Centralized Parallel Communicating Non Synchronized Pure Pattern Grammar System with Filters

F. Amjad Basha\(^1\) and Sindhu J Kumaar\(^2\)

Research Scholar\(^1\) and Associate Professor\(^2\)

Department of mathematics,

B.S. Abdur Rahman University Vandalur,

Chennai -48, Tamil Nadu, India\(^1,2\)

Abstract: Motivated by the study of synchronized pure pattern grammar system by Sindhu et al [1], and the study of non synchronized pure pattern grammars by Amjad et al [2], parallel communicating grammar system by Gh.Paun et al [6] and parallel communicating grammar systems with communication by commands by Csuhaj Varju et al [5]. We begin with centralized parallel communicating non synchronized pure pattern grammar system, besides we proposed a novel approach called centralized parallel communicating non synchronized pure pattern grammar system with filter. Then we could focus and generalize previous results and obtain a sequence of new ones on a variant in communication. It generates a subclass of context free languages, which is compared with already existing languages.

Keywords: Communication by command, Filter, Non-synchronized pure pattern grammar and Parallel communication.

I. Introduction

Sindhu et al [1] studied synchronized pure pattern grammar which links the studies of pure grammars and pattern grammars. The synchronized Pure Pattern grammar has patterns which are the strings of constants or terminal symbols. The constants are replaced initially by axioms over terminal symbols. The process is continued by replacing at any step the symbols in a pattern with the current set of words derived, there by yielding the associated language. Amjad et al [2] has studied a variant in working of a synchronized pure pattern grammar which we call it as non synchronized pure pattern grammar.

Parallel Communicating grammar system with communication by commands by Csuhaj Varju et al [5] represents the first model of networks of language processor where communication is performed through filters. A rewriting step in this system is defined as follows: each grammar generates its own string until it has no more applicable productions. Then the components communicate their strings to each other in the following manner: every grammar tries to send a copy of its string to each of the other grammars, but only those strings are accepted at a component which passes through the filter associated with it. Two important classifications of parallel communicating grammar system concern the communication graph and returning features introduced by S. Demitrescu and G, Paun [7].

In this paper a new generative device, herein after referred as centralized parallel communicating non synchronized pure pattern grammar system with filter is introduced. In this grammar system, all the components except the master are non synchronized pure pattern grammar, but the master is regular or right linear grammar with filter.

II. Non Synchronized Pure Pattern Grammar

Definition 2.1: Sindhu et al

A Non Synchronized pure Pattern Grammar (NSPPG) is a triple where \(\Sigma\) is an alphabet \(A \subseteq \Sigma^*\) is a finite subset of \(\Sigma^*\) called axioms and \(P\) is a finite subset of \(\Sigma^*\) called the set of patterns. For a set \(P\) and language \(L \subseteq \Sigma^*\), let \(P(L)\) be the set of strings obtained by replacing uniformly and in parallel, each letter of all patterns in \(P\) by strings in \(L\), different occurrences of the same letter in a pattern being replaced by the same string. The difference in the working of a NSPPG is that at the \(r\)th step, each letter of the pattern is replaced by words from \(U_{r=0}^{P(A)}\) unlike in SPP where at the \(r\)th step each letter of the pattern is replaced by words \(P(r+1)(A)\).

The language (NSPPL) generated by \(G\) denoted by \(L(G)\) is the larger language \(L \subseteq \Sigma^*\) for which we have \(P \subseteq L\), \(A \subseteq L\), and \(P(L) \subseteq L\). In fact \(L(G) = P \cup A \cup P(A) \cup P(A \cup P(A)) \ldots\).

Example 2.1
\[ G = \{(a, b), (a), (ab)\}. \] In fact initially the axiom a replaces both a and b in the pattern to yield the word \(a^2\). In the subsequent step a and \(a^2\) are treated as an axiom which replaces both a and b in the pattern yielding \(\{a^2, a^3, a^4\}\) and the process continues. Here we get the language \(L(G_1) = \{a^n/n \geq 1\}\)

**Example 2.2**

\[ G = \{(a, b), (a), (ab)\}. \] Any of the axioms a, b can initially replace independently a as well as b in the pattern ab yielding \(\{a^2, b^2, ab, ba\}\) the resulting words along with a and b can be used in a similar manner in the pattern ab and the process is continued to yield the language consisting of all possible strings over a, b. Here we get the language \(L(G_2) = \Sigma^*\).

