On extended simple eco-grammar systems [*]

Judit CSIMA[†]

Abstract

In this contribution extended simple eco-grammar systems are studied. A simple eco-grammar system is formed from an environment given by a set of 0L rules and from some agents represented by sets of CF rules. In an extended simple eco-grammar system we distinguish a subset of the alphabet of the system and only strings over this subalphabet are in the generated language. The relations between language classes generated with different parameters and derivation modes are investigated.

1 Introduction

The concept of the eco-grammar system (the EG system, for short) has been introduced in [2] as a model of communities of agents which interact with their common shared environment. Several aspects of these systems were discussed in [3] and [6], properties of a restricted variant, called simple eco-grammar system, were studied in [4], [1], [10], [5], and [9]. Briefly, a simple eco-grammar system consists of several agents (represented by sets of context-free rules) and an environment (given by a set of $0L$ rules). At any moment of time, the behaviour of the system is described by the state of the environment which is a string over the alphabet of the system. The environmental state changes by derivation steps. In a derivation step the agents act on the string by applying one of their context-free rules - each agent rewrites only one letter - and the environment replaces, according to its $0L$ rule set, in a parallel manner the symbols where the agents do not perform any action.

Starting from an initial string representing the environment, a lot of strings following each other arise, which describe the evolving system. The language generated by the eco-grammar system is the set of strings which can be obtained from the initial environmental state by a sequence of derivation steps. In the case of extended simple eco-grammar systems only those strings belong to the determined language which are over a distinguished subset of the alphabet of the system, the terminal alphabet. This notion was introduced in [5] under the name of 0-terminal EG system and some basic properties of these systems were examined as well. It

*Research supported by the Hungarian Scientific Foundation "OTKA" Grant No. T017105. Presented at the Conference of PhD Students on Computer Science, July 18-22 1998, Szeged.

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

was shown there that λ-free extended simple eco-grammar systems with one agent are as powerful as λ-free extended simple EG systems with at most n agents, if the original mode of the derivation is used.

Following this line, in this contribution we deal with a more sophisticated version of the derivation, the team behaviour of the extended simple eco-grammar systems: in each derivation step from the n agents exactly k or at most k are allowed to work. We describe the behaviour and the generative power of these systems according to some size parameters: the total number of the agents, the number of the agents being active in a derivation step. We examine the hierarchy of the language classes generated by extended simple eco-grammar systems with and without λ-rules.

The results demonstrate that while in the non-extended case the size parameters of the teams and the agent population have influence on the power of the system, in the extended case these parameters are not important: we obtain a collapsing hierarchy.

2 Preliminaries and the definition of the extended simple eco-grammar system

In this section we present the basic notions and notations used in the paper, for further information the reader is referred to [8] and [7].

An alphabet is a non-empty set of symbols. The set of all non-empty words over a finite alphabet V is denoted by V^+, the empty word is denoted by λ. The set V^* is $V^+ \cup \{\lambda\}$.

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 $a \rightarrow u$, where $a \in V$ and $u \in V^*$. A CF rule is a λ-rule (or an erasing rule) if $u = \lambda$.

A $0L$ system is a construct $H = (V, P, \omega)$, where V is a finite alphabet, P is a finite 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 on 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 $x \Rightarrow_H y$, if $x = x_1 x_2 \dots x_n$, $y = y_1 y_2 \dots y_n$, $x_i \in V$, $y_i \in V^*$ and the rules $x_i \rightarrow y_i$ are in P for $1 \leq i \leq n$.
The generated language of a $0L$ system H (denoted by $L(H)$) is the set of the words over V which can be derived (in some steps) from the axiom.

Throughout the paper, we use the customary notations: $\subseteq$ denotes inclusion and $\subset$ denotes strict inclusion.

If L is a language, we call *alphL* the set of all letters which occur in the words of L.

If V is an alphabet, we will use the following notations:

- $V^{(k)} = \{ A^{(k)} \mid A \in V \}$, where k is a positive integer,
- $V' = \{ A' \mid A \in V \}$,

- $\overline{V} = \{ \overline{A} \mid A \in V \}$.

If u is a word of the form $u = x_1 \ldots x_n$, $x_i \in V$, $1 \leq i \leq n$, then $u^{(k)} = x_1{}^{(k)} \ldots x_n{}^{(k)}$ and $u' = x_1{}' \ldots x_n{}'$.

After these basic notions we present the definition of the extended simple eco-grammar system.

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

In this construct V_E is the alphabet and P_E is the set of the evolution rules of the environment. R_i represents the ith agent, this is the set of its 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 from which all the derivations start. Δ is the terminal set, the language of the *EG* system consists of words over Δ.

The system changes its state by a simultaneous action of some agents and by a parallel rewriting according to P_E. In this contribution we consider two types of derivations, first we present the definition of derivation mode $= k$.

