Acta Cybernetica **14**, (2000) 569-582.

Two remarks on variants of simple eco-grammar systems*

Judit Csima †

Abstract

Two powerful variants of simple eco-grammar systems, namely extended tabled simple eco-grammar systems (ETEG systems) and weak extended simple eco-grammar systems (wEEG systems) are studied. It is proved that both modifications of the original definition result in universal power: all recursively enumerable languages can be obtained both by ETEG and by wEEG systems.

1 Introduction

Eco-grammar systems form a grammatical framework proposed in [2] for modelling living systems consisting of several agents and a common environment. In the original definition of an eco-grammar system, the environment is described by a Lindenmayer system which determines its evolution; the agents are represented by context-free grammars and by Lindenmayer systems : the Lindenmayer systems determine their development, while the context-free grammars describe their actions. The interaction between the agents and the environment is ensured by the computable functions ψ and ϕ, which allow the agents to adapt to the environment both in their development and in their actions.

In the original model there are no terminal and nonterminal symbols. In [2] it was shown that this model is very strong as far as the generative capacity is concerned: all recursively enumerable languages can be obtained as languages of extended eco-grammar systems (that is with systems with a distinguished terminal alphabet) with a very simple choice of the functions ψ and ϕ.

Because of this result another, simpler variant of eco-grammar systems was introduced in [2]: the simple eco-grammar system. In a simple eco-grammar system the interaction between the environment and the agents is restricted and the agents do not have an inner representation. Other variants like non-extended simple eco-grammar systems in [3] and conditional tabled eco-grammar systems in [4], [5] and [10] were introduced and studied.

*Research supported by the Hungarian Scientific Foundation "OTKA" Grant No. T029615

†Computer and Automation Research Institute, Hungarian Academy of Sciences, Kende u. 13-17, H-1111, Budapest, Hungary. E-mail: csima@cs.bme.hu

Besides these directions, the study of extended simple eco-grammar systems continued in [6], where it was proved that the hierarchy according to the number of the agents is a collapsing one and that the class of languages generated by extended simple eco-grammar systems without λ-rules is between the class of languages generated by extended $0L$ systems and the class of languages generated by matrix grammars with appearance checking. (For more information about these language classes the reader is referred to [9] and [8].) The general case, where λ-rules are allowed in the system, remained open. In [1] it was shown that the hierarchy collapses in the general case as well, even if we consider different derivation modes, but the place of this collapsing language class remained unsolved.

In [11] simple eco-grammar systems with prescribed teams were examined. It was proved there that extended tabled simple eco-grammar systems with teams of agents with prescribed members and operating according to a weak rewriting steps (that is the derivation is not blocked if some agents from a team cannot perform any action) can generate all recursively enumerable languages. In this article we present a stronger result: it is not necessary to use prescribed teams to reach the power of Turing machines. Moreover, both extended tabled simple eco-grammar systems (using the original derivation mode, not the weak one) and weak extended simple eco-grammar systems (without tables) are enough to generate all recursively enumerable languages. (We note that the definition of a weak derivation step we use in this article is a slightly different one compared to [11], therefore only the first result is stronger than the one in [11].)

More precisely, we consider two variants of extended simple eco-grammar systems for which the question of their generative power will be answered.

The first part of the article deals with extended tabled simple eco-grammar systems, where instead of one $0L$ system the environment can be represented by more than one $0L$ system, called tables. Thus the environment can vary its behaviour step by step. Allowing this possibility, all recursively enumerable languages can be obtained even with one agent.

In the second part of the article we present weak extended simple eco-grammar systems. In this case the definition of the derivation step is different from the original definition of an extended simple eco-grammar system. In this modified version the derivation is not blocked if some agents cannot perform any action on the sentential form. We show that the generative power of Turing machines can be reached also in this case.

2 Preliminaries

Here we present the notions and notations used in this article, for further information the reader is referred to [9], [8] and [7].

The set of all non-empty words over a finite alphabet V is denoted by V^+, the empty word is denoted by λ; $V^* = V^+ \cup \{\lambda\}$. For a set V, we denote by *card* (V) the cardinality of V. For a word x, we denote by $|x|$ the length of x. If $x = x_1 \cdots x_n$ is a word over an alphabet V, $x_j \in V$ for $1 \leq j \leq n$, $[x, i, k]$ denotes the word

$[x_1, i, k] \cdots [x_n, i, k]$ for $1 \leq i, k$.

By a context-free production or by a context-free rule (a CF rule, for short) over an alphabet V we mean a production of the form of $a \to u$, where $a \in V$ and $u \in V^*$. A CF rule is a λ-rule (or a deletion rule) if $u = \lambda$.

We also use the following notations: for a set of CF rules R, $dom(R)$ denotes the set of all letters appearing in the left-hand side of a rule in R. For a word x over an alphabet V, $alph(x)$ denotes the set of all letters appearing in x.

A $0L$ system is a triplet $H = (V, P, \omega)$, where V is a finite alphabet, P is a set of context-free rules over V, and $\omega \in V^*$ is the axiom. Moreover, P has to be complete, that is for each symbol a from V there must be at least one rule in P with this letter in the left-hand side. $0L$ systems use parallel derivations: we say that x directly derives y in a $0L$ system $H = (V, P, \omega)$, written as $x \overset{0L}{\Longrightarrow}_H y$, if $x = x_1 x_2 \cdots x_n$, $y = y_1 y_2 \cdots y_n$, where $x_i \in V$, $y_i \in V^*$, and the rules $x_i \to y_i$ are in P for $1 \leq i \leq n$.

