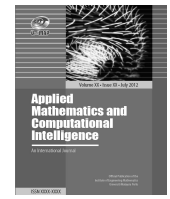
App. Math. and Comp. Intel., Vol. 2(2) (2013) 195–203
http://amci.unimap.edu.my



Languages defined by pure patterns

Sindhu J Kumaar[a], P.J. Abisha[b], D. Gnanaraj Thomas[b], Nor Haniza Sarmin[c], K.G. Subramanian[d,*]

[a] *Department of Mathematics*
B.S. Abdur Rahman University
Chennai 600048, India

[b] *Department of Mathematics, Madras Christian College*
Tambaram, Chennai 600059, India

[c] *Department of Mathematical Sciences, Faculty of Science*
Universiti Teknologi Malaysia
81310 UTM Johor Bahru, Johor, Malaysia

[d] *School of Computer Sciences*
Universiti Sains Malaysia
11800 USM, Penang, Malaysia

Received: 24 May 2013; Accepted: 4 September 2013

Abstract: *Angluin's pattern languages that use pattern strings as language descriptors, have motivated researchers in formal language theory to introduce and investigate grammars based on patterns. Both theoretical as well as application oriented properties of these grammars have been studied. On the other hand pure grammars in line with the early works of Thue have been proposed and studied for their properties. Here we introduce a new kind of language generative device, called a pure pattern grammar, linking the notions of pattern and pure grammars. Two modes of derivation in such a pattern grammar, called as synchronized and non-synchronized modes, are introduced. The resultant families of languages are compared for their generative power with certain other well-known families of languages. Certain closure properties and descriptional complexity results are also obtained.*

Keywords: *Formal language; Pattern grammar; Pure grammar.*

PACS: *02.70.-c, 89.20.Ff*

*Corresponding Author: kgsmani1948@gmail.com (K.G. Subramanian)

1 Introduction

Formal language theory [1, 2], which is one of the foundation areas of theoretical computer science, is replete with an abundance of grammars that have been introduced and investigated with different motivations. The pure grammar introduced in [3] is more in line with the early work of Thue on words [4] in the sense of not dividing the alphabet into terminals and non-terminals unlike the well-known [1] Chomskian grammars. Also in contrast to the well-investigated L systems [5] which involve rewriting in parallel, the rewriting process in a pure grammar is sequential as in the Chomskian grammars. A number of investigations on pure grammars in terms of theoretical properties and applications has been done in the literature (See for example [6, 7, 8, 9, 10]).

On the other hand a different kind of language generative model, called pattern grammar, was introduced in [11] motivated by Angluin's pattern languages that use pattern strings as language descriptors. Investigation of patterns has been of relevance in many areas such as combinatorics on words, learning theory and so on. Unlike the classical models of grammars and automata, pattern grammars are known to provide an alternative method in defining languages. This grammar involves an operation of replacing in parallel all variables in a pattern string by a specific set of strings. The replacement is done in a uniform way in the sense of replacing all occurrences of the same variable in a pattern by the same string. This kind of grammar takes motivation from the study of Angluin [12] on patterns describing a set of strings. Pattern grammars have been subsequently investigated from different points of view (See for example [13, 14, 15, 16, 17, 18]).

In this paper we consider a new generative device known as a pure pattern grammar, which was originally introduced in [19]. This provides a natural link between pure grammars [3] and pattern grammars [11] which had motivations from different directions. Pure pattern grammars have found application as well in the problem of learning of languages [20]. The pure pattern grammar has only one kind of symbol, namely, terminal symbol or constant, as in pure grammars. Generation of words involves a process that is analogous to that in a pattern grammar. In other words, the pure pattern grammar has patterns which are 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. We introduce two modes of working of a pure pattern grammar which we call as synchronized and non-synchronized modes. We compare the resultant families of synchronized and non-synchronized pure pattern languages with other families of languages such as pattern languages [11], pure languages [3], Chomskian languages [2]. Certain closure properties and descriptional complexity results are also obtained.

2 Preliminaries

We recall some needed definitions. For unexplained notions and notations, we refer to [5, 2].

An alphabet Σ is a finite set of symbols. A word over Σ is a finite sequence of symbols of Σ. The set of all words over Σ is denoted by Σ^* which includes the empty word λ. We write $\Sigma^+ = \Sigma^* - \{\lambda\}$.