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 *in mode* $= k$ ($x \in V_E{}^+, y \in V_E{}^*$ *and* $1 \leq k \leq n$), *written as* $x \overset{=k}{\Longrightarrow}_\Sigma y$, *if*

- $x = x_1 Z_1 x_2 Z_2 \ldots x_k Z_k x_{k+1}$, $Z_i \in V_E$, $x_j \in V_E{}^*$,
- $y = y_1 w_1 y_2 w_2 \ldots y_k w_k y_{k+1}$, $y_i, w_j \in V_E{}^*$,
- *there exist* k *different agents in* Σ, *namely* $R_{j_1}, R_{j_2}, \ldots, R_{j_k}$, *such that* $Z_i \to w_i \in R_{j_i}$ *for* $1 \leq i \leq k$, *and*
- $x_i = y_i = \lambda$ *or* $x_i \Rightarrow_E y_i$, *where* $E = (V_E, P_E, \omega)$ *is the 0L system of the environment.*

In the above case we say that $x \overset{=k}{\Longrightarrow}_\Sigma y$ is a derivation step.

Another mode of the derivation is derivation mode $\leq k$:

Definition 2.3
Let $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$ be an extended simple eco-grammar system. We say that x directly derives y in Σ by a derivation $\leq k$ $(1 \leq k \leq n)$, written as $x \overset{\leq k}{\Longrightarrow}_{\Sigma} y$, if $x \overset{=l}{\Longrightarrow}_{\Sigma} y$ for some l, $1 \leq l \leq k$.

We denote the transitive and reflexive closure of $\overset{=k}{\Longrightarrow}_{\Sigma}$ and $\overset{\leq k}{\Longrightarrow}_{\Sigma}$ by $\overset{=k}{\Longrightarrow}{}^*_{\Sigma}$ and $\overset{\leq k}{\Longrightarrow}{}^*_{\Sigma}$. In both cases, the generated language consists of the words which can be generated from the axiom in some derivation steps and which are over Δ.

Definition 2.4
Consider an extended simple eco-grammar system $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$. The generated languages in mode $= k$ and $\leq k$ are the following:

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

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

Now we present an example.

Example 2.1 *Let $\Sigma = (V_E, P_E, R_1, R_2, R_3, \omega, \Delta)$ be the following extended simple EG system:*

- $V_E = \{ A, B, C, a, b, c \}$,
- $P_E = \{ A \to a, B \to b, C \to c, a \to a, b \to b, c \to c \}$,
- $R_1 = \{ A \to A^2, A \to a \}$,
- $R_2 = \{ B \to B^2, B \to b \}$,
- $R_3 = \{ C \to C^2, C \to c \}$,
- $\omega = ABC$,
- $\Delta = \{ a, b, c \}$.

It is easy to see that in mode $= 3$ a derivation sequence is of the following form. (In order to simplify the writing, when a rule can be applied in several places we use the left-most one; the other derivations give the same word.)

$$ABC \Rightarrow A^2B^2C^2 \Rightarrow A^2aB^2bC^2c \Rightarrow \ldots \Rightarrow A^2a^{k-2}B^2b^{k-2}C^2c^{k-2} \Rightarrow a^kb^kc^k$$

The three agents have to finish the derivation at the same step, otherwise the derivation would be blocked. (All of the three agents have to be active in each step, if there are at most two different upper case letters in the sentential form the derivation is blocked.)

Thus the generated language is $L(\Sigma, = 3) = \{ a^n b^n c^n \mid n \geq 1 \}$.

Finally, we present some notations which we will use in the paper.
We consider extended simple eco-grammar systems with or without λ rules. When we allow λ-rules in P_E and in the sets R_i, we use the following notations: the class of the languages which can be generated by an extended simple eco-grammar system is denoted by $\mathcal{L}(EE)^\lambda$ (in this notation the first E means "extended", the second one refers to "eco"). The class of the languages generated by a system containing n agents and operating in mode $= k$ ($\leq k$) is denoted by $\mathcal{L}(EE^\lambda(n, = k))$ ($\mathcal{L}(EE^\lambda(n, \leq k))$).
We omit the notation λ if λ-rules are not allowed neither in P_E nor in the sets R_i: $\mathcal{L}(EE)$, $\mathcal{L}(EE(n, = k))$ and $\mathcal{L}(EE(n, \leq k))$. In the statements of the paper we use the notation (λ) if a statement is true both with and without λ-rules.

3 Relations between the language classes generated by extended simple EG systems

3.1 The hierarchy of language classes $\mathcal{L}(EE^{(\lambda)}(n, = k))$

In this section we are going to present results about the hierarchy of the language classes generated by n agents in derivation mode $= k$. This question was examined for the non-extended simple eco-grammar systems in [4]. It was proved there that the generated language classes are incomparable in most of the cases. We get very different results for the extended systems: here the language classes form a collapsing hierarchy.

Most of the statements and proofs of the section are true with or without λ-rules, this fact is denoted by the superscript (λ); sometimes a statement is true in both cases, but the proofs are different, in this case we will present the two different proofs together; and sometimes a statement is true only when λ-rules are allowed, in this case we will emhasize this fact.

First we examine the role of the first parameter, the number of the agents.

Lemma 3.1.1 *For* $1 \leq k \leq n \leq m$, $\mathcal{L}(EE^{(\lambda)}(n, = k)) \subseteq \mathcal{L}(EE^{(\lambda)}(m, = k))$.