### III. Parallel Communicating Grammar System

A parallel communicating (PC) grammar system is a construct consisting of several usual grammars, working synchronously, each on its own sentential form, and communicating by request; special (query) symbols are provided, \(Q_i\), with the subscript identifying a component of the system; when a component \(j\) introduces a query symbol \(Q_i\), the current sentential form of the component \(i\) is sent to the component \(j\), where it replaces the occurrence(s) of \(Q_i\) in the sentential form of component \(j\). The language generated by a specified component of the system (the master), after a series of such rewriting and communication steps (each component starts from its axiom) is the language generated by the system.

**Definition 3.1**

A parallel communicating grammar system of degree \(n\), \(n \geq 1\) is a construct \(\Gamma = (N, T, K, (P_1, S_1), \ldots, (P_n, S_n))\) where \(N, T, K\) are pair wise disjoint alphabet, with \(K = \{Q_1, Q_2, \ldots, Q_n\}\), \(S_i \in N\) and \(P_i\) are finite set of rewriting rules over \(N \cup T \cup K, 1 \leq i \leq n\); the elements of \(N\) are non terminal symbols, those of \(T\) are terminals; the elements of \(K\) are query symbols; the pairs \((P_i, S_i)\) are the components of the system. Note that by their indices, the query symbols are associated with the components, For \((x_{i1}, \ldots, x_{in}), (y_{i1}, \ldots, y_{jn})\) with \(x_{ij} = y_{ij} \in (N \cup T \cup K)^*\), \(1 \leq i \leq n\); (we call such an \(n\)-tuple a configuration), we write \((x_{i1}, \ldots, x_{in}) \Rightarrow (y_{i1}, \ldots, y_{jn})\) if one of the two cases holds

1. If there is no query symbol in \(x_{i1}\) then \(x_{i1} \Rightarrow P_i y_{i1}\) or \(x_{i1} = y_{i1} \in T^*, 1 \leq i \leq n\):
2. If \(x_{i1}\) has a query symbol, we write such a string \(x_{i1}\) as \(x_{i1} = z_i Q_1 z_2 Q_2, \ldots, z_k Q_k z_{k+1}\) for \(t \geq 1, z_i \in (N \cup T)^*, 1 \leq i \leq t + 1\); then \(y = z_1 x_{i1} z_2 x_{i2}, \ldots, z_k Q_k z_{k+1}\) and \(y_{ij} = s_{ij}, 1 \leq j \leq t\); For all unspecified \(i\) we have \(y_{i} = x_{i}\)

Point (i) defines a rewriting step (component wise, synchronously, using one rule in all components whose current strings are not terminal).

Point (ii) defines a communication step; the query symbols \(Q_{ij}\) introduced in \(x_{i1}\) are replaced by the associated string \(y_{ij}\).

### IV. Parallel Communicating Grammars with Communication by Commands

Parallel communicating grammar systems with communication by commands by Csuhaj-Varju et al [4] represent the first model of networks of language processor where communication is performed through filters. A rewriting step in these systems is defined as follows: each grammar generates its own string until it has no more applicable productions. Then the components communicate their strings to each other in the following manner: every grammar tries to send a copy of its string to each of the other grammars, but only those strings are accepted at the component which passes through the filter associated with it.

**Definition 4.1: Csuhaj Varju et al**

A parallel communicating grammar system with communication by command and with finite sets of axioms or an FCCPC grammar system (of degree \(n\)) is a construct \(\Gamma = (N, T, (F_1, P_1, R_1), (F_2, P_2, R_2), \ldots, (F_n, P_n, R_n))\), \(n \geq 1\), where \(N\) and \(T\) are disjoint finite alphabets; \(N\) is called nonterminal alphabet and \(T\) is called terminal alphabet of the system, \((F_i, P_i, R_i), 1 \leq i \leq n\), is a component of \(\Gamma\), the \(i^{th}\) component, where \(F_i = (N \cup T)^*\) is a non empty finite set called the set of axioms of the components \(P_i\) is a finite set of rewriting rules over \((N \cup T)\) and \(R_i\) is a regular language \(R_i \subseteq (N \cup T)^*\), called the selector language or the filter of the \(i^{th}\) component. The first component is designated as the master component.

