ON THE POWER OF COMMUNICATION STRUCTURE FOR DISTRIBUTIVE GENERATION OF LANGUAGES.

DANA PARDUBSKÁ[1]
Department of Artificial Intelligence, Faculty of Mathematics and Physics
Comenius University, 842 15 Bratislava, Slovakia

ABSTRACT

The communication structure of parallel comunicating grammar sysytems ($PCGS$) considered as the descriptional complexity of $PCGS$ is investigated here. The aim of the paper is to compare computational power of $PCGS$'s with respect to their communication structure. Investigating the generation of the languages over one-letter alphabet the communication structures dags(directed acyclic graphs) and cycles are proved to be less powerful then complete graphs are. Moreover it is shown that the increase of the number of component grammars of $PCGS$'s cannot compensate for a suitable communication structure.

1.Introduction

Parallel Communicating Grammar Systems (PCGS) representing a formal model for parallel (distributive) generation of languages has been introduced by G.Paun and L.Santean in [1]. Informally, a PCGS of degree n consists of n separate grammars generating synchronously words in derivations starting from their own axiom. The cooperation is based on some communications realized in so called communication steps. In one communication step one grammar G may via special nonterminals require a string generated by another grammar G' of PCGS. After receiving it G includes this string in its own string and G' continues derivation from its axiom. There is special, output grammar G generating a string considered to be the output of a whole PCGS. The precise, formal definition of PCGS follows in the next section.

In this paper we investigate the PCGSs where components are regular grammars with the rules from N x T^*N, N x T^+ , where N and T are the set of nonterminals and the set of terminals respectively. This model of PCGS is one of the most studied ones in the literature [1–5] because its components are the simpliest one (from the Chomsky hierarchy point of view), and so such PCGS corresponds to a natural, distributive system with simple processors. The aim of this paper is to investigate the generation power of these systems, their complexity measures, and the relationships among them. The most investigated complexity measure for PCGS has been the number of grammars the PCGS consists of, which is clearly a descriptional complexity

[1]This work has been done during the author's stay at the Dept. of Mathematics & Informatics of the University of Paderborn

measure. The hierarchy results claiming that cPCGS (centralised PCGS) of regular $n+1$ grammars are more powerful than cPCGS of n regular grammars for any positive integer n has been established in [3]. Because the complexity of communication shows generally to be the crutial one in parallel systems, Hromkovič et.al.[4] proposed to consider two communication complexity measures for PCGS. The first one is the communication structure of PCGS (the shape of the graph consisting of the directed communication links between the grammars of the system) which can be considered as alternative descriptional complexity measure. As the classes of structures of a principal interest, linear arrays (chains), rings, trees and directed acyclic graphs are proposed in [4]. The second communication complexity measure introduced in [2,4] and studied also in [5] is the number of exchanged messages (strings) during the generation procedure. This complexity measure is clearly a computational complexity measure which may be considered as a function of the length of the generated word.

The main aim of our paper is to investigate the power of different communication structures. In [4] some special lower bound techniques were developed for special communication structures as rings, trees and dags. These techniques have enabled to show strong hierarchies on the number of grammars in these structures as well as strong hierarchies showing that PCGS with $k+1$ communications are more powerful than PCGS with k communication for chains and trees. Unfortunately, no result showing that any type of communication structures is more powerful than another one has been achieved there. Here, we give the first result of this kind. Showing that rings and dags can generate only regular languages over one-letter alphabet (independently of the number of component grammars used) we get that rings and dags are weaker than arbitrary structures (complete graphs). Since a nonregular language over one-letter alphabet can be achieved by 3 grammars, this shows that the number of grammars cannot compensate for suitable communication structure.

2. Definitions and notations.

We assume the reader to be familiar with basic definitions and notations in formal language theory and we specify only some of them related to the PCGS.

We denote by ε the empty symbol and by $|x|$ the length of x. For a set $K, |x|_K$ denotes the number of occurrences of symbols from K in x.

Definition 1 *A PCGS of degree n,$n \geq 1$, is an n-tuple $\Pi = (G_1, \ldots, G_n)$, where*

- *$G_i = (N_i, T, S_i, P_i)$ are regular grammars satisfying*
  - *$N_i \cap T = \emptyset$ for all $i \in \{1, \ldots, n\}$*
  - *$P_i \subset N \text{ x } T^*N \cup N \text{ x } T^+$*
- *there exists a set $K \subseteq \{Q_1, \ldots, Q_n\}$ of special symbols, called communication symbols, used in communications as will be shown below.*

The communication protocol in a PCGS Π is determined by its *communication graph*. The vertices of this directed graph $G(\Pi)$ are labeled by $G_1, \ldots, G_n$ and directed edge (G_i, G_j) is presented in the $G(\Pi)$ iff the communication symbol Q_j belongs to the nonterminal alphabet of G_i.