Proof
We show that for any $1 \leq k \leq n \leq m$ and for any extended simple EG system $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$ there exists another extended simple EG system $\Sigma' = (V_E', P_E', R_1', \ldots, R_m', \omega', \Delta')$ such that $L(\Sigma, = k) = L(\Sigma', = k)$. Moreover, if Σ does not contain λ-rules then Σ' is also λ-free.
Let Σ' be the following:

- $V_E' = V_E \cup \{D\}$, where $D \notin V_E$,
- $P_E' = P_E \cup \{D \rightarrow D\}$,
- $R_i' = R_i$ for $1 \leq i \leq n$,

- ${R_i}' = \{D \rightarrow D\}$ for $n+1 \leq i \leq m$,
- $\omega' = \omega$,
- $\Delta' = \Delta$.

It is clear that $L(\Sigma, = k) = L(\Sigma', = k)$ and Σ' is λ-free if and only if Σ does not contain λ-rules. ■

The following result shows that the above inclusion is not proper. The statement is true with or without λ-rules.

Lemma 3.1.2 *For* $1 \leq k \leq n$, $\mathcal{L}(EE^{(\lambda)}(n+1, = k)) \subseteq \mathcal{L}(EE^{(\lambda)}(n, = k))$.

Proof
The λ-free case
We will show that we can simulate any λ-free extended simple EG system $\Sigma = (\, V_E, P_E, R_1, \ldots, R_{n+1}, \omega, \Delta \,)$ working in mode $= k$ by another λ-free extended simple EG system $\Sigma' = (\, {V_E}', {P_E}', {R_1}', \ldots, {R_n}', \omega', \Delta' \,)$ working also in mode $= k$. Let Σ' be the following system:

- ${V_E}' = {V_E}^{(1)} \cup {V_E}^{(2)} \cup {V_E}' \cup \{D\}$, where $D \notin V_E$,
- ${P_E}' = {P_E}^{(1)\rightarrow(2)} \cup \{\, A^{(2)} \rightarrow A^{(1)} \mid A \in V_E \,\} \cup \{\, A' \rightarrow D \mid A \in V_E \,\} \cup$
  $\cup \{\, D \rightarrow D \,\}$, where ${P_E}^{(1)\rightarrow(2)} = \{\, A^{(1)} \rightarrow v^{(2)} \mid A \rightarrow v \in P_E \,\}$,
- ${R_1}' = {R_1}^{(1)\rightarrow(2)} \cup {R'}_{n+1}^{(1)\rightarrow(2)} \cup \{A' \rightarrow A^{(1)} \mid A \in V_E \,\} \cup$
  $\cup \{A^{(2)} \rightarrow A^{(1)} \mid A \in V_E \,\}$,
- ${R_i}' = {R_i}^{(1)\rightarrow(2)} \cup {R'}_{n+1}^{(1)\rightarrow(2)} \cup \{A^{(2)} \rightarrow A^{(1)} \mid A \in V_E \,\}$ for $2 \leq i \leq n$,
  where
  ${R_j}^{(1)\rightarrow(2)} = \{\, A^{(1)} \rightarrow w^{(2)} \mid A \rightarrow w \in R_j \,\}$ for $1 \leq j \leq n$
  and
  ${R'}_{n+1}^{(1)\rightarrow(2)} = \{\, v^{(1)} \rightarrow w'^{(2)} \mid v \rightarrow w \in R_{n+1} \,\}$, where
  $w'^{(2)} = x_1^{(2)} \ldots x_{m-1}^{(2)} x_m{}'$ if $w = x_1 \ldots x_{m-1} x_m$, $x_i \in V_E$, and $1 \leq i \leq m$,
- $\omega' = \omega^{(1)}$,
- $\Delta' = \Delta^{(1)}$.

The main idea of this construction is the following. We simulate one derivation step of Σ by two derivation steps of Σ'. In the first simulation step we use the rules corresponding to the derivation of Σ, the second step checks if the simulation is correct.
In Σ' the set R_{n+1} is included in the set of rules of all agents in form of ${R'}_{n+1}^{(1)\rightarrow(2)}$. When an agent uses a rule corresponding to R_{n+1}, a primed letter is introduced as well. The only agent which is able to make these primed letters disappear is ${R_1}'$; this construction guarantees (considering the rules of ${P_E}'$) that the derivation

cannot terminate with a terminal word if the agents use more than one rule from ${R'}_{n+1}^{(1)\to(2)}$ at the same derivation step.