**Example 4.1**

Consider the FCCPC grammar with two components \(\Gamma = (N, T, (F_1, P_1, R_1), (F_2, P_2, R_2))\)

\[ N = \{A, B, C, A', B', C', A'' , B'', C'' , S, S'\}, T = \{a, b, c\}, F_1 = \{ABC\}, F_2 = \{aA\}, \]

\[ P_1 = \{A \rightarrow aA', B \rightarrow bB', C \rightarrow cC', A' \rightarrow \lambda, B' \rightarrow \lambda, C' \rightarrow \lambda\}, R_1 = \{a^*Ab^*Ba^*C + a^*A^*b^*C + C^*\}, F_2 = \{S\}\]

\[ P_2 = \{S \rightarrow S', A \rightarrow A, B \rightarrow B, C \rightarrow C, A' \rightarrow A'\} \]

\[ R_2 = a^*Ab^*Ba^*C^*\]

The first few steps of the derivation are as follows

\[(\{ABC\}, \{S\}) \Rightarrow (\{aA, bB, cC\}, \{S'\}) \Rightarrow \]

\[(\{aA, bB, cC\}, \{S'\}) \Rightarrow \]
Centralized Parallel Communicating Non Synchronized Pure Pattern Grammar System

In this section, we consider a variant in CsuhaJ Varju’s model called centralized parallel communicating non synchronized pure pattern grammar system using non synchronized pure pattern grammar system. In a centralized grammar system one component has filter and this component is designated as master component. In this model a regular or right linear grammar is considered as master component.

Definition 5.1
A Centralized PC (NSPPG) grammar system is a construct \( \Gamma = (N, T, K, (S_0, P_0), (A_1, P_1), ..., (A_n, P_n)) \) \( N = \) Set of non terminals, \( T = \) Set of terminals, \( K = \) Set of query symbols. Where \( N, T, K \) are pair wise disjoint alphabets with \( K = \{Q_1, Q_2, ..., Q_n\}, S_0 \in N, P_0 \) is a finite set of regular rules over \( N \cup T \cup K \); \( (A_i, P_i) \) are the components which are NSPPG over \( T \). The rewriting is similar to that of PC grammar systems with the following modifications. Initial configuration is \( (S_0, P_1, P_2, ..., P_n) \) where \( P_i \in P \) is a pattern of the \( i^{th} \) component. The rewriting in the component \( (A_i, P_i) \) is done according to NSPPG; that is any word \( P \) is a pattern of \( (A_i, P_i) \) is considered at any time. If the query symbol \( Q \) appears in the master component, then the string in the \( j^{th} \) component is communicated. After communicating, the components continue working from their axioms if in returning mode ‘r’ or the components continue the processing of the current strings if it is in non returning mode ‘nr’.

Example 5.1
\( \Gamma_1 = (N, T, K, (P_0, S_0), (A_1, P_1)) \) Where \( N = \{S_0, A\}, T = \{a, b\}, K = \{Q\}, P_0 = \{S_0 \rightarrow bA, A \rightarrow aA, A \rightarrow aQ\}, P_1 = \{ab\}, A = \{a\} \) \( L(\Gamma_1) = \{ba^n(a^2, a^3, ..., a^{2n+1})/n \geq 1\} \) Where \( y = ab \) if \( f = r \), \( y = a^2, a^3, ..., a^m \) if \( f = nr \) Here ‘r’ stands for returning mode and ‘nr’ stands for non returning mode. We now define a new generative device, namely parallel communicating non synchronized grammar system with communication by commands where the master component is regular with a filter and the other components are non synchronized pure pattern grammars.

Centralized Parallel Communicating Non Synchronized Pure Pattern Grammar System with Filter

In this section a parallel communicating non synchronized pure pattern grammar system with filter is introduced where the first component is designated as master component and the remaining components are non synchronized pure pattern grammar (NSP).