An n-tuple $(x_1, \ldots, x_n)$, $x_i = \alpha_i A_i$, $\alpha_i \in T^*, A_i \in (N_i \cup \epsilon)$, is called *configuration*. The n-tuple $(A_1, A_2, \ldots, A_n)$ is called the *nonterminal cut* of the configuration $(x_1, \ldots, x_n)$. If the nonterminal cut of the configuration $(x_1, \ldots, x_n)$ is such that it contains at least one communication symbol, then so called *communication cut*, that is n-tuple $(B_1, B_2, \ldots, B_n)$, where $B_i = A_i$ for $A_i \in K$ and $B_i = *$ for $A_i \notin K$, can be associated with it.
We say $(x_1, \ldots, x_n)$ *directly derives* $(y_1, \ldots, y_n)$ and write $(x_1, \ldots, x_n) \rightarrow (y_1, \ldots, y_n)$, if one of the next two cases holds:

1. $|x|_K = 0$ for all i, $1 \leq i \leq n$, and either $x_i \rightarrow y_i$ in G_i when x_i contains nonterminal or x_i is the terminal word and $y_i = x_i$.

2. if $|x_i|_K > 0$ for some i, $1 \leq i \leq n$, then for each such i we write $x_i = z_i Q_j$, where $z_i \in T^*$.

   (a) If $|x_j|_K = 0$ then $y_i = z_i x_j$ and $y_j = S_j$

   (b) If $|x_j|_K > 0$ then $y_i = x_i$.

For all remaining indexes t, for which x_t does not contain communication symbol and Q_t has not occured in any of x_i's, we put $y_t = x_t$.

A derivation consists of *rewriting* and *communication* steps.

If no communication symbol appears in any of the component grammars then we perform a *rewriting step* consisting of rewriting steps synchronously performed in each of the grammars. If some of the commponents is a terminal string, it is left unchanged. If some of the component grammars contains nonterminal that cannot be rewritten, the derivation is blocked.

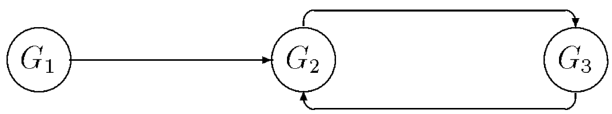
If a communication symbol is present in any of the components, then a *communication step* is performed. It consists of replacing all communication symbols with the phrases they refer to under condition these phrases do not contain communication symbol. If some communication symbols are not satisfied in this communication step, they may be satisfied in one of the next ones. Communication steps are performed until no more communication symbols are present or the derivation is blocked because no communication symbol has been satisfied in the last communication step.

The language generated by the system consists of the terminal words generated in G_1 (in the cooperation with the other grammars).

Definition 2 $L(\Pi) = \{\alpha \in T^* | \ (S_1, \ldots, S_n) \rightarrow^* (\alpha, \beta_2, \ldots, \beta_n)\}$.

Now, to illustrate the definition of PCGS and to motivate some new notions, the example of PCGS generating the language over one-letter alphabet follows.

Example 1 *Let us have a PCGS* Π *(unambiguously) given by the sets of rules of its component grammars*
P_1: $\{\ S_1 \to Q_2, Z \to Z_3, Z_3 \to Z_4, Z_4 \to Q_2|a\}$
P_2: $\{\ S_2 \to Z_2, Z_2 \to Q_3, Z_4 \to Z_2\}$
P_3: $\{\ S_3 \to a^2Q_2, Z_2 \to Z_3, Z_3 \to Z_4\}$.



communication graph

The derivation of PCGS Π can proceed as follows.
$(S_1, S_2, S_3) \xrightarrow{1} (Q_2, Z_2, a^2Q_2) \xrightarrow{2} (Z_2, S_2, a^2Z_2) \xrightarrow{3} (Z_3, Z_2, a^2Z_3) \xrightarrow{4} (Z_4, Q_3, a^2Z_4) \xrightarrow{5} (Z_4, a^2Z_4, S_3) \xrightarrow{6} (Q_2, a^2Z_2, a^2Q_2) \xrightarrow{7} (a^2Z_2, S_2, a^4Z_2) \to \ldots \to (a^2Z_4, a^2Z_2, S_3) \to (a^3, a^2Z_4, a^2Q_2)$