If in a derivation step of Σ the agents $R_{i_1}, \ldots, R_{i_k}$ work and $i_j \leq n$ for $1 \leq j \leq k$, the simulation goes as follows.
In the first simulation step in Σ', the agents ${R_{i_1}}', \ldots, {R_{i_k}}'$ work using rules from ${R_{i_j}}^{(1)\to(2)}$, corresponding to the step of Σ. Then the environment uses the same rules as in Σ (of the form ${P_E}^{(1)\to(2)}$).
In the second simulation step the agents ${R_{i_1}}', \ldots, {R_{i_k}}'$ rewrite k letters and the environment rewrites the remaining letters by using the rules of the form $A^{(2)} \to A^{(1)}$.
If R_{n+1} was among the agents $R_{i_1}, \ldots, R_{k-1}, R_k = R_{n+1}$ in a derivation step in Σ, the two simulation steps of Σ' are the following.
In the first simulation step we choose one agent ${R_s}'$ in Σ' such that $s \neq i_j$ for any $1 \leq j \leq k-1$. It is possible because $k \leq n$. This agent will simulate the work of R_{n+1} by using the corresponding rule of ${R'}_{n+1}^{(1)\to(2)}$.
The agents ${R_{i_1}}', \ldots, {R'}_{k-1}$ simulate the remaining rules of the agents and the environment rewrites the remaining letters in the same way as in Σ.
In the second simulation step ${R_1}'$ rewrites the primed letter, other $k-1$ agents rewrite $k-1$ other letters and the environment rewrites the remaining letters into symbols of ${V_E}^{(1)}$.
Since we can simulate every derivation step of Σ by a derivation sequence of Σ', we obtain that $L(\Sigma, = k) \subseteq L(\Sigma', = k)$.

On the other hand, we can simulate by Σ the derivation sequences of Σ' resulting terminal words, which gives the other inclusion $L(\Sigma', = k) \subseteq L(\Sigma, = k)$.
We will simulate two derivation steps of these sequences of Σ' by one step of Σ.
For each $v \in L(\Sigma', = k)$ (which implies $v \in {\Delta^{(1)}}^+$ because λ-rules are not allowed) there exists a derivation sequence in Σ' of the form

$$\omega' \Rightarrow_{\Sigma'} v_1 \Rightarrow_{\Sigma'} \ldots \Rightarrow_{\Sigma'} v_t = v.$$

Since $v \in {{V_E}^{(1)}}^+$, there are neither primed letters nor D in v.
Considering the possible rules in ${P_E}'$, it is easy to see that there can be at most one primed letter in the previous word v_{t-1}.

The fact that there is no primed letter in v_{t-1} means that in the $(t-1)th$ step k agents: ${R_{i_1}}', \ldots {R_{i_k}}'$ used rules from the sets ${R_{i_j}}^{(1)\to(2)}$ with $1 \leq i_j \leq n$. In this case we can simulate the last two derivation steps of Σ' by one derivation step of Σ, using the corresponding rules of the agents $R_{i_1}, \ldots R_{i_k}$ and the environment.
The other possibility gives that $k-1$ agents, ${R_{i_1}}', \ldots, {R_{i_{k-1}}}'$ use rules from the sets ${R_{i_j}}^{(1)\to(2)}$ with $1 \leq i_j \leq n$ and one agent uses a rule of ${R'}_{n+1}^{(1)\to(2)}$.
Then we can simulate the last two steps by one step in Σ when the agents $R_{i_1}, \ldots, R_{i_{k-1}}, R_{n+1}$ and the environment use the corresponding rules.
In both cases $v_{t-2} \in {{V_E}^{(1)}}^+$, thus we can continue the simulation by giving the role of v to v_{t-2}. Since the axiom is over ${V_E}^{(1)}$ and t is an even number, we can simulate the whole derivation sequence.

With λ-rules

The main idea is the same as in the λ-free case, we will show that we can simulate any extended simple EG system $\Sigma = (V_E, P_E, R_1, \ldots, R_{n+1}, \omega, \Delta)$ working in mode $= k$ by another extended simple EG system
$\Sigma' = (V_E', P_E', R_1', \ldots, R_n', \omega', \Delta')$ working also in mode $= k$.
Let Σ' be the following system:

- $V_E' = V_E^{(1)} \cup V_E^{(2)} \cup \{D, T_1, T_2\}$, where $D, T_1, T_2 \notin V_E$,
- $P_E' = P_E^{(1)\to(2)} \cup \{ A^{(2)} \to A^{(1)} \mid A \in V_E \} \cup \{ D \to D, T_1 \to D, T_2 \to \lambda \}$,
- $R_1' = R_1^{(1)\to(2)} T_2 \cup R_{n+1}^{(1)\to(2)} T_1 \cup \{ T_1 \to \lambda \} \cup \{ T_2 \to \lambda \}$,
- $R_i' = R_i^{(1)\to(2)} T_2 \cup R_{n+1}^{(1)\to(2)} T_1 \cup \{ T_2 \to \lambda \}$ for $2 \leq i \leq n$,
  where
  $R_j^{(1)\to(2)} T_2 = \{ A^{(1)} \to w^{(2)} T_2 \mid A \to w \in R_j \}$ for $1 \leq j \leq n$ and
  $R_{n+1}^{(1)\to(2)} T_1 = \{ A^{(1)} \to w^{(2)} T_1 \mid A \to w \in R_{n+1} \}$,
- $\omega' = \omega^{(1)}$,
- $\Delta' = \Delta^{(1)}$.

The proof that $L(\Sigma, = k) = L(\Sigma', = k)$ is very similar to that of the λ-free case, T_1 plays the role of the primed letters, it controls the usage of the rules of $R'^{(1)\to(2)}_{n+1}$, T_2 letters guarantee that k agents can be active in the second simulation step even if in the first simulation step λ-rules were applied. ■

We obtain from the two previous lemmas the following corollary, which means that the first parameter has no influence on the class of the generated languages.