Definition 1. *A pure grammar [3] is a triple $G = (\Sigma, P, S)$ where Σ is a finite alphabet, S is a finite set of words over Σ and P is a finite set of ordered pairs (x, y) of words over Σ. The elements of P are referred to as productions, usually written as $x \to y$. If $x \in \Sigma$, in every production $x \to y$ of P, then G is called a pure context-free grammar (PCF).*

In a pure grammar, a word w over Σ yields directly a word w' over Σ according to G if there are words $w_1, w_2 \in \Sigma^*$ and a production $x \to y$ in P such that $w = w_l x w_2$ and

$w' = w_l y w_2$. We then write $w \Rightarrow_G w'$ or briefly $w \Rightarrow w'$ (if G is understood). The reflexive, transitive closure of $\Rightarrow$ is denoted by $\Rightarrow^*$. The language $L(G)$ generated by G, called a pure language, is defined as $L(G) = \{w \mid s \Rightarrow^* w$, for some s in $S\}$. The language generated by a PCF grammar is called a PCF language.

Example 1. *The PCF grammar $G = (\{a, b\}, \{a \to ab\}, \{a\})$ generates the PCF language $L(G)$ consisting of all words of the form $ab^n, n = 0, 1, 2, \cdots$.*

Though in both the Chomskian and pure grammars, the rewriting process is sequential, it is known [3] that these two families of languages are incomparable.

Definition 2. *A pattern grammar [11, 15] is a 4-tuple $G = (\Sigma, X, A, P)$ where Σ is an alphabet whose elements are called constants, X is an alphabet whose elements are called variables, $A \subseteq \Sigma^*$ is a finite set of words, called axioms, $P \subseteq (\Sigma \cup X)^*$ is a finite set of words, called patterns where each pattern contains at least one variable.*

The rewriting in G is defined as follows: Initially, words are obtained by replacing in parallel and uniformly all the variables in a pattern in P by axioms of A with different occurrences of the same variable being replaced by the same word. The process is continued in a similar way by replacing variables by words of the current set of strings obtained. The language generated by G is $L(G) = A \cup P(A) \cup P(P(A)) \cup \ldots$, where $P(X)$ denotes the set of strings obtained from patterns in P by using strings of the set X in the manner described above.

Example 2. *$G = (\{a, b\}, \{\delta\}, \{ab\}, \{a\delta b\})$ is a pattern grammar generating the language $L(G)$ consisting of all words of the form $a^n b^n, n = 1, 2, \cdots$. In fact $A = \{ab\}$ and initially the axiom word ab replaces δ in the pattern $a\delta b$ to yield aabb. The process is repeated. Thus $P(A) = \{aabb\}$, $P(P(A)) = \{aaabbb\}$ and so on.*

It is known [11] that the pattern grammars generate a family of languages incomparable with Chomskian languages [2] and Lindenmayer languages [5].

3 Synchronized Pure Pattern Grammars

In this section, we consider the notion of a synchronized pure pattern grammar ($SPPG$). Pure pattern grammar (PPG) was introduced in [19] linking the studies of pure grammars [3] and pattern grammars [11]. Here we call the PPG as a synchronized pure pattern grammar ($SPPG$) and recall the grammar.

Definition 3. *A synchronized pure pattern grammar ($SPPG$) is a triple $G = (\Sigma, A, P)$ where Σ is an alphabet, $A \subseteq \Sigma^*$ is a finite nonempty set of elements of Σ^*, called axioms and P is a finite nonempty subset of Σ^+, called the set of patterns. For a set of words $X \subseteq \Sigma^*$, let $P(X)$ be the set of strings obtained by replacing uniformly and in parallel, all the letters in every pattern of P, by strings in X. Different occurrences of the same letter in a pattern are replaced by the same string.*
Initially the symbols in a pattern are replaced by the axioms and subsequently the replacement process is continued with the words obtained at the current step. The synchronized pattern language ($SPPL$) generated by G, denoted by $L(G)$, is the smallest language $L \subseteq \Sigma^*$ for which we have $P \subseteq L$, $A \subseteq L$ and $P(L) \subseteq L$. In fact $L(G) = P \cup A \cup P(A) \cup P(P(A)) \cup \ldots$.
We denote by $SPPL$ itself the family of languages generated by $SPPGs$.

