COLONIES: a Multi-Agent Approach to Language Generation

Erzsébet Csuhaj-Varjú
Computer and Automation Research Institute
Hungarian Academy of Sciences
H-1111 Budapest
Kende u. 13-17
`csuhaj@luna.aszi.sztaki.hu`

Abstract We discuss colonies, a variant of grammar systems, which are a generative framework introduced for modeling multi-agent systems motivated by subsumption architectures. We demonstrate that cooperating systems of very simple grammars are convenient tools for generating complicated languages and describing powerful language classes.

1. Introduction

One of the approaches to language processing is to view language as behavior of a complex multi-agent symbol system. Grammar systems theory, a recent vivid field of formal languages, provides a versatile tool for the realization of this idea.

A grammar system, roughly speaking, is a finite set of grammars which cooperate (and communicate) in the interest of generating a language. In this framework, the grammars correspond to the agents and the generated language (or languages) identifies (identify) the behavior of the multi-agent system. At any moment, the current string under derivation represents a symbolic environment for the acting agents. Motivated by different areas, there have been several variants of grammar systems introduced and studied: CD grammar systems (cooperating/distributed grammar systems) which are a generative framework for the blackboard model of problem solving ([4]), PC grammar systems (parallel communicating grammar systems) which are computational models of (virtual) networks of parallel communicating processors ([19]), eco-grammar systems which capture the basic formal properties of ecosystems ([7]), etc. The investigations have concentrated both on the study of the generative and descriptive power of such systems and the sophisticated modeling; the interested reader can find detailed information in ([5]). For connections with natural language processing we refer to [3].

A particularly interesting case of grammar systems is the colony which was proposed as a theoretical framework for the study of the subsumption architecture of multi-agent computing devices invented by R. Brooks ([11]). Subsumption architectures, implemented as a special network of message passing augmented finite state machines ([1]), model multi-agent systems of simple autonomous agents with emergent collective behavior. The proposed generative model was a grammar system with very simple components where each component is a regular grammar generating a finite language and the component grammars change the common symbolic environment (the string) by replacing an occurrence of their startsymbol by a word of the language they define.

There are parallels among the above model and the basic requirements of behavior-based systems: colonies explicit situatedness, embodiment, and emergence. The component grammars are situated in their (symbolic) environment; their actions form a part of the dynamics of the environment and they have immediate feedback on its rewriting; the generative power (the behavior) of the whole colony emerges as a result of collective activities of the very simple purely reactive components of the system.

Colonies have been studied in detail: starting from the team behavior of the agents to the behavioral stability of the system several properties have been investigated ([12], [16], [9], [17], [8]).

In this paper we discuss different variants, mainly from the point of view of their generative power. We demonstrate that complicated and powerful language classes can be described in terms of cooperating simple grammars.

2. Some formal language theoretic prerequisites

Before turning to colonies, we recall some notions from formal language theory. For further details and information the reader is referred to [21], [20], [10].

For an alphabet V, we denote by V^+ the set of all nonempty strings over V. If the empty string, λ, is included, then we use notation V^*. The length of a string x is denoted by $|x|$.

We denote a context-free grammar by $G = (V,T,P,S)$, where V is the total alphabet, $T \subseteq V$ is the terminal alphabet, P is the set of productions and S is the startsymbol. Elements of $V - T$ are called nonterminals. A context-free grammar is regular if its productions are of the form $A \to aB, A \to a$, where A and B are nonterminal symbols and a is a terminal symbol.

The language generated by a context-free grammar G is de-

[0] Research supported by the grant No. T 017105 of the Hungarian Scientific Research Fund "OTKA"

Proceedings of the ECAI 96 Workshop
Extended Finite State Models of Language
Edited by A. Kornai.

noted by $L(G)$, we denote by $\mathcal{L}(REG)$, $\mathcal{L}(CF)$ and $\mathcal{L}(CS)$ the class of regular, context-free and context-sensitive languages, respectively.

Language classes of colonies are compared to language classes generated by parallel derivations.

By an $0L$ system we mean a triple $H = (V, P, w)$, where V is an alphabet, P is a set of context-free productions over V and $w \in V^*$ is the axiom. Moreover, P is complete, that is, P has a production $a \to u$ for any symbol $a \in V$. $0L$ systems use parallel derivations: for $x, y \in V^*$ we say that x directly derives y in $0L$ system H, written as $x \Longrightarrow_H y$, if $x = x_1 \dots x_m$, $y = y_1 \dots y_m$, $m \geq 1$, and $x_i \to y_i \in P$, $1 \leq i \leq m$. The language of an $0L$ system H is the set of words that can be derived (in some steps) from its axiom.