Corollary 3.1.1 *For $1 \leq k \leq n$,* $\mathcal{L}(EE^{(\lambda)}(n, = k)) = \mathcal{L}(EE^{(\lambda)}(k, = k))$.

We have seen that there is not any hierarchy according to the number of agents. Now we turn our attention to the second parameter: the number of the agents working in a step. Considering the above corollary, it is enough to examine the relation of the language classes $\mathcal{L}(EE^{(\lambda)}(k, = k))$.

For the λ-free case, it was proved in [5] that $\mathcal{L}(EE(n+1, = n+1))$ is included in $\mathcal{L}(EE(n, = n))$. In the following lemma we present the same result for extended simple EG systems with λ-rules as well. We give a simulation which is based on the construction used in [5], thus in the first part of our proof we briefly summarize the construction and the explanation used there for the λ-free case. Then in the second part of the proof we give the modifications which are necessary to obtain the same result when λ-rules are allowed.

Lemma 3.1.3 *For $1 \leq n$,* $\mathcal{L}(EE^{(\lambda)}(n+1, = n+1)) \subseteq \mathcal{L}(EE^{(\lambda)}(n, = n))$.

Proof

The λ-free case (from [5]):

For any extended simple EG system $\Sigma = (V_E, P_E, R_1, \ldots, R_{n+1}, \omega, \Delta)$ we give another extended simple EG system $\Sigma' = (V_E', P_E', R_1', \ldots, R_n', \omega', \Delta')$, such that $L(\Sigma, = n+1) = L(\Sigma', = n)$.
Let Σ' be the following:

- $V_E' = V_E \cup V_E' \cup \{\bar{A} \mid A \to \alpha \in R_{n+1}\} \cup \{D\}$, where $D \notin V_E$,
- $P_E' = \{ A \to \alpha' \mid A \to \alpha \in P_E \} \cup \{ A \to \bar{A} \mid A \to \alpha \in R_{n+1} \} \cup$
  $\cup \{ A' \to A \mid A \in V_E \} \cup \{ \bar{A} \to D \mid A \to \alpha \in R_{n+1} \} \cup \{ D \to D \}$,
- $R_1' = \{A \to \alpha' \mid A \to \alpha \in R_1 \} \cup \{\bar{A} \to \alpha \mid A \to \alpha \in R_{n+1} \}$,
- $R_i' = \{A \to \alpha' \mid A \to \alpha \in R_i \} \cup \{A' \to A \mid A \in V_E \}$ for $2 \leq i \leq n$,
- $\omega' = \omega$,
- $\Delta' = \Delta$.

Let the step

$$x = x_1 Z_1 x_2 Z_2 \ldots x_{n+1} Z_{n+1} x_{n+2} \Rightarrow y = y_1 w_1 y_2 w_2 \ldots y_{n+1} w_{n+1} y_{n+2}$$

be a derivation step in Σ, where the rules used by the agents are $Z_i \to w_i$ for $1 \leq i \leq n+1$ and the derivations $x_j \Rightarrow y_j$ for $1 \leq j \leq n+2$ are performed by the environment. We can simulate this step by two steps of Σ' in the following way. (We suppose that R_{n+1} used the rule $Z_{n+1} \to w_{n+1}$.)

$$x = x_1 Z_1 x_2 Z_2 \ldots x_{n+1} Z_{n+1} x_{n+2} \Rightarrow x' = y_1' w_1' y_2' w_2' \ldots y_{n+1}' \overline{Z}_{n+1} y_{n+2}' \Rightarrow$$

$$\Rightarrow y = y_1 w_1 y_2 w_2 \ldots y_{n+1} w_{n+1} y_{n+2}$$

It follows from the construction that the environment cannot introduce two barred letters, otherwise the derivation would never result a terminal word.
Thus, only the sequences consisting of pairs of steps of this type can derive terminal words because otherwise the derivation would never result a terminal word because of the symbol D. (It follows from the construction of Σ').

In the case when λ-rules are allowed we have to modify the above construction in the following way.

- $V_E' = V_E \cup V_E' \cup \{\bar{A} \mid A \to \alpha \in R_{n+1}\} \cup \{D, Z\}$, where $D, Z \notin V_E$,
- $P_E' = \{ A \to \alpha' \mid A \to \alpha \in P_E \} \cup \{ A \to \bar{A} \mid A \to \alpha \in R_{n+1} \} \cup$
  $\cup \{ A' \to A \mid A \in V_E \} \cup \{ \bar{A} \to D \mid A \to \alpha \in R_{n+1} \} \cup$
  $\cup \{ D \to D \} \cup \{ Z \to \lambda \}$,

- $R_1{}' = \{A \to \alpha' Z \mid A \to \alpha \in R_1\} \cup \{\bar{A} \to \alpha \mid A \to \alpha \in R_{n+1}\}$,
- $R_i{}' = \{A \to \alpha' Z \mid A \to \alpha \in R_i\} \cup \{Z \to \lambda\}$ for $2 \leq i \leq n$,
- $\omega' = \omega$,
- $\Delta' = \Delta$.