A $T0L$ system is a triplet $H = (V, T, \omega)$, where V is a finite alphabet, $T = \{T_1, \ldots, T_k\}$ is a set of tables over V, where each table T_i for $1 \leq i \leq k$ is a complete set of CF rules over V, and $\omega \in V^*$ is the axiom. We say that x directly derives y in a $T0L$ system $H = (V, T, \omega)$, written as $x \overset{T0L}{\Longrightarrow}_H y$, if $x \overset{0L}{\Longrightarrow}_{H_i} y$ for some i , $1 \leq i \leq k$, with the $0L$ system $H_i = (V, T_i, \omega)$.

An $ET0L$ system is a quadruple $H = (V, T, \Delta, \omega)$, where $H' = (V, T, \omega)$ is a $T0L$ system, and $\Delta \subseteq V$ is the terminal alphabet. In an $ET0L$ system $H = (V, T, \Delta, \omega)$ x directly derives y, written as $x \overset{ET0L}{\Longrightarrow}_H y$, if $x \overset{T0L}{\Longrightarrow}_{H'} y$.

The transitive and reflexive closure of $\overset{ET0L}{\Longrightarrow}_H$ is denoted by $\overset{ET0L}{\Longrightarrow}^*_H$.

The generated language of the $ET0L$ system H (denoted by $L(H)$) is

$$L(H) = \{ w \in \Delta^* \mid \omega \overset{ET0L}{\Longrightarrow}^*_H w \}.$$

That is, in an $ET0L$ system only words over a distinguished subalphabet are in the generated language. A language is said to be an $ET0L$ language if there is an $ET0L$ system which generates it.

$T0L$ and $0L$ systems are special cases of $ET0L$ systems: $\Delta = V$ stands in both cases; moreover, in the case of $0L$ systems $T = \{T_1\}$ also holds. Therefore the above definition gives the generated language for these systems as well.

A random-context grammar is a quadruple $G = (N, T, P, S)$ where N is the set of nonterminals, T is the set of terminals, S is the axiom, and P is a finite set of random-context rules, that is triplets of the form of $(C \to \alpha, Q, R)$, where $C \to \alpha$ is a CF rule over $N \cup T$, where $C \in N$, and Q and R are subsets of N. For $x, y \in (N \cup T)^*$, we write $x \overset{rc}{\Longrightarrow} y$ iff $x = x_1 C x_2$, $y = x_1 \alpha x_2$ for some $x_1, x_2 \in (N \cup T)^*$, $(C \to \alpha, Q, R)$ is a triplet in P, all symbols of Q appear and no symbol of R appears in $x_1 C x_2$ (Q is called the permitting context, and R is called the forbidding context of the rule $C \to \alpha$. If Q and/or R are empty, no check is necessary.) This is a slightly modified but equivalent version of the definition presented in [7].

If the forbidding context is empty for every rule, we speak about a random-context grammar without appearance checking, otherwise the grammar is with appearance checking. For the sake of brevity we will refer to a random-context

grammar with appearance checking and with λ-rules as a random-context grammar.

The generated language of a random-context grammar consists of all the words which can be generated in some steps from axiom S. The class of languages which can be generated by random-context grammars is denoted by $\mathcal{RC}_{ac}^{\lambda}$.

Now we present the definition of an extended simple eco-grammar system, as introduced in [6].

Definition 2.1 *An extended simple eco-grammar system is a construct* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$, *where*

- V_E *is a finite, non-empty alphabet,*
- P_E *is a complete set of CF rules over* V_E, *i.e. for each letter of* V_E *there exists at least one rule in* P_E *with this letter in the left-hand side,*
- R_i *is a non-empty set of CF rules over* V_E *for* $1 \leq i \leq n$,
- $\omega \in V_E^*$, *and*
- $\Delta \subseteq V_E$.

In this construct V_E is the alphabet and P_E is the set of the evolution rules of the environment. The ith agent is represented by R_i, $1 \leq i \leq n$, its set of action rules. The current state of the environment, which is also the state of the eco-grammar system, is the current sentential form. String ω is the initial state. Δ is the terminal alphabet and shortly we will see that only words over Δ are in the generated language of Σ.

The system changes its state by a simultaneous action of the agents and by a parallel rewriting according to P_E.

Definition 2.2 *Consider an extended simple eco-grammar system* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$. *We say that* x *directly derives* y *in* Σ *(with* $x \in V_E^+$ *and* $y \in V_E^*$, *written as* $x \overset{eco}{\Longrightarrow}_{\Sigma} y$), *if*

- $x = x_1 Z_1 x_2 Z_2 \cdots x_n Z_n x_{n+1}$, *with* $Z_i \in V_E$, $x_j \in V_E^*$, $1 \leq i \leq n$, *and* $1 \leq j \leq n+1$,
- $y = y_1 w_1 y_2 w_2 \cdots y_n w_n y_{n+1}$, *with* $y_i, w_j \in V_E^*$, $1 \leq i \leq n$, *and* $1 \leq j \leq n+1$,
- *there exists a permutation of the agents, namely* $R_{j_1}, R_{j_2}, \ldots, R_{j_n}$, *such that* $Z_i \to w_i \in R_{j_i}$, *for* $1 \leq i \leq n$, *and*
- $x_i = \lambda$ *or* $x_i \overset{0L}{\Longrightarrow}_E y_i$, *for* $1 \leq i \leq n$, *where* $E = (V_E, P_E, \omega)$ *is the* $0L$ *system of the environment.*

We denote the transitive and reflexive closure of $\overset{eco}{\Longrightarrow}_{\Sigma}$ by $\overset{eco}{\Longrightarrow}_{\Sigma}^*$.
The generated language consists of the words over Δ which can be obtained in some derivation steps starting from the axiom.

