ON THE NONCLOSURE OF THE EQL CLASS
UNDER ERASING HOMOMORPHISM,

by

Yoav Raz

Technical Report #343

October 1984
ON THE NONCLOSURE OF THE EOL CLASS UNDER ERASING HOMOMORPHISM

by

Yoav Raz

ABSTRACT

The fact that the languages produced by interactive Lindenmayer systems with a fast local growth (Bounded Interaction L systems - BIL) are EOL disproves an old assumption that the EOL class is closed under erasing homomorphism. It follows that extending the mechanism of EOL systems using arbitrary homomorphism operation is as powerful as extending it using tables to get the ETOL class.

Key words: L, Lindenmayer, Developmental Systems.
1. INTRODUCTION

Interactive L systems with a fast local growth (BIL) produce EOL languages ([Raz (1979)], [Raz (1984)]). In [CK (1979)] it is shown that every ETOL language can be expressed using a BIL language, intersection with a regular set, and an erasing homomorphism. This result contradicts the above one provided that $L_{BIL} \subset L_{ETOL}$ (strictly) and that $L_{EOL}$ is closed under intersection with regular set and under erasing homomorphism. We assume that the source of contradiction is the closure under erasing homomorphism assumption and claim that the EQL class is not closed under this operation. It follows in addition that the ETOL class is the smallest full AFL containing the EQL class.

Chapter 2 of this paper presents basic notations and definitions. The construction establishing the equivalence between BIL systems and EOL systems appears in Chapter 3. The main results are included in Chapter 4. The construction described in Chapter 3 is demonstrated in the Appendix.
2. DEFINITIONS AND NOTATIONS

Definition 1:

A \( <k, l> \) system, \( k, l \geq 0 \), is a structure

\[ G = <\Sigma, P, \omega> \]

where

- \( \Sigma \) is a finite alphabet.
- \( P \) is a set of production rules, each with the following structure:
  
  \[ (a) \alpha (\beta) \rightarrow \gamma \]
  
  where

  - \( \alpha \in \Sigma \)
  - \( \alpha \in g^i \Sigma^j \) \( i + j = k \)
  - \( \beta \in \Sigma^m g^n \) \( m + n = l \)
  - \( \gamma \in \Sigma^* \)

  and \( g \) is a symbol not in \( \Sigma \).

  For each such \( \alpha \alpha \beta \in \bigcup_{i+j=k \atop m+n=l} g^i \Sigma^j \cdot \Sigma \cdot \Sigma^m g^n \) there is at least one rule in \( \cdot P \).

  \( \alpha, \beta \) are the context of \( \alpha \) in the rule.

  \( \omega \in \Sigma^* \) is the initial word or axiom.

An \( IL \) system, a system with interactions, is a \( <k, l> \) system where \( k + l > 0 \).

An \( OL \) system, a system without interactions, is a \( <k, l> \) system where \( k = l = 0 \).

In the case of \( OL \) systems, \( \alpha \) and \( \beta \) are empty strings. For this reason we will use the simpler notation \( \alpha \rightarrow \gamma \) (rather than \( (a) \alpha (\beta) \rightarrow \gamma \)) when dealing with \( OL \) systems.
Definition 2:

An ELL (EOL) system (Extended) is a structure

\[ G = \langle \Sigma, P, \omega, \Delta \rangle \]

where \( \langle \Sigma, P, \omega \rangle \) is an IL (OL) system and \( \Delta \) is a subset of \( \Sigma \).

The letters of \( \Delta \) are called terminals.

Using the same notations an ETIL (ETOL) system (Extended with Tables)

\[ G = \langle \Sigma, P, \omega, \Delta \rangle \]

where \( P \) is a set of elements \( P \) defined above. These elements are called tables.

Definition 3:

Let \( w \in \Sigma^* \), \( w = a_1a_2...a_m, a_i \in \Sigma, i = 1,2...m \).

1) The length of \( w \) is \( m \) and is denoted \( |w| \).

2) The length - \( k \) prefix of \( w \) is defined and denoted by \( \text{PRE}_k(w) = a_1a_2...a_k \).

3) The length - \( k \) suffix of \( w \) is defined and denoted by \( \text{SUF}_k(w) = a_{m-k+1}...a_m \).