An $ET0L$ system is an n+3-tuple $H = (V, T, H_1, \dots, H_n, S)$, where V is the total alphabet, T is the terminal alphabet and (V, H_i, S) is an $0L$ system for every i, $1 \leq i \leq n$. A derivation step in $ET0L$ system H is a derivation step performed by some of the production sets H_i, $1 \leq i \leq n$, in the above ($0L$) manner. The generated language consists of terminal words that can be derived from the axiom S. The class of languages generated by $ET0L$ systems is denoted by $\mathcal{L}(ET0L)$.

Grammars with regulated rewriting are compared to colonies, too. A context-free programmed grammar is a 4-tuple $G = (V, T, P, S)$, where V, T, S are as above, the total alphabet, the terminal alphabet and the startsymbol, and P is a finite set of productions of the form $(b : A \to z, E, F)$, where b is a label, $A \to z$ is a context-free production over V and E, F are two sets of labels of productions of G. (E is said to be the success field, and F is the failure field of the production.) A production of G is applied as follows: if the context-free part can be successfully executed, then it is applied and the next production to be executed is chosen from those with the label in E, otherwise, we choose a production labeled by some element of F, and try to apply it. This type of programmed grammars is said to be with appearance checking; if no failure field is given for any of the productions, then a programmed grammar without appearance checking is presented. If both the success and the failure fields coincide, then we speak of a programmed grammar with unconditional transfer. The corresponding classes of languages are denoted by $\mathcal{L}(PR_{ac})$, $\mathcal{L}(PR)$, and $\mathcal{L}(UP)$. Regulated applications of productions can be defined on the base of check of context conditions.

A 4-tuple $G = (V, T, P, S)$ is said to be a context-free grammar with local context conditions if V, T, S are as above, and P is a finite set of productions of the form $p = (E, F) : A \to w$, where $E, F \subseteq V^+$ with E, F being finite sets and $A \to w$ is the context-free core production; p can be applied to a sentential form $x \in V^*$ if each element of E and no element of F is a subword of x and $A \to w$ can be executed. If E or F is the empty set, then no check of E, respectively, F is made. E is said to be the set of permitting and F is said to be the set of forbidding context conditions of p.
If both $card(E) \leq 1$ and $card(F) \leq 1$ hold, then we speak of a context-free semi-conditional grammar. If $E, F \subseteq V$, then we speak of a context-free random context grammar; if the set of permitting context conditions is empty for every production, then we have a forbidding random context grammar. The corresponding language classes are denoted by $\mathcal{L}(RC)$ and $\mathcal{L}(fRC)$, respectively.

Some important relations concerning the above language classes are as follows:

(i) $\mathcal{L}(REG) \subset \mathcal{L}(CF) \subset \mathcal{L}(ET0L) \subset \mathcal{L}(fRC) \subset \mathcal{L}(RC) = \mathcal{L}(PR_{ac}) \subset \mathcal{L}(CS)$.
(ii) $\mathcal{L}(UP) \subset \mathcal{L}(PR_{ac})$.

3. The basic model of colony

The first language theoretic model of the colony was introduced in [11]. The simple autonomous agents are represented by regular grammars, generating a finite language each (both properties express simplicity), and the emergent behavior of the system is described by the language jointly generated by the cooperating component grammars. There is no global strategy of cooperation of the grammars in the colony explicated, the generative capacity of the system emerges from the interaction of the symbolic environment (the sentential form under generation) and the collection of the grammars.

Definition 3.1

By a colony we mean an n+3-tuple $C = (V, T, R_1, \dots, R_n, S)$, where

(i) $R_i = (V_i, T_i, P_i, S_i)$, $1 \leq i \leq n$, is a regular grammar generating a finite language; R_i is called a component of C;
(ii) $S = S_i$ for some i, $1 \leq i \leq n$; S is called the startsymbol of C;
(iii) $T \subseteq \cup_{i=1}^n T_i$; T is called the set of terminals of C;
(iv) $V = \cup_{i=1}^n V_i$; V is called the total alphabet of C.

Notice that there is no global distinction introduced for the terminal alphabets and the nonterminal alphabets of the components: a terminal symbol of some component R_i can be a nonterminal symbol of an other one. Thus, complete information at some grammatical level can be incomplete at another one and reversely.