Definition 6.1
A CsyncNSPPG grammar system is a construct \( \Gamma = (N, T, (S_0, P_0, R), (A_1, P_1), ..., (A_n, P_n)) \) \( N = \) a set of non terminals, \( T = \) a set of terminals where \( N \) and \( T \) are disjoint, \( S_0 \in N, P_0 \) is a finite set of regular rules over \( N \cup T \), \( R = \) a regular language called the selector language or the filter of the component, \( Q \in N \) is a special symbol called query symbol. Each \( (A_i, P_i), i = 1, 2, 3, ... \) is a NSPPG over \( T \). The rewriting in the component \( (A_i, P_i) \) is done according to the NSPPG; that is any word \( P_i^n \) is considered at any time. If the query symbol appears in the master component then all the other components communicate their strings \( P_i^n \) to the master and master component will accept the strings which can pass through the filter associated with it. Let the set of strings sent by the \( i^{th} \) component to the master component be defined as \( \delta(L_i) = \{\lambda\} \) if \( L_i \cap R = \phi \) or \( i = 1 \), \( \delta(L_i) = \{L_i \cap R, \text{ otherwise} \} \), for \( 1 \leq i \leq n \)

Let \( \Delta = \delta(L_1) \delta(L_2) ... \delta(L_n) \) be the concatenation of the set of strings sent to the master component that is the total message received by the master component, and let \( L_1 = \{\Delta \} \) if \( \Delta \neq \{\lambda\} \) \( L_1 = \{\lambda\} \) if \( \Delta = \{\lambda\} \) After a communication step, the obtained language \( L_1 \) is either the concatenation of the received set of strings with the previous language or it is only previous language, when the components are not involved in communication.

Definition 6.2
The language \( L(\Gamma) \) generated by a CsyncNSPPG grammar system \( \Gamma = (N, T, (S_0, P_0, R), (A_1, P_1), ..., (A_n, P_n)) \) When \( n \geq 1 \) is defined as follows:
Example 6.1
\( I_1 = (N, T, (P_0, S_0, R), (A_1, P_1)) \) Where
\( N = \{S_0, A, Q\}; Q = \text{query symbol}, T = \{a, b\} \)
\( P_0 = \{S_0 \rightarrow aQ, S_0 \rightarrow aA, A \rightarrow aA, A \rightarrow aQ\}; R = \{a^{2m}/m \geq 1\}, P_1 = \{ab\}, A_1 = \{a\} \)
\((S_0, (ab)) \Rightarrow ((aA), (a^2)); ((a^2A), (a^2, a^3, a^4)) \Rightarrow ((a^3Q), (a^2, a^3, a^4, \ldots, a^9)) \rightarrow (a^3(a^2, a^3, a^4, \ldots, a^9)) \)
\(L(I_1) = \{a^n/n \geq 3\} \) Where \( y = ab \) if \( f = r; y = (a^2, a^3, a^4, \ldots, a^9) \) if \( f = nr \)

Example 6.2
\( I_2 = (N, T, (P_0, S_0, R), (A_1, P_1)) \) Where \( N = \{S_0, A, Q\}, T = \{a, b\}, Q = \text{query symbol} \)

\( P_0 = \{S_0 \rightarrow bA, A \rightarrow aA, A \rightarrow aQ\}, R = \{a^{2m}/m \geq 1\}, P_1 = \{ab\}, A_1 = \{a\} \)
\((S_0, (ab)) \Rightarrow ((bA), (a^2)); ((a^2A), (a^2, a^3, a^4)) \Rightarrow ((ba^2Q, (a^2, a^3, a^4, \ldots, a^9)) \rightarrow (ba^2(a^2, a^3, a^4, a^5), y) \)
\(L(I_2) = \{ba^n(a^2, a^3, a^4, \ldots, a^{2n+1})/n \geq 1\} \) \( y = ab \) if \( f = r; y = (a^2, a^3, a^4, \ldots, a^9) \) if \( f = nr \)