Here the letters Z guarantee that the agents can act in Σ' in the second simulation step, even if in the first step λ-rules were used. Thus, we have $L(\Sigma, = n+1) = L(\Sigma', = n)$ because of the same reason as in the λ-free case. ■

From the above lemma we obtain the following corollary.

Corollary 3.1.2 *For* $1 \leq n$, $\mathcal{L}(EE^{(\lambda)}(n, = n)) \subseteq \mathcal{L}(EE^{(\lambda)}(1, = 1))$.

There remains only one relation to examine: whether the language classes $\mathcal{L}(EE^{(\lambda)}(n, = n))$ are included in $\mathcal{L}(EE^{(\lambda)}(n+1, = n+1))$ or the inclusion in Lemma 3.1.3 is strict.
The answer depends on the λ-rules. When λ-rules are allowed the two language classes are equal.

Lemma 3.1.4 *For* $1 \leq n$, $\mathcal{L}(EE^{\lambda}(n, = n)) \subseteq \mathcal{L}(EE^{\lambda}(n+1, = n+1))$.

Proof
For any extended simple EG system $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$ we give another extended simple EG system $\Sigma' = (V_E{}', P_E{}', R_1{}', \ldots, R_{n+1}{}', \omega', \Delta')$ containing λ-rules, such that $L(\Sigma, = n) = L(\Sigma', = n+1)$.
Let Σ' be the following:

- $V_E{}' = V_E \cup \{D\}$, where $D \notin V_E$,
- $P_E{}' = P_E \cup \{D \to D\}$,
- $R_i{}' = R_i$ for $1 \leq i \leq n$,
- $R'_{n+1} = \{D \to D, D \to \lambda\}$,
- $\omega' = \omega D$,
- $\Delta' = \Delta$.

For any $v \in L(\Sigma, = n)$ there is a derivation in Σ of the form

$$\omega \Rightarrow v_1 \Rightarrow \ldots \Rightarrow v_t = v$$

where in the ith step ($1 \leq i \leq t$) the agents R_j use the rules r_{j_i}. Let us consider the following derivation in Σ'

$$\omega D \Rightarrow v_1 D \Rightarrow \ldots \Rightarrow v_{t-1} D \Rightarrow v_t = v$$

where in the ith step $(1 \leq i \leq t-1)$ the agents $R_j{}'$ $(1 \leq j \leq n)$ apply the rules r_{j_i} corresponding to the above derivation and the agent R'_{n+1} uses the rule $D \rightarrow D$; in the last step $R_j{}'$ use the corresponding rules r_{j_t} (for $1 \leq j \leq n$) and R'_{n+1} uses the rule $D \rightarrow \lambda$. Thus, we can simulate Σ by Σ'.

Now we show that $L(\Sigma', = n+1) \subseteq L(\Sigma, = n)$. For any $v \in L(\Sigma', = n+1)$ there is a derivation in Σ' of the form

$$\omega' = \omega D \Rightarrow v_1 \Rightarrow \ldots \Rightarrow v_{t-1} \Rightarrow v_t = v$$

where in each derivation step all the $n+1$ agents work. Since v is over $\Delta' = \Delta$, there are no D letters in it. But in the last step $n+1$ agents must work and the last agent can use only the rules $D \rightarrow D$ or $D \rightarrow \lambda$. Moreover, considering that ω' contains one D and that there is no rule producing new D letters, we get that all words in the above derivation, except v, contain exactly one D letter. We also get that in the first $t-1$ step the $(n+1)$th agent has to use the rule $D \rightarrow D$ and it has to use the rule $D \rightarrow \lambda$ in the last step.
But in this case we can simulate this derivation of Σ' by the following derivation in Σ:

$$\omega = f(\omega') \Rightarrow f(v_1) \Rightarrow \ldots \Rightarrow f(v_{t-1}) \Rightarrow f(v_t) = v_t = v$$

where f is a homomorphism over $V_E \cup \{D\}$ defined as follows:

$$f(z) = \begin{cases} z & \text{if } z \in V_E \\ \lambda & \text{if } z = D \end{cases}$$

In the above derivation the agents $R_1, R_2, \ldots, R_n$ use the rules corresponding to the derivation of Σ'. ∎

We have the following corollary.

Corollary 3.1.3 *For* $1 \leq n$, $\mathcal{L}(EE^\lambda(1, = 1)) \subseteq \mathcal{L}(EE^\lambda(n, = n))$.

In the λ-free case the inclusion is proper between the language classes $\mathcal{L}(EE(n, = n))$ and $\mathcal{L}(EE(n+1, = n+1))$.

Lemma 3.1.5 *For any* $1 \leq n$ *there exists a language* L, *such that* $L \in \mathcal{L}(EE(n, = n)) \setminus \mathcal{L}(EE(n+1, = n+1))$.