Definition 2.3 *Consider an extended simple eco-grammar system* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$. *The generated language of* Σ *is the following:*

$$L(\Sigma) = \{ v \in \Delta^* \mid \omega \overset{eco}{\Longrightarrow}_{\Sigma}^* v \}.$$

3 Extended tabled simple eco-grammar systems

In this section a modified version of an extended simple eco-grammar system, called an extended tabled simple eco-grammar system, is investigated. In such a system the environment can be represented by more than one $0L$ system. We show that this device is as powerful as Turing machines.

Definition 3.1 *An extended tabled simple eco-grammar system (an ETEG system, for short) is a construct* $\Sigma = (V_E, T_E, R_1, \ldots, R_n, \omega, \Delta)$, *where*

- $(V_E, T_E, \Delta, \omega)$ *is an* $ET0L$ *system, and*
- R_i *is a non-empty set of* CF *rules over* V_E *for* $1 \leq i \leq n$.

Here V_E, R_i, ω and Δ have the same meaning as in Definition 2.1, namely they are the alphabet of the system, the production sets representing the agents, the axiom, and the terminal alphabet. T_E is the set of tables of the environment.

In an extended tabled eco-grammar system the environment can choose a $0L$ system in each derivation step to perform a parallel rewriting.

Definition 3.2 *Consider an extended tabled simple eco-grammar system* $\Sigma = (V_E, T_E, R_1, \ldots, R_n, \omega, \Delta)$. *We say that* x *directly derives* y *in* Σ *(with* $x \in V_E^+$ *and* $y \in V_E^*$, *written as* $x \overset{t-eco}{\Longrightarrow}_\Sigma y$*) if* $x \overset{eco}{\Longrightarrow}_{\Sigma_i} y$ *for the extended simple eco-grammar system* $\Sigma_i = (V_E, T_i, R_1, \ldots, R_n, \omega, \Delta)$ *for some* $1 \leq i \leq k$.

We denote the transitive and reflexive closure of $\overset{t-eco}{\Longrightarrow}_\Sigma$ by $\overset{t-eco}{\Longrightarrow}^*_\Sigma$.

Definition 3.3 *The generated language of an extended tabled simple eco-grammar system* $\Sigma = (V_E, T_E, R_1, \ldots, R_n, \omega, \Delta)$ *is the following:*

$$L(\Sigma) = \{ v \in \Delta^* \mid \omega \overset{t-eco}{\Longrightarrow}^*_\Sigma v \}.$$

The class of languages which can be generated by an $ETEG$ system with n agents is denoted by $\mathcal{L}(\mathcal{ETEG}, \backslash)$.

Now we present an example to illustrate the power of $ETEG$ systems.

Example 3.1 *Let* $\Sigma = (V_E, T_E, R_1, \omega, \Delta)$ *be the following ETEG system:*

- $V_E = \{ S, B, N, D, S', a, b \}$,
- $T_E = \{ T_1, T_2, T_3, T_4 \}$, *where*
  $T_1 = \{ S \to D, N \to N, a \to a, b \to b, D \to D, S' \to S', B \to B \}$,
  $T_2 = \{ S \to \lambda, N \to N, a \to a, b \to b, D \to D, S' \to D, B \to B \}$,
  $T_3 = \{ S \to D, N \to N, a \to a, b \to b, D \to D, S' \to D, B \to bB \}$,
  $T_4 = \{ S \to D, N \to N, a \to a, b \to b, D \to D, S' \to D, B \to b \}$,
- $R_1 = \{ S \to aBNS, N \to \lambda, S' \to \lambda \}$,
- $\omega = SS'$, *and*

- $\Delta = \{ a, b \}$.

We show that the generated language of this system is

$$L(\Sigma) = \{ (ab^n)^m \mid 1 \leq n \leq m \} \cup \{\lambda\}.$$

First we note the following: S can be deleted only by the environment and S' can be deleted only by the agent. These two events must happen in the same derivation step because of the following reasons. If the environment deletes S, that is uses table T_2, and the agent does not apply the rule $S' \to \lambda$, then the environment rewrites S' to D, which "blocks" the derivation since this D will never disappear from the sentential form. If the agent deletes S' in a derivation step and the environment does not delete S, that is does not use table T_2, it introduces symbol D again.