The first step is generative, because no communication symbol is presented in any sequential form of any component grammar. After this step the first and third component grammars contain communication symbol Q_2. Since the second component grammar does not contain any communication symbol, the second -communication - step of the derivation is succesfuly performed. During this step communication symbol Q_2 is replaced by the sequential form of the grammar G_2 while S_2 becomes the new content of the component grammar G_2. Analogously, the derivation steps $3, 4$ and 6 are generative and the derivation steps 5 and 7 are communication ones.

We will show, that $L(\Pi) = \{a^{n^2+n+1} | n \geq 1\}$. To do it, let us describe the derivation in another - more appropriate - form.

G_1 :	S_1	Q_2	Z_2	Z_3	Z_4	Z_4	Q_2	a^2Z_2		a^2Z_4	a^3
G_2 :	S_2	Z_2	S_2	Z_2	Q_3	a^2Z_4	a^2Z_2	S_2	$\cdots$	a^2Z_4	a^2Z_2
G_3 :	S_3	a^2Q_2	a^2Z_2	a^2Z_3	a^2Z_4	S_3	a^2Q_2	a^4Z_2		S_3	a^2Q_2

It is easy to see that Π works on a cycle that is the repetition of the derivation steps $3, 4, 5, 6$ (indicated by marks ||▌ ||▌). The last derivation step differs from that of 6 in the first component. The first component grammar aplies the rule $Z_4 \to a$ instead of $Z_4 \to Q_2$.

From the nonterminal cut point of view this derivation can be written as follows:

$$\begin{pmatrix} S_1 \\ S_2 \\ S_3 \end{pmatrix} \begin{pmatrix} Q_2 \\ Z_2 \\ Q_2 \end{pmatrix} \left(\begin{pmatrix} Z_2 \\ S_2 \\ Z_2 \end{pmatrix} \begin{pmatrix} Z_3 \\ Z_2 \\ Z_3 \end{pmatrix} \begin{pmatrix} Z_4 \\ Q_3 \\ Z_4 \end{pmatrix} \begin{pmatrix} Z_4 \\ Z_4 \\ S_3 \end{pmatrix} \begin{pmatrix} Q_2 \\ Z_2 \\ Q_2 \end{pmatrix} \right)^+ \begin{pmatrix} Z_2 \\ S_2 \\ Z_2 \end{pmatrix} \begin{pmatrix} Z_3 \\ Z_2 \\ Z_3 \end{pmatrix} \begin{pmatrix} Z_4 \\ Z_4 \\ Z_4 \end{pmatrix} \begin{pmatrix} \varepsilon \\ Z_2 \\ S_3 \end{pmatrix}$$

$L(\Pi) = \{a^{\{n^2+n+1\}} | n \geq 1\}$ simply follows from the fact (provable by induction), that after i repetitions of the cycle the configuration of Π is

$$\left(\ a^{i^2+i} Z_2, S^2, a^{2i+2}, Z_2 \ \right).$$

Now, let us stress our attention in more details to the derivation of the PCGS.

Let $\mathcal{D}(w) = C_0, C_1, C_2, \ldots, C_t$ be a derivation of the word w. With this derivation two sequences of nonterminal cuts could be asociated. First one is that of all nonterminal cuts of this derivation and the second one is a subsequence containing only those nonterminal cuts with at least one communication symbol in it. We will call this second sequence *the communication sequence of the derivation.* Let $I = \{t_1, t_2, ..., t_k\}, t_1 < t_2 < ... < t_k$, be the set of all communication steps of $\mathcal{D}(w)$.

The communication sequence of the derivation from Example 1 is (Q_2, Z_2, Q_2), (Z_4, Q_3, Z_4), (Q_2, Z_2, Q_2).

Notation

• The i-th *generative section* of $\mathcal{D}(w)$, $1 \leq i \leq k+1$, is the subsequence $t_{i-1}+1, t_{i-1}+2, \ldots, t_i - 1$ of derivation steps, $t_0 = 0, t_k - 1 = t$.