Proof Let L be the finite language $\{ a^n, b^n \}$. It is clear that it can be generated by an extended simple EG system working in mode $= n$. Moreover, it cannot be generated by using mode $= l$ if $l > n$ because either the letters in the axiom are not enough to perform an action of l agents or we should use λ-rules which is not allowed. ∎

Concluding the investigation of the derivation mode $= k$, from Corollary 3.1.1, Corollary 3.1.2 and Corollary 3.1.3 we obtain the following theorem. It shows that if λ-rules are allowed, neither the number of the agents being in the system nor the number of the agents working in a step have any influence on the generated language class.

Theorem 3.1 *For $1 \leq k \leq n$,* $\mathcal{L}(EE^{\lambda}(n, = k)) = \mathcal{L}(EE^{\lambda}(1, = 1))$.

In the λ-free case we use the notation $\mathcal{L}(EE(= k))$ for the class of the languages which can be generated by extended simple EG systems working in mode $= k$, without λ-rules. From Corollary 3.1.1 we have $\mathcal{L}(EE(n, = k)) = \mathcal{L}(EE(= k))$ for $1 \leq k \leq n$.
Using Corollary 3.1.1, Lemma 3.1.3 and Lemma 3.1.5, we can summarize the results of the λ-free case in the following theorem.

Theorem 3.2
$\mathcal{L}(EE(= 1)) = \mathcal{L}(EE(n_1, = 1)) \supset \mathcal{L}(EE(= 2)) = \mathcal{L}(EE(n_2, = 2)) \supset \ldots \supset \mathcal{L}(EE(= k)) = \mathcal{L}(EE(n_k, = k)) \supset \ldots$
where $n_i \geq i$ for $1 \leq i$.

3.2 The hierarchy of language classes $\mathcal{L}(EE^{(\lambda)}(n, \leq k))$

In this section we examine the hierarchy of the language classes generated by using derivation mode $\leq k$. In most of the cases the statements and proofs of the case $= k$ are valid for the derivation mode $\leq k$ as well, in such cases we refer to the previous section.

The first lemma can be proved by the same construction as in Lemma 3.1.1.

Lemma 3.2.1 *For $1 \leq k \leq n \leq m$,* $\mathcal{L}(EE^{(\lambda)}(n, \leq k)) \subseteq \mathcal{L}(EE^{(\lambda)}(m, \leq k))$.

The proof of the second lemma uses the same construction and follows the same idea as the proof of Lemma 3.1.2, thus we do not give the detailed proof here.

Lemma 3.2.2 *For $1 \leq k \leq n$,* $\mathcal{L}(EE^{(\lambda)}(n + 1, \leq k)) \subseteq \mathcal{L}(EE^{(\lambda)}(n, \leq k))$.

From these two above lemmas we obtain the following corollary:

Corollary 3.2.1
For $1 \leq k \leq n$, $\mathcal{L}(EE^{(\lambda)}(n, \leq k)) = \mathcal{L}(EE^{(\lambda)}(k, \leq k))$.

Thus, it is enough, similarly to the case $= k$, to examine the relation of the language classes $\mathcal{L}(EE^{(\lambda)}(k, \leq k))$

We have the same result as in the case of the derivation mode $= k$. For proving this statement, we can use the same construction and explanation as in Lemma 3.1.3.

Lemma 3.2.3 *For $1 \leq n$,* $\mathcal{L}(EE^{(\lambda)}(n + 1, \leq n + 1)) \subseteq \mathcal{L}(EE^{(\lambda)}(n, \leq n))$.

The above lemmas give the following corollary.

Corollary 3.2.2 *For $1 \leq n$,* $\mathcal{L}(EE^{(\lambda)}(n, \leq n)) \subseteq \mathcal{L}(EE^{(\lambda)}(1, \leq 1))$.

In the case of the derivation mode $= k$ the language classes formed a collapsing hierarchy if and only if λ-rules were allowed in the system. If we consider the derivation mode $\leq k$, we have different results.

Lemma 3.2.4 *For* $1 \leq k \leq n$, $\mathcal{L}(EE^{(\lambda)}(n, \leq n)) \subseteq \mathcal{L}(EE^{(\lambda)}(n+1, \leq n+1))$.

Proof
For any extended simple EG system $\Sigma = (V_E, P_E, R_1, \dots, R_n, \omega, \Delta)$ we give another extended simple EG system $\Sigma' = (V_E', P_E', R_1', \dots, R_{n+1}', \omega', \Delta')$, such that $L(\Sigma, \leq n) = L(\Sigma', \leq n+1)$. Let Σ' be the following:

- $V_E' = V_E \cup \{D\}$, where $D \notin V_E$,
- $P_E' = P_E \cup \{D \rightarrow D\}$,
- $R_i' = R_i$ for $1 \leq i \leq n$,
- $R_{n+1}' = \{D \rightarrow D\}$,
- $\omega' = \omega$,
- $\Delta' = \Delta$.

The generated language of Σ in mode $\leq n$ is the same as that of Σ' in mode $\leq n+1$ because in Σ' an $\leq n+1$ derivation means an $\leq n$ derivation, considering the structure of Σ'. ■

From the above lemma we have the following corollary.

Corollary 3.2.3 *For* $1 \leq n$, $\mathcal{L}(EE^{(\lambda)}(1, \leq 1)) \subseteq \mathcal{L}(EE^{(\lambda)}(n, \leq n))$

Concluding the investigation of the case $\leq k$, from Corollary 3.2.1, Corollary 3.2.2 and Corollary 3.2.3 we obtain that the hierarchy of the generated language classes is a collapsing one, both with and without λ-rules.