We have seen that S and S' disappear from the sentential form in the same derivation step. Before this step the agent has to use the rule $S \to aBNS$, or otherwise the environment introduces a D; the environment has to use the first table T_1, or otherwise S' would be rewritten to D. Thus either the derivation is $SS' \overset{t-eco}{\Longrightarrow} \lambda$, or the first few steps are $SS' \overset{t-eco}{\Longrightarrow}{}^{*} (aBN)^m SS' \overset{t-eco}{\Longrightarrow} (aBN)^m$. After these steps the only possibility for the agent to work is by using rule $N \to \lambda$. During these steps the environment can use any of its tables, therefore it can introduce b letters before all the B's or it can rewrite either all B's to B or all B's to b's. Because there are only m symbols N in the sentential form, the derivation lasts exactly m more steps. Hence at the end of the derivation, when there are no more N symbols left, there are at least one but at most m symbols b after each a.

The above explication showes that all non-empty generated words are of the form of $(ab^n)^m$, $1 \leq n \leq m$. It follows from the construction that all words of this form as well as the empty word, λ, are in the generated language, which is thus indeed:

$$L(\Sigma) = \{ (ab^n)^m \mid 1 \leq n \leq m \} \cup \{\lambda\}.$$

This example shows that even a very simple extended tabled simple eco-grammar system with only one agent is able to produce a quite complicated language, namely a language which is not an $ET0L$ one (see [8]).

We show that the generative capacity of these systems reaches that of Turing machines. This is a direct consequence of the following lemma.

Lemma 3.1 $\mathcal{RC}_{ac}^{\lambda} \subseteq \mathcal{L}(\mathcal{ETEG}, \infty)$

Proof Let $G = (N, T, S, P)$ be a random-context grammar. Without loss of generality, we can asssume that the rules in P have the form $(C \to \alpha, Q, R)$, with $C \in N$, $\alpha \in (N \cup T)^*$, $card(Q) \leq 1$, and $card(R) \leq 1$ (see [7]). Moreover, we can assume that there are no rules with $Q = R$, $R = \{C\}$ or $Q = \{C\}$, because rules of the first two types are not applicable and in the last case the rule is equivalent to the rule $(C \to x, \emptyset, R)$.

We denote by r the number of rules in P and by V the set $N \cup T$. The rules of P are enumerated as $p_i = (C_i \to \alpha_i, Q_i, R_i)$. We will refer to the components of the ith rule as C_i, α_i, Q_i, and R_i.

Now we will construct a simple extended tabled eco-grammar system $\Sigma = (V_E, T_E, R_1, \omega, \Delta)$ such that $L(\Sigma) = L(G)$.

Let

- $V_E = V \cup \{[X,i] \mid X \in V, 1 \leq i \leq r\}$
  $\cup \{[X',i] \mid X \in V, 1 \leq i \leq r\}$
  $\cup \{D, U, Z, Z'\}$
  $\cup \{[U,i] \mid 1 \leq i \leq r, Q_i = \emptyset\}$
  $\cup \{[U',i] \mid 1 \leq i \leq r\} \cup$
  $\cup \{[\widehat{U},i] \mid 1 \leq i \leq r, Q_i \neq \emptyset\}$,
  where $D, U, Z \notin V$,

- $T_E = \{T_{(i)}, T'_{(i)}, T''_{(i)} \mid 1 \leq i \leq r\}$, where

  $T_{(i)} = \{X \to [X,i] \mid X \in V\}$
  $\cup \{U \to [U,i] \mid Q_i = \emptyset\}$
  $\cup \{U \to [\widehat{U},i] \mid Q_i \neq \emptyset\}$,
  for $1 \leq i \leq r$,

  $T'_{(i)} = \{[X,i] \to [X',i] \mid X \in V, \{X\} \neq R_i\}$
  $\cup \{[B,i] \to D \mid \{B\} = R_i\}$
  $\cup \{Z' \to Z'\}$
  $\cup \{[\widehat{U},i] \to [U',i] \mid Q_i \neq \emptyset\}$
  for $1 \leq i \leq r$,

  $T''_{(i)} = \{[X',i] \to X \mid X \in V\}$
  $\cup \{Z' \to Z, Z' \to \lambda, [U',i] \to U, [U',i] \to \lambda\}$
  for $1 \leq i \leq r$,

- $R_1 = \{Z \to Z'\}$
  $\cup \{[U,i] \to [U',i] \mid 1 \leq i \leq r, Q_i = \emptyset\}$
  $\cup \{[A,i] \to [A',i] \mid 1 \leq i \leq r, Q_i = \{A\}\}$
  $\cup \{[C'_i,i] \to \alpha_i \mid 1 \leq i \leq r\}$,

- $\omega = SUZ$, and

- $\Delta = T$.

Those symbols which are not mentioned above in the tables are rewritten into D; these rules make the tables complete.

We introduce different alphabets according to the rules of P in the following way. These alphabets are multiplied versions of V: for the ith rule of P we have alphabets $\{[X,i] \mid X \in V\}$ and $\{[X',i] \mid X \in V\}$. Moreover, we have some special additional

symbols in the ETEG system in order to coordinate the derivation. These are $\{D, U, Z, Z'\} \cup \{[U', i] \mid 1 \leq i \leq r\} \cup \{[U, i] \mid 1 \leq i \leq r, Q_i = \emptyset\} \cup \{[\widehat{U}, i] \mid 1 \leq i \leq r, Q_i \neq \emptyset\}$. By D the derivation is "blocked": if this symbol appears, the derivation never results in a terminal word. Symbol U allows the agent to work when $Q_i = \emptyset$.