• We denote by $g(i,j)$ the string generated in G_i during the $j-th$ generative section of $\mathcal{D}(w)$. More precisely: let $C_j^{'} = (\alpha_1 A_1, \alpha_2 A_2, \ldots, \alpha_n A_n)$ be the configuration just at the begining of the $j-th$ generative section and let $C_i^{''} = (\alpha_1\beta_1 B_1, \alpha_2\beta_2 B_2, \ldots,$ $\alpha_n\beta_n B_n)$ be that at the end of this section. Then $g(i,j)$ is the substring β_i of $\alpha_i\beta_i B_i$.

Now, it is not difficult to realize that the word w generated by PCGS Π in derivation $\mathcal{D}(w)$ can be composed from some $g(i,j)$'s.

• Let $n(i,j)$ be the number of ocurrences of $g(i,j)$ in w.

Some of actual values of $n(i,j)$'s and $g(i,j)$'s from derivation mentioned in Example 1 are $n(1,1) = n(1,2) = n(1,3) = 1$

$g(1,1) = g(1,2) = g(1,3) = \varepsilon$

$g(2,1) = g(2,2) = g(2,3) = \varepsilon$

$g(3,1) = g(3,3) = a^2$.

Fact. *Let k be the number of generative sections of a derivation. Then the value $p(i,j), 1 \leq j \leq k$, depend on communication sequence, not on the derivation at whole.*

The last notion we recall is the notion of *communication complexity.* This measure corresponds to the number of communications between the component grammars

necessary to generate the language.
Let us denote by x-PCGS(n)-f(m) the PCGS of degree n with the communication graph x and at most $f(m)$ communications during the generation of any word of the length m. As usual, $\mathcal{L}(Y)$ denote the class of languages that are generable by the system of the type Y, $Y \in \{PCGS, x-PCGS, PCGS-f(m), \ldots\}$

3. Results.

In this section the results relating the computational power of PCGS to communication graphs and to communication complexity are presented. First, we separate some communication structures by showing which properties are sufficient and necessary for a communication graph to be able to generate a nonregular language over one-letter alphabet.

Theorem 1 *Let* Π *be a PCGS with the following three properties:*

(i) $G(\Pi)$ *contains at most one cycle*

(ii) if $G(\Pi)$ *contains a cycle, then this cycle involves the grammar* G_1

(iii) $G(\Pi)$ *generates a language over one-letter alphabet .*

Then $L(\Pi)$ *is a regular language.*

Proof. Let $\Pi = (G_1, G_2, \ldots, G_n)$ be a PCGS satisfying (i),(ii), and (iii). We shall describe a nondeterministic TM M simulating an derivation of Π in a constant memory. Let $w \in L(\Pi), \mathcal{D}(w) = C_0, C_1, \ldots, C_n$ be a derivation of w. Let s be the number of generative sections of $\mathcal{D}(w)$. Then

$$|w| = \sum_{\substack{0<i\leq m \\ 0<j\leq s}} n(i,j).|g(i,j)|,$$

where $n(i,j)$ is the number of the occurrences of the generative section $g(i,j)$ in w.

First, let us describe a possible simulation of Π **provided all n(i,j)'s are known.** Each state of M contains $(A_1, A_2, \ldots, A_m)$ - an actual nonterminal cut of Π. According to this nonterminal cut, either generative step or a communication step will be simulated as follows:
• a generative step in the j-th generative sequence
Let $A_i \to a^{l_i} M_i \in P_i$ for some $A_i \neq \varepsilon$, then M reads $\sum n(i,j) \cdot l_i$ symbols from the input and sets the new state to $(M_1, M_2, \ldots, M_m)$, where $M_i = A_i$ for those i, $A_i = \varepsilon$.
• a communication step
A communication step is simulated by setting a new state, that corresponds to the nonterminal cut of Π just after communication step of Π performed.

So, the crutial part of the simulation is nondeterministic guessing and deterministic checking of $n(i,j)$ in constant memory. In order to show how this works,we introduce the following formalism.

• Let $CS(i,j) = t_1, t_2, \ldots, t_{i_j}$ be a sequence of those time steps of the derivation $\mathcal{D}(w)$ that correspond to communication steps in which some occurrence of $g(i,j)$ is succesfuly transformed from one component grammar to onother one. Then the graph $\overline{\mathcal{G}}(i,j)$ representing motion of $g(i,j)$ during the computation $\mathcal{D}(w)$according to $CS(i,j)$ can be built up as follows:

1. take one vertex labeleld by i to $\overline{\mathcal{G}}(i,j)$, and call this node the source of $CS(i,j)$
2. for $l = t_1, t_2, \ldots, t_{i_j}$ do

   for every leaf f of $\overline{\mathcal{G}}(i,j)$ set Q(f)= { r|the communication nonterminal Q_f is at the r-th possition present at C_{t_l}};

   for every r $\in$Q(f) put a new node v_r with the label r to $\overline{\mathcal{G}}(i,j)$ as a son of f.