Definition 4:

Let \( G \) be a \( <k,l> \)L system.

The relation \( (a)\alpha_1,\alpha_2,...,\alpha_r(\beta) \rightarrow \alpha_1\alpha_2,...,\alpha_r, r \geq 1 \), is a generalized rule of \( G \) iff

\[ (\alpha)\alpha_1(\text{PRE}_i(\alpha_2\alpha_3...\alpha_r\beta)) \rightarrow \alpha_i \]

\[ ... \]

\[ (\text{SUF}_k(\alpha\alpha_1\alpha_2...\alpha_{i-1})) \alpha_i (\text{PRE}_i(\alpha_{i+1}...\alpha_r\beta)) \rightarrow \alpha_i, \quad i = 2,3,...,r-1 \]

\[ ... \]

\[ (\text{SUF}_k(\alpha\alpha_1...\alpha_{r-1})) \alpha_r(\beta) \rightarrow \alpha_r \]

are production rules of \( G \).
Remark: Sometimes we shall drop some pairs of corresponding commas from both sides of the relation \( \rightarrow \) (at the cost of losing information, of course) and denote the above generalized rule as:

\[
(a) \beta_1, \beta_2, \ldots, \beta_t(\beta) \rightarrow \gamma_1, \gamma_2, \ldots, \gamma_t, \quad 1 \leq t \leq r
\]

where

\[
\beta_1 = a_1 a_2 \ldots a_{j_1}, \quad \gamma_1 = a_1 a_2 \ldots a_{j_1}
\]

\[
\vdots
\]

\[
\beta_t = a_{j_{t-1}+1} \ldots a_{j_t}, \quad \gamma_t = a_{j_{t-1}+1} \ldots a_{j_t}, \quad i = 2, \ldots, t
\]

and \( j_t = r \).

Definition 5:

The language \( L(G) \) of a \(<k, l>l\) system \( G \) is defined recursively as follows:

1) The initial word \( \omega \in L(G) \).

2) \( w_1 \in L(G) \) and \( (g^k) w_1(g^i) \rightarrow w_2 \) imply \( w_2 \in L(G) \).

When the left hand side of the implication in 2) holds, we say that \( w_1 \) directly derives \( w_2 \) in \( G \). Notation: \( w_1 \xrightarrow{G} w_2 \).

Let \( \xrightarrow{G} \) be the reflexive-transitive closure of \( \xrightarrow{G} \). When \( w_1 \xrightarrow{G} w_2 \) we say that \( w_1 \) derives \( w_2 \) in \( G \).

The language \( L(G) \) of an EIL (EOL) system \( G = \langle \Sigma, \Delta \rangle \) is \( L(G) = L(G') \cap \Delta^* \), where \( G' \) is the IL (OL) system \( \langle \Sigma, \Delta \rangle \).

The language \( L(G) \) of an ETIL (ETOL) system \( G = \langle \Sigma, \Delta \rangle \) is \( L(G) = L(G') \cap \Delta^* \), where \( G' \) is the system \( \langle \Sigma, \Delta \rangle \) (called TIL (TOL) system). \( L(G') \) is defined as for an IL (OL) system but every direct derivation uses rules of exactly one \( P \in P \).
Definition 6:

Let \((a) a_1, a_2, \ldots, a_r, (\beta) \rightarrow a_1, a_2, \ldots, a_r\) be a generalized rule of \(G\) and let \(\alpha a_1 \cdots a_r \beta\) be a subword of a word in \(g^k L(G)g^l\), then \(\alpha a_1 \cdots a_r \beta\) produces \(a_1 a_2 \cdots a_r\) in \(G\).

Definition 7:

A \(<k, l>_L\) system \(G\) is \(R\)-bounded (\(R\)-\(<k, l>_L\) system) if every subword with length \(R-1\) of a word in \(g^k L(G)g^l\) produces only words of length \(R-1\) or longer. It is bounded if it is \(R\)-bounded for some \(R\).

Remark:

It is clear by Definition 6 that \(R > k + l + 1\).

Definition 8:

An active subword in a \(<k, l>_L\) system \(G\) is a subword of a word in \(g^k L(G)g^l\).