First we show how a derivation step of G can be simulated by Σ. During the simulation the sentential form has the form wUZ, where the word w corresponds to the sentential form of G, while U and Z coordinate the simulation. Let us suppose that in a derivation step with sentential form x rule $(C_i \to \alpha_i, Q_i, R_i)$ is used. The simulation in the ETEG system is as follows.

In the first step the environment applies table $T_{(i)}$ to rewrite xU into $[x, i][U, i]$ or into $[x, i][\widehat{U}, i]$ depending on whether or not $Q_i = \emptyset$; the agent rewrites Z into Z'. This is the only role of Z: it allows the agent to work during the first step of the simulation.

In the second step the agent applies the rule $[U, i] \to [U', i]$ if $Q_i = \emptyset$ or applies the rule $[A, i] \to [A', i]$ if $Q_i = \{A\}$. The environment rewrites the remaining letters by using table $T'_{(i)}$.

In the third simulation step the agent applies its rule corresponding to the rule of G, namely rule $[C'_i, i] \to \alpha_i$, while the environment rewrites the remaining letters by using table $T''_{(i)}$. During this last step, the environment can delete the special symbols Z' and $[U', i]$, thus allowing the possibility of finishing the derivation if the sentential form would be a terminal word.

Now we have showed that we can simulate the derivation steps of the random-context grammar. It follows from the construction of the simulating $ETEG$ system that the behaviour described above is the only one which can result in a terminal word. The only possibility to start a derivation from a word over $V \cup \{U, Z\}$ is to use one of the tables $T_{(i)}$ and the rule $Z \to Z'$ of the agent. If the sentential form contains some forbidding letters from R_i, the environment blocks the derivation in the next step by introducing a D; if the permitting symbol referring to the non-empty set Q_i does not appear in the sentential form, the derivation is blocked because the agent cannot work. (It cannot use the other rule $[U, i] \to [U', i]$, because this symbol appears in the sentential form iff $Q_i = \emptyset$.) In the next step the agent has to use the rule $[C'_i, i] \to \alpha_i$ and the environment has to use table $T''_{(i)}$. These three consecutive steps simulate the application of one of the rules of P. □

Using the fact that $\mathcal{RC}^{\lambda}_{ac} = RE$ and the fact that we can construct a Turing machine simulating an extended tabled simple eco-grammar system, we obtain the following theorem:

Theorem 3.1 $\mathcal{L}(\mathcal{ETEG}, \infty) = \mathcal{RE}$

4 Weak extended simple eco-grammar systems

In this section another variant of extended simple eco-grammar systems is studied: the weak extended simple eco-grammar system. This variant has the same compo-

nents as the extended simple eco-grammar system but it works in a different way. Informally speaking, in a weak system the derivation is not blocked if there are some agents which cannot perform any action.

Compared to [11], the definition of a weak rewriting step is slightly different. There, in a weak rewriting step, the derivation is blocked if no agent can work, whereas here we allow this possibility.

Definition 4.1 *A weak extended simple eco-grammar system (a wEEG system, for short) is a construct* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$, *where*

- *all the components are the same as in Definition 2.1.*

In a weak extended simple eco-grammar system $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$ x *directly derives* y *(with* $x \in V_E^+$ *and* $y \in V_E^*$, *written as* $x \overset{w-eco}{\Longrightarrow}_\Sigma y$*) if*

- $x = x_1 Z_1 x_2 Z_2 \cdots x_k Z_k x_{k+1}$, *with* $Z_i \in V_E$, $x_j \in V_E^*$, $0 \leq k \leq n$, $1 \leq i \leq k$, *and* $1 \leq j \leq k+1$,
- $y = y_1 w_1 y_2 w_2 \cdots y_k w_k y_{k+1}$, *with* $y_i, w_j \in V_E^*$, $0 \leq k \leq n$, $1 \leq i \leq k$, *and* $1 \leq j \leq k+1$,
- *there exists a permutation of some agents, namely* $R_{j_1}, R_{j_2}, \ldots, R_{j_k}$, *such that* $Z_i \rightarrow w_i \in R_{j_i}$, *for* $1 \leq i \leq k$,
- $\{dom(R_t) \mid 1 \leq t \leq n, t \neq j_i, \text{ for } 1 \leq i \leq k\} \cap alph(x_1 x_2 \cdots x_{k+1}) = \emptyset$, *and*
- $x_i = \lambda$ *or* $x_i \overset{0L}{\Longrightarrow}_E y_i$, *for* $1 \leq i \leq k+1$, *where* $E = (V_E, P_E, \omega)$ *is the* $0L$ *system of the environment.*

We denote by $\overset{w-eco}{\Longrightarrow}{}^*_\Sigma$ the transitive and reflexive closure of $\overset{w-eco}{\Longrightarrow}_\Sigma$.

That is, in a weak extended simple eco-grammar system we choose some agents to perform a common action in the following way: the chosen agents can perform an action together and there is no symbol among the remaining letters where any of the other agents could act. The chosen agents perform their actions and the remaining letters are rewritten by the environment. In the particular case when there is only one agent in the system, this definition implies that the agent has to work if it is able to but if no letter can be rewritten by the agent it is the environment itself that continues the derivation.

