G_1 = (\{a, b\}, \{S\}, \{S \to aSb,$ $S \to ab\}, S)$ and $G_2 = (\{a, b\}, \{S, A\}, \{S \to A, S \to aSb,$ $A \to aA, A \to a\}, S)$ are expressed as follows using our Haskell Modules.

```
gex1::Grammar
gex1=(['a','b'],['S'],[('S',"aSb"),('S',"ab")],'S')
gex2::Grammar
gex2=(['a','b'],['S','A'],[('S',"A"),('S',"aSb"),
('A',"aA"),('A',"a")],'S')
```

```
*GrammarExChar> gex1
("ab","S",[('S',"aSb"),('S',"ab")],'S')
*GrammarExChar>take 10 $ Grammar.language gex1
["ab","aabb","aaabbb","aaaabbbb",
"aaaaabbbbb","aaaaaabbbbbb",
"aaaaaaabbbbbbb","aaaaaaaabbbbbbbb",
"aaaaaaaaabbbbbbbbb","aaaaaaaaaabbbbbbbbbb"]
*GrammarExChar> gex2
("ab","SA",[('S',"A"),('S',"aSb"),('A',"aA"),
('A',"a")],'S')
*GrammarExChar> take 10 $ Grammar.language
 GrammarEx.gex2
["a","aa","aab","aaabb","aaa",
"aaaabbb","aaaa","aaab","aaaab","aaaaabbb"]
```

For a string $w = x_1 x_2 \cdots x_n$ and $1 \le i \le n$, $Left(w, i) = x_1 \cdots x_i$ and $Right(w, i) = x_{n-i+1} \cdots x_n$.

Definition 9. Let $N = \{X_1, X_2, \cdots, X_k\}$ be a finite set of k non-terminal symbols and $S = X_1$. For a linear grammar $G = (\Sigma, N, P, S)$, the sticker system $\gamma_G = (\sigma, \rho, A, R)$ is defined similar to [3] as follows.

$$
\begin{aligned}
\rho &= \{(a, a) \mid a \in \Sigma\} \\
X_1 &= S\,(\textit{if } i = 1 \textit{ then } X_i = S) \\
T(i, k) &= \{w \in \Sigma^* \mid X_i \Rightarrow^* w, |w| = k\} \\
T(i, l, r) &= \{(w_l, j, w_r) \in (\Sigma^* \times \boldsymbol{N} \times \Sigma^*) \mid \\
&\qquad X_i \Rightarrow w_l X_j w_r, |w_l| = l, |w_r| = r\} \\
A &= A_1 \cup A_2 \cup A_3 \\
A_1 &= \{(x, x, 0) | x \in T(1, m), m \le 3k + 2\} \\
A_2 &= \bigcup_{i=1}^{k} \{(ux, x, |u|) \mid \\
&\qquad w \in T(i, m),\ i + 1 \le m \le 3k + 2, \\
&\qquad x = Right(w, m - i),\ u = Left(w, i)\} \\
A_3 &= \bigcup_{i=1}^{k} \{(xu, x, 0) \mid \\
&\qquad w \in T(i, m),\ i + 1 \le m \le 3k + 2, \\
&\qquad x = Left(w, m - i),\ u = Right(w, i)\} \\
R &= R_1 \cup R_2 \cup R_3 \cup R_4 \cup R_5 \cup R_6 \\
R_1 &= \bigcup_{i=1}^{k} \bigcup_{l=0}^{k+1} \{((ux, xv, |u|), (z, z, 0)) | \\
&\qquad (w, j, z) \in T(i, k + 1, l),\ u = Left(w, i), \\
&\qquad x = Right(w, i),\ |v| = j\} \\
R_2 &= \bigcup_{i=1}^{k} \bigcup_{l=0}^{k+1} \{((x, xv, 0), (zu, z, 0)) | (x, j, w) \\
&\qquad \in T(i, l, k + 1),\ z = Left(w, k + 1 - i), \\
&\qquad u = Right(w, i),\ |v| = j\}
\end{aligned}
$$