Note that the patterns in a $SPPG$ are also in the language of the grammar.

Example 3. *We illustrate with some $SPPGs$ and the corresponding languages generated.*

i) $G_1 = (\{a\}, \{a\}, \{aa\})$. *Here* $L(G_1) = \{a^{2^n}/n = 0, 1, 2, \cdots\}$. *In fact initially the axiom* a *replaces both* $a's$ *in the pattern* aa *to yield the word* a^2 *which is then used to replace again both the* $a's$ *in the pattern* aa *giving* a^4 *and the process continues.*

ii) $G_2 = (\{a, b\}, \{\lambda, a, b\}, \{ab\})$. $L(G_2) = \Sigma^*$. *Any of the axioms* λ, a, b *can initially replace independently* a *as well as* b *in the pattern* ab *yielding* $\lambda, a, b, aa, ab, ba, bb$. *The resulting words can be used in a similar manner in the pattern* ab *and the process can be continued to yield the language consisting of all strings over* a, b *including the empty string.*

iii) $G_3 = (\{a, b\}, \{a, b\}, \{aaa\})$. $L(G_3) = \{a^{3^n}/n = 0, 1, 2, \cdots\} \cup \{b^{3^n}/n = 0, 1, 2, \cdots\}$. *Again initially the axiom* a *replaces all the* $a's$ *in the pattern* aaa *yielding* a^3. *Likewise the axiom* b *yields* b^3. *It can be seen that only powers of* a^3 *or* b^3 *are generated subsequently.*

iv) $G_4 = (\{a, b\}, \{aa, bb\}, \{aa\})$.
$L(G_4) = \{(aa)^{2^n}/n = 0, 1, 2, \cdots\} \cup \{(bb)^{2^n}/n = 0, 1, 2, \cdots\}$.

Lemma 1. *Any* $\lambda-$*free finite non-empty set* F *is a synchronized pure pattern language if and only if* F *contains atleast one word of length one.*

Proof. If a word p of length one is in the finite set F, $\lambda \notin F$, then a $SPPG$ G generating F has the axiom set F itself and the singleton set $\{p\}$ is the set of patterns of G. Conversely, let $F = \{w_1, w_2, ..., w_m\}$ be a finite set over an alphabet Σ and $|w_i| > 1$, for $i = 1, \cdots, m$. Suppose F is a $SPPL$ generated by a $SPPG$ G_F with alphabet Σ, axiom set F and the set P of patterns. Then P contains at least one $w_j \in F$ and A atleast one $w_k \in F$. Let $|w_j| = l$. Then G_F generates a language L containing the infinite set $\{w_k^l, w_k^{l^2}, ...\}$ which is a contradiction. □

Remark 1. *i) Note that we can not relax the condition of* F *being* $\lambda-$*free in the Lemma 1, since the set* $\{\lambda, a^2\}$ *is a* $SPPL$ *generated by a* $SPPG$ *with axiom* λ *and pattern* aa.
ii) It can be noticed from examples $3(i), 3(iii)$ *or* $3(iv)$ *that if* $G = (\Sigma, A, P)$ *is a SPPG with* P *containing at least one pattern with length greater than or equal to 2 and all the axioms being nonempty, then* $L(G)$ *is not always a context-free language.*

Lemma 2. *If* $L \in SPPL$, *then there is a constant* k *such that for all* $z \in L$ *with* $|z| > k$, *we can write* $z = u_1 u_2 \dots u_n$, $u_i \in L$, $i = 1, 2, \dots n$ *where not all* u_i *need be distinct.*

Proof. Let $G = (\Sigma, A, P)$ be such that $L(G) = L$. Take $k = max\{|x|/x \in A \cup P\}$. If $z \in L$ and $|z| > k$ then $z \notin A$ and $z \notin P$. Therefore z must have been obtained from a pattern $p \in P$ using strings in $P^r(A)$ for some r more than 1. If $|p| = n$, then $z = u_1 u_2 \dots u_n$ where $u_i \in P^r(A)$, $i = 1, 2, \dots, n$. But then $u_i \in L$. □