Definition 9:

An active rule in an \(\langle k, l \rangle_L\) system \(G\) is a production rule which acts at least once in a derivation of a word in \(L(G)\).

Definition 10:

The \(R\)-shifts of a word \(w = a_1 a_2 \cdots a_m \in \Sigma^*\) is the word \(S_R(w)\) defined as follows:

\[
S_R(w) = \begin{cases} 
[a_1 a_2 \cdots a_R][a_2 a_3 \cdots a_{R+1}] \cdots [a_{m-R+1} \cdots a_m] & \text{if } m \geq R \\
\epsilon & \text{if } m < R
\end{cases}
\]

(\(\epsilon\) is the empty word.)
3. AN EOL SYSTEM ASSOCIATED WITH A HIL SYSTEM

3.1 An Informal Description

An EOL system \( G'' \) associated with an \( R - B \langle k, l \rangle \) system \( G = \langle \Sigma, P, \omega \rangle \) is a system without interactions ("context-free") simulating the interactive system \( G \). Its letters are \# \( , \) \( S \) and letters of the type

\[
[i, w, j] \quad 0 \leq i, j \leq M, \quad w \in \Sigma^R
\]

where \( \Sigma \) is the alphabet of \( G \) and \( M \) is a constant that will be determined later. The words \( w \) are subwords of words generated by \( G \), and the indices \( i, j \) define uniquely what is directly derived from \( w \), and are used as interfaces with the left and right neighboring letters respectively. The terminals of \( G'' \) are the letters \( [i, w, j] \) above. The meaning of "simulation" is that the sequence of the words \( w \) above, defined by the order of letters in a word of \( L(G'') \) is the \( R \)-shifts of a word in \( g^k L(G) g^l \).

Definition 11:

Let \( [i, w, j] \) be a letter of \( G'' \).

\[ w = \alpha \beta \gamma, \quad \alpha, \beta, \gamma \in \Sigma^*, \quad |\gamma| = l, \quad |\alpha| = k. \]

Let \( w, i \) determine uniquely the generalized rule of \( G \)

\[ (\alpha)a, \beta b\text{PRE}_{i-1}(\gamma) \rightarrow \delta_1, \delta_2 \quad \delta_1, \delta_2 \in \Sigma^*. \]

and let \( w, j \) determine uniquely the generalized rule

\[ (\text{SUF}_{k-1}(\alpha)a)\beta b(\gamma) \rightarrow \delta_3, \delta_4 \quad \delta_3, \delta_4 \in \Sigma^*. \]

\( i \) is compatible with \( j \) for \( w \) if and only if \( \delta_2 = \delta_3 \).

The production rules of \( G'' \) are of four types:
1) \([i, w, j] \rightarrow [i, w_1, i_1][i_1, w_2, i_2][i_2, w_3, i_3][i_3, \ldots, w_n, j], 0 \leq i, j, i_p \leq M, p = 1, 2, \ldots, n-1.\)

where \([w_1][w_2]\ldots[w_n] = S_R(w')\) for some \(w'\) defined later, if \(i\) compatible with \(j\) for \(w\).

2) \([i, w, j] \rightarrow \#, 0 \leq i, j \leq M, \) if \(i\) is not compatible with \(j\) for \(w\).

3) \(# \rightarrow \#.\)

4) \(S \rightarrow [i, w_1, i_1][i_1, w_2, i_2]\ldots[i_{n-1}, w_n, i_n], 0 \leq i_p \leq M, p = 0, 1, \ldots, n\)

and \(w_p, p = 1, 2, \ldots, n\) are defined by:

\([w_1][\bar{w}_2]\ldots[w_n] = S_R(g^k w g')\)

where \(w\) is the initial word of \(G\).

\(S\) is the initial word of \(G''\) and is used only at the beginning of the generation.

The definition of type 1 rules will guarantee that for every word \(w\) in \(L(G)\) there will be a word

\([i_0, w_1, i_1][i_1, w_2, i_2]\ldots[i_{n-1}, w_n, i_n] \text{ in } L(G'')\)

such that

\([w_1][\bar{w}_2]\ldots[w_n] = S_R(g^k w g')\)

Type 2,3 rules guarantee that such words are the only words in \(L(G'')\). If some \(i_{p-1}\) is not compatible with \(i_p\) for \(w_p, (p = 1, 2, \ldots, n)\) then a type 2 rule generates the appearance of the letter \(#\) which exclude the generated word from the language \(L(G'')\). The type 3 rule keeps the letter \(#\) in all words generated later.