$$
\begin{aligned}
R_3 &= \bigcup_{l=1}^{2k+2} \{((x, xv, 0), (z, z, 0)) \mid (x, j, z) \in T(0, l, m), \\
&\quad 0 \le m \le 2k+2-l,\ |v| = j\} \\
R_4 &= \bigcup_{i=1}^{k} \bigcup_{l=0}^{k+1} \{((z, z, 0), (xu, vx, -|v|)) \mid (z, j, w) \\
&\quad \in T(i, l, k+1),\ x = Left(w, k+1-i), \\
&\quad u = Right(w, i),\ |v| = j\} \\
R_5 &= \bigcup_{i=1}^{k} \bigcup_{l=0}^{k+1} \{((uz, z, |u|), (x, vx, -|v|)) \mid (w, j, z) \\
&\quad \in T(i, k+1, l), u = Left(w, i), \\
&\quad x = Right(w, k+1-i),\ |v| = j\} \\
R_6 &= \bigcup_{l=1}^{2k+2} \{((z, z, 0), (x, vx, -|v|)) \mid (z, j, x) \\
&\quad \in T(1, m, l),\ 0 \le m \le 2k+2-l, |v| = j\} \\
k &= |N|
\end{aligned}
$$

We modified the limitation of the production rules ([3]) in G to allow the form $X \to xYy$ for $|x| = |y| = 1$. To prove the next generalized theorem, we change the limit length of w in A from $3k+1$ to $3k+2$, the length of z in R_1, R_2, R_4 and R_5 from k to $k+1$, and the length of z in R_3 and R_6 from $2k+1$ to $2k+2$.

Theorem 2 ([3]). *Define the sticker system $\gamma = \gamma_G$ for a linear grammar $G = (\Sigma, N, P, S)$. If a linear grammar G has only production rules of the forms $X \to xYy$ and $X \to x$ for $X, Y \in N$, $x, y \in T^*$, $1 \le |xy|$, $|x| \le 1$ and $|y| \le 1$, then $L(\gamma_G) = L(G)$.*

(Proof)

We define a set Y_i for $i = 1, \cdots, k$ as follows.

$$
\begin{aligned}
Y_i &= \{xu \in \Sigma^* \mid a \Rightarrow_\gamma^* (xu, x, 0),\ a \in A,\ |u| = i\} \\
&\quad \cup \{ux \in \Sigma^* \mid a \Rightarrow_\gamma^* (x, ux, -|u|),\ a \in A,\ |u| = i\}
\end{aligned}
$$

It is easy to show that $Y_i \subset \{w \mid X_i \Rightarrow_G^* w,\ |w| \ge k+1\}$ and $L(\gamma_G) \subset L(G)$. We prove $Y_i \supset \{w \mid X_i \Rightarrow_G^* w,\ |w| \ge k+1\}$ using induction on the length of $|w|$. Assume $X_i \Rightarrow_G^* w$ and $|w| \ge k+1$. If $|w| \le 3k+2$ then there exist x and u satisfying $w = xu$ and $|u| = i$ such that $(xu, x, 0) \in A_3$. So we have $w \in Y_i$. We assume $X_i \Rightarrow_G^* w$ and $w' \in Y_j$ for any w' and j satisfying $X_j \Rightarrow_G^* w'$ and $|w'| < |w|$. According to the limitation of production rules in G, we have $X_i \Rightarrow_G x_1 X_{i_1} y_1 \Rightarrow_G x_1 x_2 X_{i_2} y_2 y_1 \Rightarrow_G^* x_1 x_2 \cdots x_n y_n \cdots y_2 y_1 = w$ for $x_p, y_p \in T^*$, $|x_p| \le 1$, $|y_p| \le 1$ and $1 \le |x_p y_p|$ $(p = 1, \cdots, n)$. If $|w| > 3k+2$ then there exist m and X_j such that $((|x_1 x_2 \cdots x_m| = k+1$ and $|y_1 y_2 \cdots y_m| \le k+1)$ or $(|x_1 x_2 \cdots x_m| \le k+1$ and $|y_1 y_2 \cdots y_m| = k+1)$ $)$ and $X_i \Rightarrow_G^* x_1 x_2 \cdots x_m X_j y_m \cdots y_2 y_1$. We prove the case for $|x_1 x_2 \cdots x_m| \le k+1$ and $|y_1 y_2 \cdots y_m| = k+1$ in the followings. The other case is similarly proved. Let $w' = x_{m+1} \cdots x_n y_n \cdots y_{m+1}$. We note $|w'| > 3k+2-(k+1)-(k+1) = k$ by $|w| > 3k+2$. Since $X_j \Rightarrow_G^* w'$ and $|w'| < |w|$, we have $w' \in Y_j$ using the assumption of the induction. Since $X_j \Rightarrow_G^* w'$ and $|w'| \ge k+1$, there exist x' and u' satisfying $w' = x'u'$ and $|u'| = j$ such that $a \Rightarrow_\gamma^* (x'u', x', 0)$ for some $a \in A$. Let $x = y_m \cdots y_{m-i+1}$, $u = y_{m-i} \cdots y_1$ and $z = x_1 x_2 \cdots x_m$. Since $X_i \Rightarrow_G^* zX_j xu$ and $|u| = i$, we have $((z, z, 0), (xu, u'x, -|u'|)) \in R_4$ and $(x'u', x', 0) \Rightarrow_\gamma (zx'u'xu, zx'u'x, 0)$. Since $zx'u'xu = w$ and $|u| = i$, we have $w \in Y_i$