The functioning of the colony is realized by a generation process: at every derivation step a component grammar replaces an occurrence of its axiom in the current sentential form by a word from its corresponding language. The component is chosen nondeterministically from those grammars which are enabled on the sentential form, that is, able to execute an action.

We distinguish two derivation modes, both correspond to a particular cooperation strategy of the regular grammars.

Definition 3.2

Let $C = (V, T, R_1, \dots, R_n, S)$ be a colony and let $x, y \in V^+$. We say that

(i) x directly derives y in C in the basic mode (b-mode) of derivation, denoted by $x \stackrel{b}{\Longrightarrow}_C y$, if $x = x_1 S_i x_2$ for some i, $1 \leq i \leq n$, $y = x_1 w x_2$, $x_1 x_2 \in V^*$ and $w \in L(R_i)$.
(ii) x directly derives y in C in the terminal mode (t-mode) of derivation, denoted by $x \stackrel{t}{\Longrightarrow}_C y$, if $x = x_1 S_i \dots x_m S_i x_{m+1}$ for some i, $1 \leq i \leq n$, $y = x_1 w_1 x_2 w_2 \dots x_m w_m x_{m+1}$, $x_1 x_2 \dots x_{m+1} \in (V - \{S_i\})^*$ and $w_j \in L(R_i)$ for j, $1 \leq j \leq m$.

Both derivation modes are based on the competence of the active linguistic agent (the grammar): the b-mode presumes a minimal competence (the derivation proceeds one step), while in the t-mode the component activates its total competence by contributing to any incomplete information that it is able to complete (changes each occurrence of its startsymbol in the sentential form).

The language generated by C in the q-mode of derivation for $q \in \{b, t\}$ is $L_q(C) = \{w \mid S \stackrel{q}{\Longrightarrow}_C^* w, w \in T^*\}$, where $\stackrel{q}{\Longrightarrow}_C^*$ denotes the reflexive and transitive closure of $\stackrel{q}{\Longrightarrow}_C$.

If no confusion arises, then subscript C can be omitted from $\stackrel{q}{\Longrightarrow}_C$.

Before turning to language classes of colonies, we note that according to the different selections of the terminal set from the total alphabet of the colony we distinguish colonies with different styles of acceptance. Acceptance styles express the agreement of the linguistic agents (grammars) on the set of grammatically acceptable sentences.

Definition 3.3

We say that a colony $C = (V, T, R_1, \ldots, R_n, S)$ is of acceptance style

(i) "arb" if $T \subseteq \cup_{i=1}^n T_i$; (any word that is acceptable for the whole colony is acceptable for some component grammar);
(ii) "one" if $T = T_i$; for some i, $1 \leq i \leq n$; (the acceptable words are those ones that are acceptable for a distinguished component);
(iii) "ex" if $T = \cup_{i=1}^n T_i$; (any word that is acceptable for some grammar is acceptable for the whole colony);
(iv) "all" if $T \subseteq \cap_{i=1}^n T_i$; (the acceptable words of the colony are those one that are acceptable for all components).

Language classes of colonies were studied in detail in [12], [16], [17].

Results on the generative power of colonies demonstrate that the behavior of the system is more complex than that of the components, thus, systems of cooperating syntactically simple grammars provide tools for the description of powerful language classes. Both the choice of the cooperation strategy and that of the acceptance style have significant impact on the power of the generated language class.

Let us denote by $\mathcal{L}(Col, q, f)$ the class of languages generated by colonies with acceptance style f in the q-mode of derivation, for $f \in \{arb, one, ex, all\}$ and $q \in \{b, t\}$.

In [12] and [16] it was shown that the context-free language class can be represented as the class of behavior of colonies with acceptance style one, arb and all under derivation mode b. But, acceptance style ex leads to a less powerful language class, namely a proper subclass of pure context-free languages.

Theorem 3.1 ([16])

For $f \in \{arb, one, all\}$

$$\mathcal{L}(Col, b, ex) \subset \mathcal{L}(Col, b, f) = \mathcal{L}(CF).$$

Total competence as cooperation strategy is more powerful than the minimal one, that is,

Theorem 3.2 ([16])

For $f \in \{arb, one, all\}$

$$\mathcal{L}(Col, b, f) \subset \mathcal{L}(Col, t, f) \subset \mathcal{L}(ETOL).$$

Acceptance style ex is less powerful than the other ones in the case of t-derivation, too, since, by [16] we have

Theorem 3.3

For $f \in \{arb, one, all\}$

$$\mathcal{L}(Col, t, ex) \subset \mathcal{L}(Col, t, f).$$

In some cases acceptance styles and cooperation strategies determine incomparable language classes.