Definition 4.2 *For a weak extended simple eco-grammar system* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$ *the generated language is the following:*

$$L(\Sigma) = \{ v \in \Delta^* \mid \omega \overset{w-eco}{\Longrightarrow}{}^*_\Sigma v \}.$$

We denote by $wEEG(n)$ the class of languages which can be generated by weak extended simple eco-grammar systems with n agents.

In the following we show that $wEEG$ systems can generate all recursively enumerable languages. The result is based on the following lemma.

Lemma 4.1 $\mathcal{RC}_{ac}^{\lambda} \subseteq wEEG(1)$

Proof For a random-context grammar $G = (N, T, P, S)$ we will give a weak extended simple eco-grammar system $\Sigma = (V_E, P_E, R_1, \Delta, \omega)$ such that $L(G) = L(\Sigma)$.

First we present the definition of this system Σ and explain its functioning. Similar to Lemma 3.1, the notation V stands for $N \cup T$, r denotes the number of rules in P, the rules in P are enumerated as $p_i = (C_i \to \alpha_i, Q_i, R_i)$, we assume there are no rules with $Q = R$, $Q = \{C\}$ or $R = \{C\}$, and we refer to the components of the ith rule as C_i, α_i, Q_i, and R_i.

Let

- $V_E = V \cup \{[X,i,j] \mid X \in V, 1 \le i \le r, 1 \le j \le 5\}$
  $\cup \{[Z,i,j], [\widehat{Z},i,k] \mid 1 \le i \le r, 1 \le j \le 5, 1 \le k \le 4\}$
  $\cup \{[\widehat{C}_i,i,k] \mid 1 \le i \le r, 1 \le k \le 4\}$
  $\cup \{[U,i,j] \mid 1 \le i \le r, Q_i = \emptyset, 1 \le j \le 2\}$
  $\cup \{D\}$
  $\cup \{[D,i,3] \mid 1 \le i \le r\}$ where $U, D, Z \notin V$,

- $P_E = \{[X,i-1,5] \to [X,i,1] \mid 2 \le i \le r, X \in V \cup \{Z\}\}$
  $\cup \{[X,r,5] \to [X,1,1] \mid X \in V \cup \{Z\}\}$

  $\cup \{[X,i,1] \to [X,i,2] \mid 1 \le i \le r, X \in V \cup \{Z, \widehat{Z}\}\}$
  $\cup \{[\widehat{C}_i,i,1] \to [\widehat{C}_i,i,2] \mid 1 \le i \le r\}$
  $\cup \{[U,i,1] \to [U,i,2] \mid 1 \le i \le r, Q_i = \emptyset\}$

  $\cup \{[X,i,2] \to [X,i,3] \mid 1 \le i \le r, X \in V \cup \{Z, \widehat{Z}\}\}$
  $\cup \{[\widehat{C}_i,i,2] \to [\widehat{C}_i,i,3] \mid 1 \le i \le r\}$
  $\cup \{[U,i,2] \to D \mid 1 \le i \le r, Q_i = \emptyset\}$

  $\cup \{[X,i,3] \to [X,i,4] \mid 1 \le i \le r, X \in V \cup \{Z, \widehat{Z}\}\}$
  $\cup \{[\widehat{C}_i,i,3] \to [\widehat{C}_i,i,4] \mid 1 \le i \le r\}$
  $\cup \{[D,i,3] \to D \mid 1 \le i \le r\}$

  $\cup \{[X,i,4] \to [X,i,5] \mid 1 \le i \le r, X \in V \cup \{Z\}\}$
  $\cup \{[\widehat{Z},i,4] \to [Z,i,5] \mid 1 \le i \le r\}$
  $\cup \{[\widehat{C}_i,i,4] \to [C_i,i,5] \mid 1 \le i \le r\}$

  $\cup \{[X,i,5] \to X \mid 1 \le i \le r, X \in V \cup \{Z\}\}$
  $\cup \{X \to D \mid X \in V \cup \{D\}\}$,

- $R_1 = \{[Z,i-1,5] \to [\widehat{Z},i,1] \mid 2 \le i \le r\}$
  $\cup \{[Z,r,5] \to [\widehat{Z},1,1]\}$
  $\cup \{[C_i,i-1,5] \to [\widehat{C}_i,i,1] \mid 2 \le i \le r, Q_i \ne \emptyset\}$
  $\cup \{[C_1,r,5] \to [\widehat{C}_1,1,1] \mid Q_1 \ne \emptyset\}$
  $\cup \{[C_i,i-1,5] \to [\widehat{C}_i,i,1][U,i,1] \mid 2 \le i \le r, Q_i = \emptyset\}$
  $\cup \{[C_1,r,5] \to [\widehat{C}_1,1,1][U,1,1] \mid Q_1 = \emptyset\}$

  $\cup \{[\widehat{Z},i,1] \to [\widehat{Z},i,2] \mid 1 \le i \le r\}$

$$\cup \{[B,i,1]\to D \mid 1 \le i \le r, \{B\} = R_i \neq \emptyset\}$$
$$\cup \{[A,i,2]\to[A,i,3][D,i,3] \mid 1 \le i \le r, \{A\} = Q_i \neq \emptyset\}$$
$$\cup \{[U,i,2]\to[D,i,3] \mid 1 \le i \le r, Q_i = \emptyset\}$$
$$\cup \{[\widehat{Z},i,2]\to[\widehat{Z},i,3] \mid 1 \le i \le r\}$$
$$\cup \{[D,i,3]\to\lambda\}$$
$$\cup \{[\widehat{C}_i,i,3]\to D \mid 1 \le i \le r,\}$$
$$\cup \{[\widehat{C}_i,i,4]\to[\alpha_i,i,5] \mid 1 \le i \le r, \alpha_i \neq \lambda\}$$
$$\cup \{[\widehat{C}_i,i,4]\to\lambda \mid 1 \le i \le r, \alpha_i = \lambda\}$$
$$\cup \{[Z,i,5]\to\lambda \mid 1 \le i \le r\},$$

- $\Delta = T$, and
- $\omega = [S,r,5][Z,r,5]$.