It follows that there is a nonerasing homomorphism \(h\) such that

\(L(G) = h(L(G''))\),

implying that \(L(G)\) is an EOL language.
3.2 A Formal Description

Definition 12:
Let $G$ be a $<k,l>L$ system over the alphabet $\Sigma$. The nondeterminism degree of $G$ is $N$ if for some fixed $\alpha \in \Sigma$, $\beta \in \Sigma^l$ there are $N$ different production rules in $G$

$$(\alpha)a(\beta) \Rightarrow \gamma_0 | \gamma_1 | \ldots | \gamma_{N-1} \quad (\gamma_i \neq \gamma_j \text{ for } i \neq j, 0 \leq i, j \leq N-1).$$

(this is an abbreviation for the $N$ rules $$(\alpha)a(\beta) \Rightarrow \gamma_i, 0 \leq i \leq N-1$$, and there is no letter in $\Sigma$ developed within a fixed context into more than $N$ different words.

If $N = 1$ then $G$ is deterministic.

If $N > 1$ then $G$ is nondeterministic.

Definition 13:
Let $G = <\Sigma, P, \omega>$ an $R-B <k,l>L$ system with a degree of nondeterminism $N$.

An indexed system $G'$ associated with $G$ is a system $<\Sigma, P', \omega>$ in which:

1) If $(\alpha)a(\beta) \Rightarrow \gamma$ is an active rule in $P$ then $(\alpha)a(\beta) \Rightarrow i.\gamma$ is a rule in $P'$. $i$ is the index of the rule. $0 \leq i \leq N-1$.

2) All the rules in $P'$ are distinct. Every two rules are different at least in the indices.

3) for every active subword $a\alpha\beta$, $|\alpha| = k$, $|\beta| = l$ of $G$ there are exactly $N$ rules in $P'$:

$$(\alpha)a(\beta) \Rightarrow i.\gamma_i, 0 \leq i \leq N-1$$

Not all $\gamma_i$ should be distinct for a fixed $\alpha, \beta, \beta$. (But since the nondeterminism degree of $G$ is $N$, for some $\alpha, \beta$ they are distinct.)

The language $L(G')$ of an indexed system is defined as for a regular $L$ system while the indices are ignored, and do not take part in the rewriting process.
Remarks:

1) All active rules for a bounded IL system can be effectively found. See [Raz (1983)].
2) Clearly $L(G') = L(G)$.
3) Note that $\alpha, \beta, \gamma$ and $i$ uniquely determine $\gamma' \in (\alpha)\alpha(\beta) \to i, \gamma$.

For an indexed system an indexed generalized rule and its index can be defined:

Definition 14:

Following the definition of a generalized rule (Def. 4) an indexed generalized rule of an indexed system has the form

$$(\alpha)a_0,a_1,\ldots,a_{n-1}(\beta) \rightarrow i_0,\gamma_i,i_1,\gamma_1,\ldots,i_{n-1},\gamma_{n-1}$$

The index of the above generalized rule is

$$i = \sum_{p=0}^{n-1} i_p N^p$$

where $N$ is degree of nondeterminism of the original system.

Remarks:

1) In a deterministic system ($N = 1$) the index is always zero.
2) The index $i$ can take all values $0 \leq i < N^n$ and together with the left side of the generalized rule uniquely determines $\gamma_0,\gamma_1,\ldots,\gamma_{n-1}$.
3) In the counting base $N$, $i$ is described by the string $i_{n-1}i_{n-2}\cdots i_1i_0$.