Theorem 3.4 ([16])

(i) $\mathcal{L}(Col, b, f)$ and $\mathcal{L}(Col, t, ex)$ are incomparable.
(ii) $\mathcal{L}(Col, b, ex)$ and $\mathcal{L}(Col, t, ex)$ are incomparable.

By virtue of Definition 3.2., a component grammar is allowed to derive a string which contains some symbol being not in its total vocabulary. The motivation of this definition is that the grammars model simple agents which are able to recognize/interact with only a very restricted part of their environment. If the regular grammars can be active only on sentential forms that are defined over their own alphabet, then we obtain significantly different and more powerful language classes. Let us first introduce a definition.

Let $C = (V, T, R_1, \ldots, R_n, S)$ be a colony and let $x, y \in V^+$. We speak of a direct derivation step in the basic mode of derivation in the strict sense and denote it by $x \stackrel{d}{\Longrightarrow}_C y$, if $x = x_1 S_i x_2$, $x_1 x_2 \in V_i^*$, for some i, $1 \leq i \leq n$, and $y = x_1 w x_2$, where $w \in L(R_i)$. (Both x and y are over V_i.)

The language generated by C in the d-mode of derivation is defined in the usual way and it is denoted by $L_d(C)$; $\mathcal{L}(Col, d, f)$ denotes the class of languages generated by colonies with acceptance style f in the d-mode of derivation, where $f \in \{arb, one, all, ex\}$.

Let us denote by fRC the class of forbidding random context grammars. Then,

Theorem 3.5

For $f \in \{arb, all\}$

$$\mathcal{L}(Col, d, f) = fRC.$$

Sketch of the proof: Let $C = (V, T, R_1, \ldots, R_n, S)$ be a colony. Then, forbidding random context grammar $G = (V, T, P, S)$, where $P = \{(\emptyset, (V - V_i)) : S_i \to \alpha \mid \alpha \in L(R_i), 1 \leq i \leq n\}$ generates the same language as C. (Every production corresponds to an action of component R_i : first, the grammar checks whether the sentential form contains a symbol not in its vocabulary or not, then, if the answer is negative, replaces S_i by a word from $L(R_i)$.) Reversely, let $G = (V, T, P, S)$ be a forbidding random context grammar. By some standard techniques it can be shown that there is a forbidding random context grammar $G' = (V', T, P' S')$ which is equivalent to G such that for any production $p' = (\emptyset, Q) : A \to \alpha$ in P' it holds that $Q \cap T = \emptyset$ and A does not occur in α. Then, by ordering to any production p' the corresponding component

$R_i = (V - Q, T \cup alph(\alpha), A \to \alpha, A)$ we can construct a colony C with $L_d(C) = L(G')$.

4. Extended colonies

According to the basic model, the sentential form that corresponds to the symbolic environment of the agents changes only by the actions of the component grammars. This restriction is very strict in some sense: a sophisticated way of generation presumes some additional operations on the obtained sentential form (transformation, ordering, etc). A natural idea to solve this conflict is to furnish the colony with some mechanism which provides own development for the environment. One possibility is if this extension is defined by a generative system and an other one is when the change is done by the help of some translating mechanism (a finite state transducer).

The notion we introduce here corresponds to a variant of eco-grammar systems, a model introduced for describing ecosystems ([7]).

Definition 4.1
By an e-colony (an extended colony) we mean an $s+n+3$-tuple
$E = (V, T, H_1, \ldots, H_s, R_1, \ldots, R_n, S)$, where

(i) V and T are alphabets, $T \subseteq V$; called the total alphabet and the terminal alphabet of E, respectively;
(ii) H_i, $1 \le i \le s$ is a finite set of context-free rules over V called a set (a table) of developmental rules of the symbolic environment; H_i is complete (for any symbol $a \in V$ there is at least one production $a \to \alpha \in H_i$);
(iii) $R_j = (V_j, T_j, P_j, S_j)$, $1 \le j \le n$, is a regular grammar generating a finite language, $V_j \subseteq V$; R_j is called a component of E;
(iv) $S \in V$ is the startsymbol of E.
(v) $T \subseteq \cup_{i=1}^{n} T_i$.