The main point of the simulation is that we simulate the application of the rules in their order from 1 to r, each time either simulating the rule or skipping the rule. After having simulated or skipped the rth rule we continue with the first one.
We do the simulation of a rule by introducing five different alphabets for each rule of G: for the ith rule we introduce the alphabets $[V,i,j]$ for $1 \le j \le 5$. We start the simulation or the skipping of the ith rule with a word over the alphabet $[V,i-1,5]$, then during the simulation we go through the alphabets $[V,i,j]$ for $1 \le j \le 4$, and finish with a word over the alphabet $[V,i,5]$. Consequently we can finish the whole derivation or we can continue with the next the rule.

There are more additional alphabets for coordinating the simulation: the letters $[Z,i,j]$ and $[\widehat{Z},i,k]$ for $1 \le i \le r$, $1 \le j \le 5$, and $1 \le k \le 4$ make it possible to skip the ith rule of G; the symbols $[\widehat{C}_i,i,j]$ let the agent simulate the ith rule of G; the symbols $[U,i,j]$ are introduced only if $Q_i = \emptyset$ and make it possible to deal with this case; the symbols $[D,i,3]$ ensure that the derivation is blocked if the agent simulates the ith rule of G while the non-empty permitting condition is missing.

In the following, we first show how the application of a rule of G can be simulated and we also show how the application can be skipped. Then we show why the construction of the above $wEEG$ system guarantees that only those derivations that follow a derivation of the random-context grammar G result in a terminal word.

Let us suppose that we want to simulate the application of the first rule of G: $(C_1\to\alpha_1, Q_1, R_1)$ (the case of the other rules is similar) and let us first suppose that $Q_1 \neq \emptyset$. Before the simulation the sentential form in Σ is over $\{[W,r,5] \mid W \in V \cup \{Z\}\}$.

In the first step the agent "decides" whether the current rule (in this case the first rule of G) will be simulated or will be skipped. Let us suppose that the rule is to be simulated. In this case the agent uses the rule $[C_1,r,5]\to[\widehat{C}_1,1,1]$. The other letters are rewritten by the environment, using the rules $\{[X,r,5]\to[X,1,1] \mid X \in V \cup \{Z\}\}$.

In the next step the agent checks whether or not the forbidding context is present in the sentential form. This is done in the following way: the agent introduces a D if $[B,1,1]$ is present (where $\{B\} = R_1$), while otherwise the agent does not work because $[\widehat{Z},1,1]$ is not present in the sentential form. The environment increases the second index of the symbols from 1 to 2 in this step.

In the third step the agent uses its rule $[A,1,2]\rightarrow[A,1,3][D,1,3]$ for $\{A\} = Q_1$; the environment increases the second indices from 2 to 3 in the other symbols.

In the fourth step the agent deletes $[D,1,3]$ while the environment increases the second indices from 3 to 4. In the fifth and final step the agent applies the rule $[\widehat{C}_1,1,4]\rightarrow[\alpha_1,1,5]$ or the rule $[\widehat{C}_1,1,4]\rightarrow\lambda$, which correspond to the first rule of G; the environment increases the second indices. Therefore we obtain a word over $\{[W,1,5] \mid W \in V \cup \{Z\}\}$.

If $Q_1 = \emptyset$, that is when the permitting condition is empty, the simulation is different. While the environment does the same as in the previous case, the agent applies different rules. The rule the agent uses in the first step is $[C_1,r,5]\rightarrow[\widehat{C}_1,1,1][U,1,1]$ and thus $[U,1,1]$ is introduced. In the third step this symbol is used to introduce $[D,1,3]$ and from that point the simulation continues in the same way as described above, that is when $Q_1 \neq \emptyset$.

Now we show how we can do the skipping of the first rule (the case of the other rules is the same). Let us suppose again that we have a word over the alphabet $\{[W,r,5] \mid W \in V \cup \{Z\}\}$.

The environment works in the same way as it did in the previous case, the only difference is in the behaviour of the agent. In the first step the agent chooses the rule $[Z,r,5]\rightarrow[\widehat{Z},1,1]$, in the next step the rule $[\widehat{Z},1,1]\rightarrow[\widehat{Z},1,2]$, and in the third step the rule $[\widehat{Z},1,2]\rightarrow[\widehat{Z},1,3]$. In the fourth and the fifth step the agent no longer has any rule to apply, hence it does not perform any action. By the end of these five steps we have the same word as we had before, apart from the first indices in the symbols: we have the same word over the alphabet $\{[W,1,5] \mid W \in V \cup \{Z\}\}$.

At this point the simulation or the skipping of the second rule can start and can be carried out in the previous manner. We can continue this process until the last rule, the rth one, when we can restart the whole procedure with the first rule again.