Observation 1 *n(i,j) is equal to the number of leaves of $\mathcal{G}$ with the label* 1.

• W.l.o.g. one can assume $G_1, G_2, \ldots, G_C$ are the vertices that form (unique) cycle in G(Π). Every path in $\overline{\mathcal{G}}(i,j)$ leading from the vertex i to the leaf 1 - so called *p(i)-path*, or *$i \rightarrow 1$ path* - can be written in the following form:
$\bar{\text{p}}$(i)=$i, i_2, \ldots, i_p, (1,2,\ldots,C)^*, 1$, where $i_s \neq 1, i_s \neq i_r$ for $s,r \in \{1,\ldots,p\}$.
Now, let $\mathcal{G}(i,j)$ be the graph that is obtained from $\overline{\mathcal{G}}(i,j)$ in such a way, that every $\bar{\text{p}}$(i) is rewriten to p(i)=$i, i_2, \ldots, i_p, (1,2,...,C), 1$.

Realizing that $\mathcal{G}(i,j)$ is at most m-ary tree (where m is the number of component grammars) with the depth at most $2m$, the following observations simply follow.

Observation 2 *$\overline{\mathcal{G}}(i,j)$ and $\mathcal{G}(i,j)$ have the same number of leaves labelled by* 1.

Observation 3 *$\forall i\ \exists d_i$ bounding the number of different $\mathcal{G}(i,j)$'s. Moreover, $\forall i\ \exists D_i$ $n(i,j) \leq D_i$. Constants d_i, D_i depend on G(Π), and are independent of $\mathcal{D}(w)$ (i.e., independent of the length of the word generated).*

Now, we are ready to describe the NTM aquivalent to $G(\Pi)$.

• **The set of states** of M consists of pairs (NC,o), where

NC is from the set of all possible nonterminal cuts of Π, and

o=$(o_1, o_2, \ldots, o_m) \in \{0,1,\ldots,d\}^m$. This last m-tuple o expresses actual values of $n(i,j)$ guessed.

• **Simulation** The computation of M corresponds to the following step-by-step simulation of Π.

(1) At the beginning of the simulation M nondeterministically sets its state to $((S_1, S_2, \ldots, S_m), (o_1, o_2, \ldots, o_m))$.

(2) Let $\mathcal{P}(i) = \{p(i) | p(i)$ is the i-1 path$\}$

- $\forall\, i$ such that $o_i = 0$ M writes $\forall p(i) \in$P(i) 'N-path p(i)'on the i -th memory tape with the first vertex marked $\forall p(i) \in \mathcal{P}(i)$
- $\forall\, i$ such that $o_i \neq 0$ Let $\mathcal{P}'(i), \mathcal{P}''(i)$ is a (nondeterministically guessed) decomposition of P(i), with $|\mathcal{P}'(i)| = o_i$.

  M writes 'Y-path p(i)' $\forall p(i) \in \mathcal{P}'(i)$ and
  M writes 'N-path p(i)' $\forall p(i) \in \mathcal{P}''(i)$ to its memory tape.

The above described nondeterministic guess partitioning $\mathcal{P}(i)$ into $\mathcal{P}'(i)$ and $\mathcal{P}''(i)$ means that M guesses $n(i,1)$ by guessing $n(i,1)$ i-1 pathes which have to be realized in the communication steps of $\mathcal{D}(w)$.

Generative step
Let $((A_1, A_2, \ldots, A_m), (o_1, o_2, \ldots, o_m))$ be a state of M indicating the generative step will be performed. Let $A_i \to a^{1_i} M_i \in P_i$ for $A_i \neq \varepsilon$. Then M sets new state of M to $((M_1, M_2, \ldots, M_m), (o_1, o_2, \ldots, o_m))$, where $M_i = A_i$ for $A_i = \varepsilon$, and M reads $\sum o_i * l_i$ symbols from the input. When the reading cannot be succesfuly performed (because of the absence of input symbols), M stops and rejects.