The functioning of the system consists of an action of a component and a developmental step of the remaining part of the symbolic environment (the sentential form), both substeps executed simultaneously. If no action is possible, then only the developmental rules are applied. Moreover, tables H_i, $1 \le i \le s$, are applied in parallel ($0L$) manner.

We define only the basic mode of functioning, the terminal mode can be defined analogously to Definition 3.2.

Definition 4.2
Let $E = (V, T, H_1, \ldots, H_s, R_1, \ldots, R_n, S)$ be an e-colony and let $x, y \in V^+$. We say that x directly derives y in E in the basic mode of derivation (in the b-mode of derivation), written as $x \overset{b}{\Longrightarrow}_E y$, if one of the following two cases hold:

(i) there is startsymbol S_i of some component R_i, $1 \le i \le n$, such that $x = x_1 S_i x_2$, $x_1, x_2 \in V^*$. Then $y = y_1 w y_2$, $w \in L(R_i)$, and $x_1 \Longrightarrow_{H_j} y_1$, $x_2 \Longrightarrow_{H_j} y_2$ for some j, $1 \le j \le s$, H_j is applied in the $0L$ manner.
(ii) $x \ne x_1 S_i x_2$ for any i, $1 \le i \le n$, $x_1, x_2 \in V^*$. Then $x \Longrightarrow_{H_j} y$ for some j, $1 \le j \le s$, H_j is applied in the $0L$ manner.

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

The language generated by an extended colony E in the b-mode of derivation is $L_b(E) = \{w \in T^* \mid S \overset{b}{\Longrightarrow}{}^*_E w\}$.

Let us denote by $\mathcal{E}(Col, b, f)$ the class of languages generated by e-colonies with acceptance style f in derivation mode b, where $f \in \{arb, one, all, ex\}$.

As it is expected, by some simple technical considerations we obtain

Theorem 4.1

$$\mathcal{L}(ET0L) \subset \mathcal{E}(Col, b, arb).$$

The case when the sentential form is changed by a finite state transducer after an action of a component grammar is close to linguistic motivations. Such model was introduced and examined in [17]. The idea beyond the notion is that the components "speak" different languages and the finite state transducer intermediates between them.

Definition 4.3

A t-colony (a colony with a finite state transducer) is an $n+4$-tuple
$K = (V, T, R_1, \ldots, R_n, \tau, S)$, where elements V, T, $R_1, \ldots, R_n$, S are elements of a colony, defined in the same way as in Definition 3.1., and $\tau = (V, V, Q, s_0, F, P)$ is a finite state transducer (Q is the set of states, s_0 is the initial state, F is the set of final states, P is the set of translation rules of the form $sa \to xs'$, $s, s' \in Q$, $a \in V$, $x \in V^+$).

For $q \in \{b, t\}$ we define the language generated by K in the q-mode of derivation by $L_q(K) = \{w \in T^* \mid S \overset{q}{\Longrightarrow}_{R_{i_1}} w_1 \equiv> g(w_1) \overset{q}{\Longrightarrow}_{R_{i_2}} w_2 \equiv> g(w_2) \ldots \overset{q}{\Longrightarrow}_{R_{i_s}} w_s = w,\ s \ge 1, 1 \le i_j \le n, 1 \le j \le s\}$.
($\overset{q}{\Longrightarrow}_{R_{i_j}}$ denotes that component R_{i_j} executes an action in the q-mode.)

Let $\mathcal{K}(Col, q, f)$ denote the class of languages generated by t-colonies with acceptance style f in the derivation mode q, where $f \in \{arb, one, all, ex\}$ and $q \in \{b, t\}$.

Since generalized sequential mappings induced by finite state transducers are able to verify whether the sentential form belongs to a regular language or not, by [17] the following statement holds.

Theorem 4.2

$\mathcal{K}(Col, b, arb) = \mathcal{L}(CS)$.

The proof is based on the result that any context-sensitive language can be generated by a conditional context-free grammar, where every production p is associated by the same regular language R, and p can be applied to the sentential form x iff $x \in R$. (For the result see [10].)

5. Components with context conditions

Components of the basic generative model of colony are functioning on the base of a poor ability of recognition, namely, they are able to start with the derivation if they identify an occurrence of their startsymbol in the sentential form. Enhanced recognition (verification of more complex parts or properties of the string) usually results in a more sophisticated intrinsic behavior of the system, thus, colonies with such members are expected to be convenient tools for describing powerful language classes with complicated structure. Colonies with components associated with context conditions are such models. In this case the grammar can execute an action if and only if the symbolic environment (the sentential form) satisfies some requirements.