Next we prove $L(G) \subset L(\gamma_G)$. Let $w \in L(G)$. If $|w| \le 3k+2$ then $(w, w, 0) \in A$ and $w \in L(\gamma_G)$. Assume $|w| > 3k+2$. According to the limitation of production rules in G, we have $S \Rightarrow_G x_1 X_{i_1} y_1 \Rightarrow_G x_1 x_2 X_{i_2} y_2 y_1 \Rightarrow_G^* w = x_1 x_2 \cdots x_n y_n \cdots y_2 y_1$ for $x_p, y_p \in T^*$, $|x_p| \le 1$, $|y_p| \le 1$ and $1 \le |x_p y_p|$ $(p = 1, \cdots, n)$. There exist m and X_i such that $((|x_1 x_2 \cdots x_m| = k+1$ and $|y_1 y_2 \cdots y_m| \le k+1)$ or $(|x_1 x_2 \cdots x_m| \le k+1$ and $|y_1 y_2 \cdots y_m| = k+1)$ $)$ and $S \Rightarrow_G^* x_1 x_2 \cdots x_m X_i y_m \cdots y_2 y_1$. We prove the case for $|x_1 x_2 \cdots x_m| \le k+1$ and $|y_1 y_2 \cdots y_m| = k+1$ in the followings. Let $w' = x_{m+1} \cdots x_n y_n \cdots y_{m+1}$. Since $X_i \Rightarrow_G^* w'$ and $|w'| \ge k+1$, we have $w' \in Y_i$ and there exist x' and u' satisfying $w' = x'u'$ and $|u'| = i$ such that $a \Rightarrow_\gamma^* (x'u', x', 0)$ for some $a \in A$. Since $S = X_1 \Rightarrow_G^* x_1 x_2 \cdots x_m X_i y_m \cdots y_2 y_1$ and $|x_1 x_2 \cdots x_m| + |y_1 y_2 \cdots y_m| \le 2k+2$, we have $((x_1 \cdots x_m,\ x_1 \cdots x_m,\ 0), (y_m \cdots y_1,\ u'y_m \cdots y_1,\ -i)) \in R_6$. Since $\mu((x_1 \cdots x_m,\ x_1 \cdots x_m,\ 0), \mu((x'u', x', 0), (y_m \cdots y_1,\ u'y_m \cdots y_1,\ -i))) = (w, w, 0)$, we have $w \in L(\gamma_G)$. □

Example 5. Consider the Language generated by linear grammar $G = (\{S\}, \{a, b\}, S, \{S \to ab, S \to aSb\})$.
The language generated by G is $L(G) = \{a^n b^n | n \ge 1\}$.
Now we can induce the domino $\boxed{\begin{matrix}\text{aaaaabbbbb}\\\text{aaaaabbbbb}\end{matrix}}$ by using pair of elements $(\boxed{\begin{matrix}\text{a}\\\text{a}\end{matrix}}, \boxed{\begin{matrix}\text{b}\\\text{bb}\end{matrix}}) \in R_6$, $(\boxed{\begin{matrix}\text{aa}\\\text{aa}\end{matrix}}, \boxed{\begin{matrix}\text{bb}\\\text{bb}\end{matrix}}) \in R_4$ and $\boxed{\begin{matrix}\text{aabb}\\\text{aab}\end{matrix}} \in A_3$. All of elements in A and R are listed in Appendix.

5. Conclusion

We can define the dominoes using set theoretical notations in Haskell and simulate sticker systems, finite automata and grammar systems. Using our system, we could find some insufficient conditions to construct the sticker systems written in [3]. One of related work is implementation of HaLex [5]. HaLex is a Haskell library enables us to model and manipulate a regular language. HaLex also provide the facilities for defining deterministic and non deterministic finite automata, regular expressions etc. It does not represent an infinite set as a language. One of the merits of our modules is treating the generated languages as an infinite set using lazy evaluations.