Communication step
Let $CC = (q_1, q_2, \ldots, q_m)$ be a communication cut associated with a communication step. Let $q_i = Q_{j_i}$.
The simulation of this step by M is described as follows:
• Modification of N-pathes
M has to distinguish four possibilities

- the first vertex on an N-path is $l \neq j_i$ - *no action*

- the first vertex on an N-path is j_i
  action: M removes j_i from N-path and if the next vertex is 1, then it rewrites this symbol by $(1, S)$

- the first vertex on an N-path is $(1, S)$, what indicates, the (only) cycle follows on this N-path and the $g(a, b)$ corresponding to this N-path is just located in G_1 at this moment.
  action: M rewrites $(1, S)$ by $(1, *)$ and rewrites the next vertex to the same one with the mark S

• the first vertex on an N-path is $(1, *)$, what indicates, this N-path is in other component grammar located at the moment just simulated. The grammar, it is located in, is by (c, S) marked.
action: M reads this N-path until (l, S) is found. If $l = j_i$ then it rewrites this symbol by $(l, *)$ and marks the next (according to a cycle) vertex by S.

• Modification of Y-pathes
There are two possibilities:

• There is at least one symbol j_i resp.(j_i, S) on some of the memory tapes present.
action: M removes the first vertex j from every memory tape, resp. rewrites (j_i, S) by $(j_i, *)$ and markes the next (according to a cycle) vertex with S. Since the new generative section starts for j_i-th grammar, M has to guess new value for o_{j_i} and sets Y-pathes and N-pathes according to (2).

• There is no symbol j_i resp.(j_i, S) on any of the memory tapes present - *no action.*

End of the simulation
M accept iff the next four conditions hold:

there is no nonterminal symbol present in component grammar G_1, resp. first component of nonterminal cut stored in the state of M

the input tape was succesfuly read

no N-path is empty, resp. with symbol $(1, S)$ on it

every Y-pathes are empty, resp. with symbol $(1, S)$ on them.

Since NTM working in the constant memory is equivalent to NFA, that recognize just the class of the regular languages, our proof is done.

□

Corollary 1 *Let* Π *be a x-PCGS,* $x \in \{cycle, dag\}$*, generating a language over one-letter alphabet. Then* $L(\Pi)$ *is a regular language.*

Corollary 2 L(PCGS) - L(x-PCGS) $\neq \emptyset$, $x \in \{cycle, dag\}$.

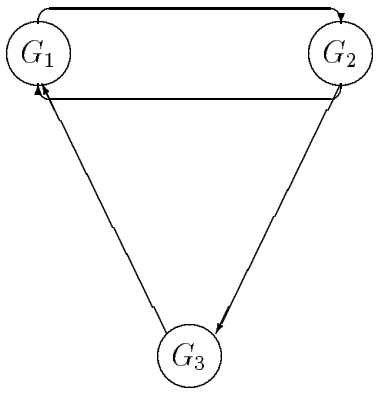
Proof. The language of Example 1 demonstrates that PCGS Π with $G(\Pi)$ containing only one cycle but G_1 outside it is able to generate nonregular language over one-letter alphabet.

According to [2] the language $L = \{a^{2^n} | n \in N\}$ can be generated by the following PCGS Π:

$P_1 : \{S_1 \rightarrow aB | Q_2, B_1 \rightarrow B, B_1 \rightarrow \varepsilon\}$

$P_2 : \{S_2 \rightarrow Q_1, B \rightarrow Q_3\}$

$P_3 : \{S_3 \rightarrow Q_1, B \rightarrow B_1\}$



The communication graph $G(\Pi)$ is the simplest one with two cycles.

□

Since there are examples of PCGS generating nonregular language over one-letter alphabet with $G(\Pi)$ containing two cycles, resp. one cycle that doesnot contain G_1, we have separated regular and nonregular languages over oneletter alphabet in terms of PCGS.

4. References

1. Gh.Paun,L.Santean, *Parallel communicating grammar systems; the regular case. Ann.Univ.Buc.Ser.Mat.-Inform.37* **vol. 2**(1989), pp.55-63.
2. L.Santean, *Parallel Communicating Systems. EATCS Bulletin* **42** (1990), pp 160 - 171.
3. L.Santean, J.Kari, *The Impact of the Number of Cooperative Grammars on the Generative Power, Theoretical Computer Science* (1992) pp.249-263.
4. J.Hromkovič, J.Kari, L.Kari, *Some hierarchies for the communication complexity measures of cooperating grammar systems. Theoretical Computer Science* to appear
5. D.Pardubská, *The communication complexity hierarchy of parallel communicating systems. Proceedings of IMYCS'92*