Context conditions can be defined on the base of some qualitative property of the string (subword occurrence) or on some quantitative property (the number of symbols, etc.)

To demonstrate the increased power of grammars with context conditions, we recall a statement from [2], by means of that every context-sensitive language can be generated by a grammar which has context-free core productions associated with an identical finite set of strings as global forbidding context conditions. (The context-free production can be applied if the sentential form does not have any of the associated strings as a subword.) Moreover, because the context-free core productions can be chosen non-recursive ones, we can formulate the result in the frame of colonies as follows: the class of context-sensitive languages is equal to the class of languages generated by colonies, where the components can perform an action iff the sentential form satisfies a set of forbidding context conditions associated as global context condition to the system.

Context conditions in the above statement were forbidding (negative) ones. An interesting question is, what can we say of language classes determined by colonies with permitting (positive) context conditions. (To check forbidding context conditions the whole sentential form have to be tested, while in the case of permitting context conditions it is enough to find the first subword occurrence.)

A nice result can be presented for extended colonies with context conditions. In this case, we are able to reach the power of the context-sensitive language class by components associated with only permitting (positive) context conditions (the negative context check is done by the developmental rule sets of the symbolic environment).

Definition 5.1

An $s+n+3$-tuple $F = (V, T, H_1, \ldots, H_s, R_1, \ldots, R_n, S)$ is said to be an extended colony with context conditions (an ec-colony) if the following holds:

(i) $V, T, H_1, \ldots, H_s, S$ are the total alphabet, the terminal alphabet, the sets of environmental developmental rules and the axiom of F, respectively, defined in the same way as in the case of extended colonies (Definition 4.1);

(ii) $R_i = (V_i, T_i, (u_i, v_i) : P_i, S_i), 1 \leq i \leq n$, where (V_i, T_i, P_i, S_i) is a regular grammar generating a finite language (defined in the same way as in Definition 3.1), $V_i \subseteq V$, and (u_i, v_i) is a global context condition for P_i over V_i.

(String u_i and v_i is either a nonempty word over V_i or it is equal to the empty set, respectively. A production p from P_i can be applied to a sentential form x if x contains u_i as a subword (the permitting context condition) and does not contain v_i as a subword (the forbidding context condition). If u_i and/or v_i is equal to the empty set, then no context check is made.)

The way of functioning of the ec-colony can be derived from that of the extended colonies. We give here only the notion of the basic mode of derivation, the definition of the terminal derivation can be obtained in a similar way.

Definition 5.2.

Let $F = (V, T, H_1, \ldots, H_s, R_1, \ldots, R_n, S)$ be an ec-colony and let $x, y \in V^+$. We say that x directly derives y in F in the basic mode of derivation (in the b-mode of derivation), written as $x \stackrel{b}{\Longrightarrow}_F y$, if one of the two cases hold:

(i) $x = x_1 S_i x_2$, $x_1, x_2 \in V^*$ and x satisfies the corresponding context condition (u_i, v_i) for some i, $1 \leq i \leq n$. Then $y = y_1 w y_2$, $w \in L(R_i)$, and $x_1 \Longrightarrow_{H_j} y_1$, $x_2 \Longrightarrow_{H_j} y_2$ for some j, $1 \leq j \leq s$, H_j applied in the $0L$ manner.

(ii) $x \neq x_1 S_i x_2$ for any i, $1 \leq i \leq n$, $x_1, x_2 \in V^*$. Then $x \Longrightarrow_{H_j} y$ for some j, $1 \leq j \leq s$, H_j applied in the $0L$ manner.

The language generated by F in the b-mode of derivation, is $L_b(F) = \{w \in T^* \mid S \stackrel{b}{\Longrightarrow}{}^*_E w\}$.

Theorem 5.1

The class of context-sensitive languages is equal to the class of languages generated by ec-colonies of acceptance style "arb" in the basic mode of derivation, where the components have only permitting context conditions of length at most two.

Hints of the proof: By [18] it is known that any context-sensitive language L can be generated by a grammar with local context conditions (every production is associated with context conditions), where the length of every permitting context condition is at most two and every forbidding context condition is either a symbol or it is empty. Moreover, we can choose the core productions non-recursive. When constructing the equivalent ec-colony, the productions with the associated permitting context conditions correspond to the component grammars and the check of the presence of the forbidding context symbol is done by some appropriate developmental rule set (the property of completeness is used). The reverse inclusion can be proved by using some standard simulation techniques.