We can define now the associated EOL system.
Definition 15:

Let $G$ be an $R-B <k,l>L$ system and let $G' = <\Sigma,P',\omega>$ be an indexed system associated with $G$. $G''$, the EOL system associated with $G'$ and $G$ is defined as follows:

$$G'' = <\Sigma'',P'',\Sigma,\Delta''>$$

where

$$\Sigma'' = \{[i,w,j]|w \text{ is an active subword in } G$$

with length $R; 0 \leq i,j \leq M = NR^{-k-l-1-1} \cup \{S,\#\}.$$ 

$P''$ includes four types of rules:

**Type 1:** Suppose

$$w = a\alpha \beta b\gamma, |\alpha| = k, |\gamma| = l,$$

is an active subword in $G$ with the length $R$. Let

$$(\alpha)\alpha,\beta(b\text{PRE}_{i-1}(\gamma)) \rightarrow \delta_1,\delta_2$$

$$(SUF_{k-1}(\alpha)\alpha)\beta(b(\gamma)) \rightarrow \delta_3,\delta_4$$

be generalized rules of $G$ with corresponding indexed generalized rules of $G'$ having the indices $i$ and $j$ respectively. (Since $|\alpha\beta| = |\beta\gamma| = R-k-l-1$ we have $0 \leq i,j \leq M = NR^{-k-l-1-1}$.)

If $\delta_2 = \delta_3 (i \text{ is compatible with } j \text{ for } w)$ define $\beta_1, \beta_2$ as follows:

$$\beta_1\beta_2 = \delta_1\delta_2, |\beta_2| = R-1.$$ 

(Explicitly $\beta_2 = SUF_{R-1}(\delta_1\delta_2)$, and its existence is guaranteed since $G$ is $R$ bounded. $\beta_1$ can be an empty string.)

From the generalized rule combining the two generalized rules above:

$$(\alpha)\alpha,\beta(b(\gamma)) \rightarrow \beta_1\beta_2,\delta_4$$

the following type 1 rules are derived:
\[ [i,w,f] \rightarrow [i,w_1,i_1][i_1,w_2,i_2]...[i_{n-1},w_n,f] \]

where

\[
[w_1][w_2]...[w_n] = S_P(\delta_{i,w,f})
\]

\[
\delta_{i,w,f} = \begin{cases} 
  a & \text{if } \alpha=g, \gamma=g^i \\
  b & \text{if } \alpha=g, \gamma \neq g^i \\
  \beta & \text{if } \alpha \neq g, \gamma=g^i \\
  \gamma & \text{if } \alpha \neq g, \gamma \neq g^i 
\end{cases}
\]

and \(0 \leq i_p \leq M, p = 1,2,...,n-1\) in all possible combinations.

**Type 2:** Using the denotations above, if \(\delta_1 \neq \delta_3\) (\(i\) is not compatible with \(f\) for \(w\)) then \(P^n\) includes the rule \([i,w,f]\) \(\rightarrow \#\).

**Type 3:** \# \(\rightarrow \#\).

**Type 4:** \(S \rightarrow [i_0,w_1,i_1][i_1,w_2,i_2]...[i_{n-1},w_n,i_n]\) \(\text{ where} \)

\[
[w_1][w_2]...[w_n] = S_P(g^k \omega g^l)
\]

and \(0 \leq i_p \leq M, p = 0,1,...,n\) in all possible combinations.

\(\Lambda'' = \Sigma'' \setminus \{S,\#\}\).

### 3.3 Results

The main results concerning the languages of \(\text{BIL}\) systems and their associated \(\text{EOL}\) systems are summarized in the following theorem and corollary.

**Theorem 1:** [Raz (1984)].

Let \(G\) be an \(R-B<k,l>L\) system and let \(G''\) be its associated \(\text{EOL}\) system.

1. \(w \in L(G)\) if and only if

\(\hat{\cdot}\)
where
\[ [i_0, w_1, i_1][i_1, w_2, i_2]...[i_{n-1}, w_n, i_n] \in L(G'') \]

for some sequence \( i_0, i_1, \ldots, i_n \), \( 0 \leq i_p \leq M, \ p = 0, 1, \ldots, n \). (\( M \) is a parameter of \( G'' \), defined in Definition 15.)

(2) A nonerasing homomorphism \( h \) can be constructed such that
\[ L(G) = h(L(G'')) \]

Corollary:

\( L(G) \) is an EOL language. (The EOL class is closed under nonerasing homomorphism, see [Herman & Rozenberg (1975)].)
4. THE NONCLOSURE OF THE EOL CLASS UNDER ERASING HOMOMORPHISM

In [Culik & Karhumaki (1979)] $s$-G2L systems are defined and a constructive proof is given to the following theorem:

**Theorem 2: [Culik & Karhumaki (1979)]**

For any ETOL language $L$ there is an $s$-G2L system $G$, an erasing homomorphism $h$ and a regular set $R$ such that

$$L \neq h(L(G)) \cap R.$$  

The relationship between the $s$-G2L class and the EOL class is expressed in the following lemma.