Lemma 3. *If* $L \in SPPL$ *is an infinite language over* Σ, *then there is* $u \in \Sigma^+$ *such that for all* $n > 1$, *a string of the form* u^{k^n} *is in* L, *where* k *is the length of a pattern.*

Proof. Let $G = (\Sigma, A, P)$ be a $SPPG$ generating L. As L is infinite, P must contain at least one pattern p with $|p| > 1$. Let $|p| = k$ and $p = a_1 a_2 \dots a_k$, $a_i \in \Sigma$. Here an a_i may be equal to a_j for some i and j. If $u \in A \cap \Sigma^+$ is an axiom, then $L(G)$ contains all strings of the form u^{k^n}. This can be proved using induction on n. As $u \in A$, $P(A)$ contains a string of the form $u^k = u^{k^n}, n = 1$ which is obtained by replacing all a_i's in $p = a_1 a_2 \dots a_k$ by u. Let us assume that there is a $u \in A$ such that $u^{k^m} \in L$. We need to prove that $u^{k^{m+1}} \in L$. since $u^{k^m} \in L$ we have $u^{k^m} \in P^r(A)$ for some $r > 1$. Therefore $P^{r+1}(A) = P(P^r(A))$ will contain $u^{k^{m+1}}$. This is obtained by replacing every a_i in $p = a_1 a_2 \dots a_k$ by u^{k^m} to get $u^{k^m} u^{k^m} \dots u^{k^m} = u^{k^m} k = u^{k^{m+1}}$. This completes the induction. □

Theorem 1. *The family of synchronized pure pattern languages* $(SPPL)$ *is incomparable but not disjoint with*

1. *the family of pattern languages*
2. *the family of pure context free languages*
3. *the family of regular and hence with the family of context-free languages.*

Proof. The incomparability of $SPPL$ with the family of pattern languages is seen by considering the language $L_1 = \{a^n b^n / n = 1, 2, \cdots\}$ which is a pattern language (example 2). But if L_1 is generated by a $SPPG$, then there must be atleast one pattern in the $SPPG$ with length greater than or equal to 2 for, otherwise the set generated is finite or might contain λ. Also the axioms of the $SPPG$ are in $\{a^n b^n / n = 1, 2, \cdots\}$. Hence if $a^j b^j$ is an axiom and p is a pattern with $|p| > 2$ then words having ba as a subword are also generated, which is a contradiction. Therefore L_1 is not in $SPPL$. On the other hand $L_2 = a^* \cup \{ab\}$ is generated by the $SPPG$, $G = (\{a, b\}, \{\lambda, a\}, \{ab\})$. But L_2 is not a pattern language [11].

For proving the incomparability with the family of pure context free languages, we note that the language $L_3 = \{ab^n / n = 1, 2, \cdots\}$ is generated by the pure context free grammar $(\{a, b\}, \{a \to ab\}, \{ab\})$. But L_3 can not be generated by any $SPPG$. For, if L_3 is generated by a $SPPG$, then there must be atleast one pattern p in the grammar with $|p| > 1$ and so words with ba as a sub word will be generated. On the other hand $L_4 = \{a^{2^n} / n = 0, 1, 2, \dots\}$ is in $SPPL$ (example 3(i)). But pure context-free languages are included in the family of context-free languages [3]. Hence L_4 being a non context-free language, is not a pure context-free language.

The incomparability with the families of regular and context-free languages follows from the fact that the language $\{a^2, a^3\}$ is regular (in fact finite) but is not in $SPPL$ by Lemma 1, whereas the language $\{a^{2^n} / n = 0, 1, \cdots\}$ is in SPPL but is known [2] to be non-context-free.