6. Structured colonies

Examining colonies, one question which immediately arises is whether some relations among the component grammars improve the generative and descriptive power of the system. In the basic model of colonies there is no structure of the collection of the regular grammars explicated: some intrinsic hierarchies and heterarchies of the components arise dynamically from the functioning of the system.

Imposing an appropriate organizational structure or utilizing inner relations among the grammars is a reasonable way to improve efficiency of cooperation and enhance the descriptive power of the system.

Structures can be static ones (imposed together with the sets of components) or dynamic ones (determined by the functioning of the colony). Examples for static structures are graph-controlled colonies (the components are associated by nodes of a digraph and follow each other in the derivation according to paths in the graph) and colonies with teams of components (sets of components), where grammars belonging to the same team have to act simultaneously on the sentential form. Both cases lead to enhancement of the generative power.

An interesting variant of colonies with team organization is the symbiotic colony, introduced in [8], motivated by multi-agent systems exhibiting life-like behavior. In this case the team members have to act simultaneously on adjacent letters of the sentential form. Moreover, there are no teams with a common member. Thus, components in the same team highly determine the activity of each other. The model provides a generative mechanism as powerful as the context-sensitive grammar.

The above variants correspond to colony models with static structures.

A dynamic structure among the components is determined implicitly by the appearance of the corresponding axiom in the sentential form. Whenever its startsymbol shows up, the component enters competition with the other enable ones for the possibility of executing an action. This competitive behavior of the agents (the grammars) leads to competitive parallelism in colonies. There are two variants of competition distinguished: the strong competitive parallelism, where each grammar which is enable has to execute an action on the sentential form (otherwise the system aborts) and the weak competitive parallelism, when a largest number of enabled components acts parallel on the sentential form. Both models were introduced and studied in [9].

The formal definitions are as follows:

Definition 6.1
Let $C = (V, T, R_1, \ldots, R_n, S)$ be a colony and $x, y \in V^*$. We say that x directly derives y in C the

(i) strongly competitive mode of derivation, written as $x \stackrel{sp}{\Longrightarrow}_C y$, iff $x = x_1 S_{i_1} x_2 S_{i_2} \ldots x_k S_{i_k} x_{k+1}$ and $y = x_1 w_{i_1} x_2 w_{i_2} \ldots x_k w_{i_k} x_{k+1}$, where $w_{i_j} \in L(R_{i_j})$, $1 \leq j \leq k$, $i_u \neq i_v$ for all $u \neq v$, $1 \leq u, v \leq k$, $|x|_{S_t} > 0$ implies $i_j = t$ for some j, $1 \leq j \leq k$.
(If a component can be used, then it must be used.)

(ii) weakly competitive mode of derivation, written as $x \stackrel{wp}{\Longrightarrow}_C y$, iff $x = x_1 S_{i_1} x_2 S_{i_2} \ldots x_k S_{i_k} x_{k+1}$ and $y = x_1 w_{i_1} x_2 w_{i_2} \ldots x_k w_{i_k} x_{k+1}$, where $w_{i_j} \in L(R_{i_j})$, $1 \leq j \leq k$, $i_u \neq i_v$ for all $u \neq v$, $1 \leq u, v \leq k$, $|x|_{S_t} > 0$ implies $i_j = t$ for some j, $1 \leq j \leq k$, and k is the maximal integer with the previous properties.

As usually, we define the corresponding generated language by $L_q(C) = \{w \in T^* \mid S \stackrel{q}{\Longrightarrow}_C^* w\}$ for $q \in \{sp, wp\}$.
($\stackrel{q}{\Longrightarrow}_C^*$ denotes the reflexive and transitive closure of $\stackrel{q}{\Longrightarrow}_C$.)

Let us denote by $\mathcal{L}(Col, q, f)$ the class of languages generated by colonies of acceptance style f in derivation mode q, where $f \in \{arb, all, one, ex\}$ and $q \in \{sp, wp\}$.

Both the strong and the weak competition of the components in colonies results in interesting powerful language classes.

Theorem 6.1. ([9])

$$\mathcal{L}(CF) \subset \mathcal{L}(Col, sp, arb) \subseteq \mathcal{L}(PR_{ac}).$$

Theorem 6.2. ([9],[14])

$$\mathcal{L}(CF) \subset \mathcal{L}(Col, wp, arb) \subseteq \mathcal{L}(UP).$$

More precise comparisons of the two modes of derivations have not been presented yet.