In order to finish the derivation, after having finished the simulation of a rule of G the agent chooses the rule of the form of $[Z,i,5]\rightarrow\lambda$ while the environment rewrites the remaining letters according to its rules $[X,i,5]\rightarrow X$.

Thus we have seen that $L(G) \subseteq L(\Sigma)$.

In the following we show that the eco-grammar system must follow one of the sequences of steps presented above, or otherwise the derivation would never terminate.

In the first step, when the sentential form is over $[W,i-1,5]$, the agent can work because either the left-hand side of the current rule of G is present (and thus the agent can rewrite $[C_i,i-1,5]$) or the symbol $[Z,i-1,5]$ can be rewritten. (At the end of the proof we explain why we can suppose that Z has not yet disappeared from the sentential form.)

Therefore, in this first step the agent marks a place where it can perform the application of the current rule or it can mark Z. If it marks a place for the current rule in the next steps it must check the appearance of the forbidding and the permitting context. The derivation can result in a word not containing letters D only if the check is successful. This is done in the following way: the derivation is blocked by the rule $[B, i, 1] \rightarrow D$ if the forbidding context is present, or by the rule $[\widehat{C}_i, i, 3] \rightarrow D$ if the non-empty permitting condition is missing. In the last step the agent must apply the rule corresponding to the rule of G.

Thus, we have seen that if the agent decides to mark a place for applying the current rule, then he must check whether or not the rule is applicable, and he must simulate it during the five steps. If the agent chooses the other possibility and marks Z, then in the next two steps he must increase the second index of $[\widehat{Z}, i, j]$ from 1 to 2 and from 2 to 3. In the next two steps the agent cannot work. Hence if the agent chooses to mark $[Z, i, 5]$, then the work of the whole system follows the strategy of skipping the current rule, or otherwise the derivation would be blocked.

As far as the end of the derivation is concerned, the environment has to apply the rules of the form $[X, i, 5] \rightarrow X$ for all the letters in the same derivation step, or otherwise the derivation is blocked in the next step. It can happen that the agent deletes Z before the end of the derivation but this fact does not allow any new word to be generated, so we can safely assume that the deletion of Z happens in the same derivation step as the rewritings $[X, i, 5] \rightarrow X$.

We have seen the other direction of the inclusion, $L(\Sigma) \subseteq L(G)$, which completes the proof of the lemma. □

Because $\mathcal{RC}_{ac}^{\lambda} = RE$ and because weak extended simple eco-grammar systems can be simulated by Turing machines, we obtain the following theorem:

Theorem 4.1 $\mathcal{L}(\sqsupseteq\mathcal{EEG}, \infty) = \mathcal{RE}$

5 Conclusions

In this article we presented two variants of extended simple eco-grammar systems. In both cases we have found that the modifications lead to systems withlarge generative power: all recursively enumerable languages can be obtained in these ways with only one agent.

The question of the generative power of the original model, the extended simple eco-grammar system, remains open.

Acknowledgement

The author expresses her thanks to two anonymous referees who suggested a series of improvements of the article.

References

[1] J. Csima. On extended simple eco-grammar systems. *Acta Cybernetica*, 13(4):359–373, 1998.

[2] E. Csuhaj-Varjú, J. Kelemen, A. Kelemenová, and Gh. Păun. Eco-Grammar Systems: A Grammatical Framework for Studying Lifelike Interactions. *Artificial Life*, 3:1–28, 1997.

[3] E. Csuhaj-Varjú and A. Kelemenová. Team Behaviour in Eco-Grammar Systems. *Theoretical Computer Science*, (209):213–224, 1998.

[4] E. Csuhaj-Varjú, Gh. Păun, and A. Salomaa. Conditional Tabled Eco-Grammar Systems. In Gh. Păun, editor, *Artificial Life: grammatical models*, pages 227–239. Black Sea Univ. Press, Bucharest, 1995.

[5] E. Csuhaj-Varjú, Gh. Păun, and A. Salomaa. Conditional Tabled Eco-Grammar Systems versus (E)T0L Systems. *Journal of Universal Computer Science*, 5(1):252–268, 1995.

[6] J. Dassow and V. Mihalache. Eco-Grammar Systems, Matrix Grammars and E0L Systems. In Gh. Păun, editor, *Artificial Life: grammatical models*, pages 210–226. Black Sea Univ. Press, Bucharest, 1995.

[7] J. Dassow and Gh. Păun. *Regulated Rewriting in Formal Language Theory.* Springer-Verlag, 1989.

[8] Grzegorz Rozenberg and Arto Salomaa. *The Mathematical Theory of L-systems.* Academic Press, 1980.

[9] A. Salomaa. *Formal languages.* Academic Press, 1973.

[10] P. Sosík. Eco-Grammar Systems, Decidability and the Tiling Problem. In A. Kelemenová, editor, *Proceedings of the MFCS'98 Satellite Workshop on Grammar Systems*, pages 195–213. Silesian University, Opava, 1998.

[11] M. H. ter Beek. Simple Eco-Grammar Systems with Prescribed Teams. In Gh. Păun and A. Salomaa, editors, *Grammatical Models of Multi-Agent Systems*, pages 113–135. Gordon and Breach Science Publishers, 1999.

Received December, 1999