Acknowledgment

We would like to thank Ai Omodaka for her contribution to write Haskell codes and the great support made during

the research.

References

[1] Alhazov,A.,Cavaliere,M.: Computing by Observing Bio-systems:The Case of Sticker Systems, *LNCS 3384,Proc.DNA.*,**10**(2005) 1–13.

[2] Kari,L., Paun,G., Rozenberg,G., Salomaa,A.,Yu,S.: DNA computing, sticker systems, and universality, *Acta Informatica*,**35**(1998) 401–420.

[3] Paun,G.,Rozenberg,G.: Sticker systems, *Theoretical Computer Science*,**204**(1998) 183–203.

[4] Sakakibara,Y.,Kobayashi,S.: Sticker systems with complex structures, *Soft Computing*,**5**(2001) 114–120.

[5] Saraiva,J.: HaLeX: A Haskell Library to Model, Manipulate and Animate Regular Languages , *proceedings of the ACM Workshop on Functional and Declarative Programming in Education (FDPE/PLI'02), Pittsburgh, USA*(2002).

Appendix

In Appendix, we show examples of sticker systems generated from automata and grammar by using our Haskell module functions.

Example 6. For an automaton $M_1 = (\{0,1\}, \Sigma, \delta, 0, \{1\})$ in Example 1, we have the sticker system γ_{M_1} as follows.

$$\begin{aligned}\gamma_{M_1} &= (\Sigma, \rho, A, R)\\ \rho &= \{(a,a),(b,b)\}\\ A &= A_1 \cup A_2\end{aligned}$$

(A1) $\begin{bmatrix}\mathtt{b}\\\mathtt{b}\end{bmatrix}$ $\begin{bmatrix}\mathtt{ab}\\\mathtt{ab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{ba}\\\mathtt{ba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aab}\\\mathtt{aab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{baa}\\\mathtt{baa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aba}\\\mathtt{aba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bbb}\\\mathtt{bbb}\end{bmatrix}$ ...

(A2) $\begin{bmatrix}\mathtt{aaa}\\\mathtt{aa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bba}\\\mathtt{bb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{abb}\\\mathtt{ab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bab}\\\mathtt{ba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{baa}\\\mathtt{b}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aab}\\\mathtt{a}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aba}\\\mathtt{a}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bbb}\\\mathtt{b}\end{bmatrix}$

$$R = D \cup F$$

(D) $\begin{bmatrix}\mathtt{aaa}\\\mathtt{aaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bba}\\\mathtt{abb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aaa}\\\mathtt{baa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bba}\\\mathtt{bbb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{abb}\\\mathtt{aab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bab}\\\mathtt{aba}\end{bmatrix}$
$\begin{bmatrix}\mathtt{abb}\\\mathtt{bab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bab}\\\mathtt{bba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aba}\\\mathtt{aaab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{baa}\\\mathtt{aaba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aba}\\\mathtt{abab}\end{bmatrix}$
$\begin{bmatrix}\mathtt{baa}\\\mathtt{abba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aba}\\\mathtt{baab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{baa}\\\mathtt{baba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aba}\\\mathtt{bbab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{baa}\\\mathtt{bbba}\end{bmatrix}$
$\begin{bmatrix}\mathtt{aab}\\\mathtt{aaaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bbb}\\\mathtt{aabb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aab}\\\mathtt{abaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bbb}\\\mathtt{abbb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aab}\\\mathtt{baaa}\end{bmatrix}$
$\begin{bmatrix}\mathtt{bbb}\\\mathtt{babb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aab}\\\mathtt{bbaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bbb}\\\mathtt{bbbb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{baa}\\\mathtt{ab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{baa}\\\mathtt{bb}\end{bmatrix}$
$\begin{bmatrix}\mathtt{aab}\\\mathtt{aa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aab}\\\mathtt{ba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aba}\\\mathtt{aa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aba}\\\mathtt{ba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bbb}\\\mathtt{ab}\end{bmatrix}$
$\begin{bmatrix}\mathtt{bbb}\\\mathtt{bb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aaa}\\\mathtt{aaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aaa}\\\mathtt{aba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aaa}\\\mathtt{baa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aaa}\\\mathtt{bba}\end{bmatrix}$
$\begin{bmatrix}\mathtt{bab}\\\mathtt{aab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bab}\\\mathtt{abb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bab}\\\mathtt{bab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bab}\\\mathtt{bbb}\end{bmatrix}$
$\begin{bmatrix}\mathtt{bba}\\\mathtt{aab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bba}\\\mathtt{abb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bba}\\\mathtt{bab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bba}\\\mathtt{bbb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{abb}\\\mathtt{aaa}\end{bmatrix}$
$\begin{bmatrix}\mathtt{abb}\\\mathtt{aba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{abb}\\\mathtt{baa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{abb}\\\mathtt{bba}\end{bmatrix}$