Example 6.3
\( I_3 = (N, T, (P_0, S_0, R), (A_1, P_1), (A_2, P_2), (A_3, P_3)) \) Where
\( N = \{S_0, Q\}, T = \{a, b\}, Q = \text{query symbol} \)
\( P_0 = \{S_0 \rightarrow aS_0, S_0 \rightarrow aQ\} \)
\( R = \{a^2)^2 + (b^2)^2 + (c^3)^3\}, P_1 = \{ab\}, A_1 = \{a\}, P_2 = \{aa\}, A_2 = \{b\}, P_3 = \{aaa\}, A_3 = \{c\} \)
\((S_0, (ab)) \Rightarrow ((aa), (a^2)); ((a^2S_0, (a^2, (b^2), (c^3)) \Rightarrow ((a^2S_0, (a^2, a^3, a^4), (b^4), (c^9)) \rightarrow (a^2a^2, a^3, a^4, a^5, b^2, c^9), y_1, y_2, y_3) \)
\(L(I_3) = \{a^n \{a^2, a^3, a^4\} b^n \{c^9\} /n \geq 1\} \) \( y = ab, aa, aaa \) if \( f = r; y_1 = \{a^2, a^4, \ldots, a^2n\}, y_2 = \{b^2n\}, y_3 = \{c^3n\} \) if \( f = nr \)

Note
If a pattern consists of different symbols with a \( \lambda \) free axiom, then obviously the language generated from the centralized parallel communicating non synchronized grammar system could be different from the centralized parallel communicating synchronized grammar system.

**Proposition 6.1**
1. CCPC (NSPPL) - PC (NSPPL) \( \neq \phi \)
2. CCPC (NSPPL) \( \cap \) PC (NSPPL) \( \neq \phi \)

**Proof**
1. This is seen from the example 6.2. The language \( \{ba^n(a^2, a^4, \ldots, a^{2n+1})/n \geq 1\} \) generated by \( I_2 \) is not a PC (NSPPL), because in PC (PPG) if a query symbol Q appears in the master component, then the string in the \( j^{th} \) component alone is communicated, as only one component can communicate to the master component at a time. But in CCPC (NSPPG), if a query symbol Q appears in the master component, then the strings of the entire set of components are communicated, if they can pass through the filter and are concatenated with the string generated in the master component. Thus the languages generated by CCPC (NSPPG) and PC (PPG) are incomparable.
2. This is seen from examples 5.1 and 6.2. The language can be generated by PC (NSPPG) and CCPC (NSPPL).

**Proposition 6.2**
The class of CCPC (NSPPG) languages in the returning mode coincides with non - returning mode if the master component is regular or linear.

**Proof**
Let \( I = (N, T, (P_0, S_0, R), (A_1, P_1), (A_2, P_2), \ldots, (A_n, P_n)) \) where \( (P_0, S_0, R) \) is regular with a filter. The number of non terminals in the right side of each rule is one. Hence when the designated terminal of the master component is replaced by a string from the \( i^{th} \) component after passing through the filter, we get a terminal word. No further generation is possible along this line. Hence there is no difference between the returning and non - returning modes.
Proposition 6.3
In a parallel communicating non synchronized pure pattern grammar system (PCNSPPG) if the non synchronized pure pattern grammar components contain patterns of different letters then
1. \( RL - PC (\text{NSPPL}) \neq \emptyset \)
2. \( RL \cap PC (\text{NSPPL}) \neq \emptyset \)

Proof
1. Let \( L = \{w/w \text{ is in } (a, b)^* \text{ and } w \text{ consists of an even number of } a\text{'s and even number of } b\text{'s. This language is a regular language generated by right linear grammar } G = (N, T, P, S) \text{ where } N = \{S, A, B, C\}, T = \{a, b\}, P \text{ consists of the rules} \)
   1. \( S \rightarrow aA \)
   2. \( A \rightarrow aS \)
   3. \( S \rightarrow bB \)
   4. \( B \rightarrow bS \)
   5. \( A \rightarrow bC \)
   6. \( C \rightarrow aB \)
   7. \( B \rightarrow aC \)
   8. \( A \rightarrow a10 \)
   9. \( B \rightarrow b. \)
   But \( L \notin PC (\text{NSPPL}) \) since it is non synchronized and even filter is not being used here so it is not possible to get only even number of \( a\)’s and \( b\)’s in parallel communicating non synchronized pure pattern languages. Therefore both \( RL \) and \( PC (\text{NSPPL}) \) are incomparable.
2. Let \( L = \{a^n \mid n \geq 3\}. \)This language can be generated by both \( RL \) and \( PC (\text{NSPPL}). \)

VII. Conclusion
As expected the result of the above combination is a very powerful mechanism. In this model we have used one filter and only one query symbol. Henceforth if the model is adapted by introducing more number of query symbols then the new variant might be of interest.

REFERENCES