7. Some final remarks

Grammar systems and, thus, colonies give a flexible tool for modeling different levels and aspects of natural language generation by providing a unified treatment for handling many questions at different levels of language processing. The same formalism can be used, for example, for generating discourses, texts, and sentences, but the model is useful for describing social nature of language, too. There is a wide variety of applications of colonies (and grammar systems in general): a new approach to handling grammatically not correct sentences (acceptance fields), style of speaking as behavior of the same colony with different acceptance fields.

One important field for future research can be examining colonies with finite transducers as components and compare the accepting model to the generative one, both from the point of view of power and size (how many components are necessary, etc.). Questions of size, stability, delay, step-limited activity and message passing are of interest, too. Except message passing there have been some basic investigations in these topics ([12],[17]).

REFERENCES

[1] Brooks, R. (1991): Intelligence without representation. Artificial Intelligence, 139-159.

[2] Csuhaj-Varjú, E. (1992): On grammars with local and global context conditions. Intern. Journal of Computer Mathematics, 17-27.

[3] Csuhaj-Varjú, E. (1994): Grammar systems: a framework for natural language generation. In: Mathematical Aspects of Natural and Formal Languages. World Scientific Series in Computer Science, Vol. 43., (Gh. Păun, ed.) World Scientific, Singapore, 63-78.

[4] Csuhaj-Varjú
E., Dassow, J (1990): On cooperating/distributed grammar systems, J. Inf. Process. Cybern. EIK 26, 49-63.

[5] Csuhaj-Varjú, E., Dassow, J., Kelemen, J., Păun, Gh. (1994): Grammar Systems - A Grammatical Approach to Distribution and Cooperation. Gordon and Breach Science Publishers, London

[6] Csuhaj-Varjú, E., Dassow, J., Kelemen, J., Păun, Gh. (1994): Stratified Grammar Systems. Computers and Artificial Intelligence 13(5), 409-422.

[7] Csuhaj-Varjú, E., Kelemen, J., Kelemenová, A., Păun, Gh. (1994): Eco(grammar)systems - a preview. In: Cybernetics and Systems '94, (R.Trappl, Ed.), World Scientific, Singapore, 941–948.

[8] Csuhaj-Varjú, E., Păun, Gh. (1993-94): Structured Colonies: Models of Symbiosis and Parasitism. An. Univ. Bucuresti, Ser. Mat.-Inf. XLII-XLIII, 15-31.

[9] Dassow, J., Kelemen, J., Păun, Gh. (1993): On Parallelism in Colonies. Cybernetics and Systems 24, 37-49.

[10] Dassow, J., Păun, Gh. (1989): Regulated Rewriting in Formal language Theory. Springer Verlag, Berlin.

[11] Kelemen, J., Kelemenová, A. (1992): A subsumption architecture for generative symbol systems. In: Cybernetics and System Research'92 (R. Trappl, ed.), World Scientific, Singapore, 1529-1536.

[12] Kelemen, J., Kelemenová, A. (1992): A grammar-theoretic treatment of multiagent systems. Cybernetics and Systems 23, 621–633.

[13] Kelemenová, A., Kelemen, J. (1994): From colonies to eco (grammar) systems. In: Important Results and Trends in Theoretical Computer Science, (J.Karhumaki, H.Maurer, G.Rozenberg, eds.), LNCS 812, Springer-Verlag, Berlin, 213–231.

[14] Kelemenová, A. (1996): Private communication.

[15] Kelemenová, A., Csuhaj-Varjú, E. (1994): On the power of colonies. In: Proc. 2nd Colloquium on Words, Languages and Combinatorics, Kyoto, 1992, (M. Ito and H. Jürgensen, eds.), World Scientific, Singapore, 1994, 223-234.

[16] A. Kelemenová, Csuhaj-Varjú, E. (1994): Languages of colonies. Theoretical Computer Science 134, 119-130.

[17] Păun, Gh. (1995): On the generative capacity of colonies. Kybernetika, 83-97.

[18] Păun, Gh. (1985): A variant of random context grammars: semi-conditional grammars. Theoretical Computer Science 41, 1-17.

[19] Păun, Gh., Santean, L. (1989): Parallel Communicating grammar systems: the regular case, Annales. Univ. Bucuresti., Ser. Matem.-Inform. 38, 55 - 63.

[20] Rozenberg, G., Salomaa, A. (1980): The Mathematical Theory of L Systems. Academic Press, New York.

[21] Salomaa, A. (1973): Formal Languages. Academic Press, New York.