(F) $\begin{bmatrix}\mathtt{b}\\\mathtt{ab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{b}\\\mathtt{bb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{a}\\\mathtt{aaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{a}\\\mathtt{aba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{a}\\\mathtt{baa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{a}\\\mathtt{bba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{ab}\\\mathtt{aab}\end{bmatrix}$
$\begin{bmatrix}\mathtt{ba}\\\mathtt{aba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{ab}\\\mathtt{bab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{ba}\\\mathtt{bba}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aa}\\\mathtt{aaaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bb}\\\mathtt{aabb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aa}\\\mathtt{abaa}\end{bmatrix}$
$\begin{bmatrix}\mathtt{bb}\\\mathtt{abbb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aa}\\\mathtt{baaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bb}\\\mathtt{babb}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aa}\\\mathtt{bbaa}\end{bmatrix}$ $\begin{bmatrix}\mathtt{bb}\\\mathtt{bbbb}\end{bmatrix}$ ...

Example 7. For a linear grammar $G_1 = \{\{S\}, \{a,b\}, S, \{S \to ab, S \to aSb\}\}$, we have the sticker system γ_{G_1} as follows.

$$\begin{aligned}\gamma_{G_1} &= (\Sigma, \rho, A, R)\\ \rho &= \{(a,a),(b,b)\}\\ A &= A_1 \cup A_2 \cup A_3\end{aligned}$$

(A1) $\begin{bmatrix}\mathtt{ab}\\\mathtt{ab}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aabb}\\\mathtt{aabb}\end{bmatrix}$

(A2) $\begin{bmatrix}\mathtt{ab}\\\mathtt{b}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aabb}\\\mathtt{abb}\end{bmatrix}$

(A3) $\begin{bmatrix}\mathtt{ab}\\\mathtt{a}\end{bmatrix}$ $\begin{bmatrix}\mathtt{aabb}\\\mathtt{aab}\end{bmatrix}$

$$R = R_1 \cup R_2 \cup R_3 \cup R_4 \cup R_5 \cup R_6$$

(R1) $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aa}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{bb}\end{bmatrix}\right)$ $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{ab}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{bb}\end{bmatrix}\right)$

(R2) $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aaa}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{b}\end{bmatrix}\right)$ $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aab}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{b}\end{bmatrix}\right)$

(R3) $\left(\begin{bmatrix}\mathtt{a}\\\mathtt{aa}\end{bmatrix}, \begin{bmatrix}\mathtt{b}\\\mathtt{b}\end{bmatrix}\right)$ $\left(\begin{bmatrix}\mathtt{a}\\\mathtt{ab}\end{bmatrix}, \begin{bmatrix}\mathtt{b}\\\mathtt{b}\end{bmatrix}\right)$
$\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aaa}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{bb}\end{bmatrix}\right)$ $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aab}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{bb}\end{bmatrix}\right)$

(R4) $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aa}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{ab}\end{bmatrix}\right)$ $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aa}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{bb}\end{bmatrix}\right)$

(R5) $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{a}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{abb}\end{bmatrix}\right)$ $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{a}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{bbb}\end{bmatrix}\right)$

(R6) $\left(\begin{bmatrix}\mathtt{a}\\\mathtt{a}\end{bmatrix}, \begin{bmatrix}\mathtt{b}\\\mathtt{ab}\end{bmatrix}\right)$ $\left(\begin{bmatrix}\mathtt{a}\\\mathtt{a}\end{bmatrix}, \begin{bmatrix}\mathtt{b}\\\mathtt{bb}\end{bmatrix}\right)$
$\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aa}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{abb}\end{bmatrix}\right)$ $\left(\begin{bmatrix}\mathtt{aa}\\\mathtt{aa}\end{bmatrix}, \begin{bmatrix}\mathtt{bb}\\\mathtt{bbb}\end{bmatrix}\right)$

K.K.K.R. Perera
Graduate School of Mathematics, Kyushu University, Japan.
E-mail: kissani(at)math.kyushu-u.ac.jp

Y.Mizoguchi
Faculty of Mathematics, Kyushu University,Japan.
E-mail: ym(at)math.kyushu-u.ac.jp