**Lemma:**

Any $s$-G2L language is a $\Sigma^B<1,1>L$ language.

The proof follows directly from the definitions.

Now, provided that $L_{EOL} \subseteq L_{ETOL}$ and that $L_{EOL}$ is closed under intersection with a regular set, we are able to prove the main result.

**Theorem 3:**

The EOL class is not closed under erasing homomorphism.

**Remark:** It is closed under nonerasing one. See [Herman & Rozenberg (1975)].

**Proof:** Suppose that it is closed under erasing homomorphism. Choose $L \in L_{ETOL} - L_{EOL}$ and characterize it using Theorem 2.
where \( L(G) \) is a BIL language (the Lemma) and hence an EOL language (corollary of Theorem 1). Because of the closure under \( h \) and under intersection with \( R \), \( L \) should be an EOL language, and hence not a language in \( L_{\text{STOL}} - L_{\text{EOL}} \).

Remark: In [Culik & Karhumaki (1979)] there is a result saying that there exists a \( s-\text{G2L} \) language (a BIL one) which is not an EOL one, and hence contradicting the corollary of Theorem 1. The proof there is based on the closure of \( L_{\text{EOL}} \) under erasing homomorphism.

The following theorem is another consequence of the discussion above:

**Theorem 4: [Karhumaki & Raz (1984)]**

\[ L_{\text{STOL}} \text{ is the smallest full-AFL containing } L_{\text{EOL}}. \]

**Proof:** By Theorems 1, 2 and the lemma any \( L \in L_{\text{STOL}} \) can be constructed from an EOL language through full-AFL operations. \( L_{\text{STOL}} \) is a full-AFL ([Herman & Rozenberg (1975)]). Hence it is the smallest. (Otherwise take a language in the difference of the two full-AFLs to get a contradiction.)

The meaning of the theorems above is that applying erasing homomorphism to EOL languages is as powerful as extending the EOL class using tables.
REFERENCES


APPENDIX

AN EXAMPLE OF A 3-B<1,0>L SYSTEM AND ITS ASSOCIATED EOL SYSTEM

Let $G$ be the following $<1,0>L$ system:

$$G = \langle \{a, b\}, P, ab \rangle$$

where

$$P = \{ (g)a \rightarrow aa, (a)a \rightarrow aa, (a)b \rightarrow ab, (b)b \rightarrow bb \} \quad \text{(active rules only)}.$$

It is easy to verify that this is a 3-B$<1,0>L$ system generating the following language $L(G)$:

$$L(G) = \{ a^k b^l \mid k + l = 2^n, l = 1, 2, ..., 2^n - 1, n = 1, 2, ... \}$$

In an indexed associated system $G'$

$$G' = \langle \{a, b\}, P', ab \rangle$$

a possible $P'$ is:

$$P' = \{ (g)a \rightarrow 0.aa \mid 1.aa \}
\quad (a)a \rightarrow 0.qa \mid 1.qa
\quad (a)b \rightarrow 0.ab \mid 1.bb
\quad (b)b \rightarrow 0.bb \mid 1.bb \}$$

The degree of nondeterminism is $N = 2$.

Hence $M = N^{R-k-l-1} - 1 = 2^1 - 1 = 1$.