Theorem 3.3 *For* $1 \leq k \leq n$,

$$\mathcal{L}(EE^{(\lambda)}(n, \leq k)) = \mathcal{L}(EE^{(\lambda)}(1, \leq 1))$$

We have already seen that only the language classes $\mathcal{L}(EE^{\lambda}(1, = 1))$, $\mathcal{L}(EE(k, = k))$, $\mathcal{L}(EE^{\lambda}(1, \leq 1))$ and $\mathcal{L}(EE(1, \leq 1))$ are interesting, the other classes are equal to one of these language families.
Thus, the only remaining thing is to examine the relation of these special language classes.
The following lemma holds by the definition of the derivation modes ≤ 1 and $= 1$.

Lemma 3.2.5 $\mathcal{L}(EE^{(\lambda)}(1, = 1)) = \mathcal{L}(EE^{(\lambda)}(1, \leq 1))$

Moreover, language classes generated without λ-rules are included in the language classes generated with the same parameters but with λ-rules.

Lemma 3.2.6 $\mathcal{L}(EE^{\lambda}(1, = 1)) \supset \mathcal{L}(EE(1, = 1))$

Proof The inclusion holds by definition; it is proper because languages containing at least one non-empty word and the empty word λ cannot be generated without λ-rules. ■

4 Summary and open problems

We can summarize the results of the paper in the form of a diagram. In this diagram, a straight arrow indicates a proper inclusion; the class the arrow leaves is included in the class the arrow points at.
We use the notation n_i for $1 \leq i$, where n_i denotes an arbitrary positive integer, such that $i \leq n_i$.

$$\begin{tikzcd}
\mathcal{L}(EE^{\lambda}(n, \leq k)) & = & \mathcal{L}(EE^{\lambda}(1, \leq 1)) & = & \mathcal{L}(EE^{\lambda}(1, = 1)) & = & \mathcal{L}(EE^{\lambda}(n, = k)) \\
\mathcal{L}(EE(n, \leq k)) & = & \mathcal{L}(EE(1, \leq 1)) & = & \mathcal{L}(EE(1, = 1)) \arrow[u] & = & \mathcal{L}(EE(n_1, = 1)) \\
 & & & & \mathcal{L}(EE(2, = 2)) \arrow[u] & = & \mathcal{L}(EE(n_2, = 2)) \\
 & & & & \vdots \arrow[u] & & \\
 & & & & \mathcal{L}(EE(k, = k)) \arrow[u] & = & \mathcal{L}(EE(n_k, = k))
\end{tikzcd}$$

The main result of this contribution is that while in the non-extended case the size parameters (n and k) and the mode of the derivation have influence on the generated language classes (see [4]), in the extended case the hierarchy is a collapsing one.

There have remained some open problems. In this paper we investigated only the relation of the language classes generated by extended simple EG systems with different parameters and derivation modes. The precise place of these language classes in a traditional hierarchy, that is their relation to the Lindenmayer or the Chomsky language classes will be the subject of future investigation. Some results concerning this question were given in [5], it was shown there that the λ-free classes are between the language families $E0L$ and MAT_{ac}.

More new questions arise if we consider another definition (different from Definition 2.3) for the derivation mode $\leq k$. By Definition 2.3, in every derivation steps at least one agent has to work. By omitting this condition and allowing that only the environment works in a derivation step, we obtain a new definition for this derivation mode and we obtain new language classes as well.

Moreover, similarly to the $\leq k$ mode we can give the definition of the $\geq k$ derivation.
The relation of the language classes defined by these derivation modes and the language classes generated by the original ones can be examined in the future.

References

[1] J. Csima. Formal language theoretic models of ecosystems (In Hungarian). Master's thesis, ELTE, Budapest, 1997.

[2] E. Csuhaj-Varjú, J. Kelemen, A. Kelemenová, and Gh. Păun. Eco(-grammar) systems. A preview. In R. Trappl, editor, *Proc 12th European Meeting on Cybernetics and System Research*, pages 941–948. World Sci. Publ., Singapore, 1994.

[3] E. Csuhaj-Varjú, J. Kelemen, A. Kelemenová, and Gh. Păun. A grammatical approach to likelife interactions. *Artificial Life*, 3:1–28, 1997.

[4] E. Csuhaj-Varjú and A. Kelemenová. Team behaviour in simple eco-grammar systems. *Theoretical Computer Science*, 209:213–224, 1998.

[5] 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.

[6] Gh. Păun, editor. *Artificial Life: grammatical models.* Black Sea Univ. Press, Bucharest, 1995.

[7] G. Rozenberg and A. Salomaa. *The Mathematical Theory of L-systems.* Academic Press, 1980.

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

[9] M. H. ter Beek. Teams in grammar systems. Master's thesis, Leiden University, Leiden, 1996.

[10] M. H. ter Beek. Teams in grammar systems: Hybridity and weak rewriting. *Acta Cybernetica*, 12(4):427–444, 1996.