The appropriate associated EOL system $G''$ is defined as follows:

$$G'' = \langle \Sigma'', P'', S, \Delta'' \rangle$$

where

$$\Sigma'' = \{ [i, gaa, j], [i, gab, j], [i, aaa, j],
\quad [i, aab, j], [i, abb, j], [i, bbb, j] \mid 0 \leq i, j \leq 1 \} \cup \{ S \}$$

Remark: Since $R - k - l = 2$ the letter # is not necessary (see Remark 3 in Definition 15).
The rules in \( P'' \) are constructed as follows: Since \( R-k-l=2 \) the generalized rules determining the indices \( i, j \) in the letters of \( \Sigma'' \) are based on a single production rule each (e.g. \( ab \rightarrow 0.ab \)). \( P'' \) includes only rules of types 1 and 4 in Definition 15. Type 1 rules can be generated using the following table:

<table>
<thead>
<tr>
<th>( [i,w,j] )</th>
<th>generalized Rule for ( w )</th>
<th>( \delta_{i,w,j} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i,j=0,1 )</td>
<td>([i,gaa,0])</td>
<td>((g)a,a \rightarrow aa,aa)</td>
</tr>
<tr>
<td>( i=0,1 )</td>
<td>([i,gab,0])</td>
<td>((g)a,b \rightarrow aa,ab)</td>
</tr>
<tr>
<td>( i=0,1 )</td>
<td>([i,gab,1])</td>
<td>((g)a,b \rightarrow aa,bb)</td>
</tr>
<tr>
<td>( i,j=0,1 )</td>
<td>([i,aaa,0])</td>
<td>((a)a,a \rightarrow aa,aa)</td>
</tr>
<tr>
<td>( i=0,1 )</td>
<td>([i,aab,0])</td>
<td>((a)a,b \rightarrow aa,ab)</td>
</tr>
<tr>
<td>( i=0,1 )</td>
<td>([i,aab,1])</td>
<td>((a)a,b \rightarrow aa,bb)</td>
</tr>
<tr>
<td>( j=0,1 )</td>
<td>([0,abb,0])</td>
<td>((b),b \rightarrow ab,bb)</td>
</tr>
<tr>
<td>( j=0,1 )</td>
<td>([1,abb,0])</td>
<td>((b),b \rightarrow bb,bb)</td>
</tr>
<tr>
<td>( i,j=0,1 )</td>
<td>([i,bbb,j])</td>
<td>((b),b \rightarrow bb,bb)</td>
</tr>
</tbody>
</table>

The type 1 rules are the following where \( i_p=0,1 \) for \( p=1,2,0 \leq i,j \leq 1 \).

\[
\begin{align*}
[i,gaa,j] & \rightarrow [i,gaa,i_1][i_1,aaa,i_2][i_2,aaa,j] \\
[i,gab,0] & \rightarrow [i,gaa,i_1][i_1,aaa,i_2][i_2,aab,0] \\
[i,gab,1] & \rightarrow [i,gaa,i_1][i_1,aab,i_2][i_2,abb,1] \\
[i,aaa,j] & \rightarrow [i,aaa,i_1][i_1,aaa,j] \\
[i,aab,0] & \rightarrow [i,aaa,i_1][i_1,aab,0] \\
[i,aab,1] & \rightarrow [i,aab,i_1][i_1,abb,1] \\
[0,abb,j] & \rightarrow [0,abb,i_1][i_1,bbb,j] \\
[1,abb,j] & \rightarrow [1,bbb,i_1][i_1,bbb,j] \\
[i,bbb,j] & \rightarrow [i,bbb,i_1][i_1,bbb,j]
\end{align*}
\]
The type 4 rules are:

\[ S \rightarrow [i, g, a, b, j], \quad 0 \leq i, j \leq 1 \]

The terminal alphabet is:

\[ \Delta'' = \Sigma'' - \{S\} \]

A simulation of \( G \) by \( G'' \) is demonstrated by the following derivation chains:

For the derivations

\[ ab \Rightarrow aabb \Rightarrow aaaaabbb' \Rightarrow \ldots \]

the corresponding derivations in \( G'' \) may be

\[ S \Rightarrow [0, g, a, b, 1] \Rightarrow [0, g, a, a, 0][0, a, a, b, 0][0, a, b, b, 1] \Rightarrow \]

\[ \Rightarrow [0, g, a, a, 1][1, a, a, 0][0, a, a, a, 0][0, a, a, a, 1][1, a, a, b, 0][0, a, b, b, 0][0, b, b, b, 1] \Rightarrow \ldots \]

Note that

\[ S_9(ga) = [g a b] \]

\[ S_9(gaabb) = [g a a][a a b][b b b] \]

\[ S_9(ga^5b^3) = [g a a][a a a][a a a][a a a][a a b][b b b] \]