Finally, we note that the families of pure context-free languages and pattern languages and $SPPL$ are not disjoint as $\{a^n / n = 0, 1, 2, \cdots\}$ is a member of all the three families. □

Theorem 2. *The family SPPL is not closed under i) Catenation ii) Intersection iii) Intersection with regular sets iv) Kleene Closure v) Union.*

Proof. i) Consider the languages $K_1 = \{a^{2^n} / n = 0, 1, \cdots\}$ generated by the $SPPG$ $G_1 = (\{a\}, \{a\}, \{aa\})$ and $K_2 = \{a, a^3\}$ generated by the $SPPG$ $G_2 = (\{a\}, \{a^3\}, \{a\})$ so that

$$K_1 K_2 = \{a^{2^n+1}, a^{2^n+3} / n = 0, 1, 2, \cdots\}.$$

The empty word λ is not in $K_1 K_2$. Hence the word a^2 which is in $K_1 K_2$ has to be an axiom or a pattern if $K_1 K_2$ is generated by a $SPPG$. In any case this will give rise to several words over a with even powers of a. But only a^2 and a^4 with even powers are in $K_1 K_2$. For all other words in $K_1 K_2$, the words are over a and the powers of a are odd. Therefore $K_1 K_2$ is not in $SPPL$.

ii) The language $K_3 = \{ab\} \cup a^*$ is generated by the $SPPG$
$G_3 = (\{a, b\}, \{\lambda, a\}, \{ab\})$ and the language $K_4 = \{ab\} \cup b^*$ is generated by the $SPPG$ $G_4 = (\{a, b\}, \{\lambda, b\}, \{ab\})$. Both K_3 and K_4 are in $SPPL$ but $K_3 \cap K_4 = \{ab\}$ which is not in the family $SPPL$, by Lemma 1.

iii) Let $K_5 = \{\lambda, ab, a^2\} \cup \{a^{2n-1} / n = 1, 2, \cdots\}$ which is a regular language generated by the regular grammar $(\{S, A\}, \{a, b\}, P, S)$ where P consists of the rules $S \to \lambda$, $S \to ab$, $S \to a^2$, $S \to a^2 A$, $A \to a, A \to a^2 A$. The language $K_6 = \{a^{2^n} / n = 1, 2, \cdots\}$ is also a $SPPL$, generated by the $SPPG$ $(\{a\}, \{aa\}, \{aa\})$. But $K_5 \cap K_6 = \{a^2\}$ which is not in $SPPL$ again by Lemma 1.

iv) The language K_6 in *iii)* above is a $SPPL$. The Kleene Closure of K_6 is the language $K_6^* = \{a^{2n} / n = 1, 2, \cdots\}$. If K_6^* is generated by a $SPPG$ G, then any axiom in G is of the form a^{2p} and any pattern is of the form a^{2r} so that only words of the form a^{4rp} will be

generated. But this would mean words like a^6, a^{10}, a^{14} which are in K_6 cannot be generated. For example if $4rp = 6$, then $2rp = 3$ which does not hold as $2rp$ is even.

v) Let $K_7 = \{a^{3^n}/n = 1, 2, \cdots .\}$ The union language $K_6 \cup K_7 \notin SPPL$ with K_6 in *iii*) above. The reason is that there cannot be a $SPPG$ with a pattern set which generates words which are powers of a with the powers being only multiples of 2 or multiples of 3. □

Various descriptional complexity measures of language generating mechanisms have been studied by different researchers and continue to be of interest (See for example [21] and references therein). Certain descriptional complexity measures [22] for synchronized pure pattern grammar (SPPG) are now obtained. We recall these measures but define them with respect to $SPPG$.

For a SPPG $G = (\Sigma, A, P)$,
$Ax(G) = card\ A$,
$Pat(G) = card\ P$, where $cardX$ is the number of elements in the finite set X;
$LAx(G) = max\{|w| : w \in A\}$,
$LPat(G) = max\{|w| : w \in P\}$.

Extending these measures to languages we have the following: For a $SPPL\ L$,

$$M(L) = inf\{M(G) : L = L(G)\}, M \text{ is any of } Ax, Pat, Lax \text{ or } LPat.$$

A measure M is said to be connected [22] if for all $n = 1, 2, \cdots$ there is a L_n in $SPPL$ such that $M(L_n) = n$.

Theorem 3. *Each of the measures, $Ax, Pat, LAx, LPat$ is connected with respect to the family $SPPL$.*

Proof. For each $n = 1, 2, \cdots$, the language $B_n = \{a, a^2, a^3, \ldots, a^n, a^{n+1}\}$ is a $SPPL$. Then $Ax(B_n) = n$, for any $SPPG\ G$ generating B_n must have all the n words $a^2, \cdots, a^{n+1}$ in the axiom set A and cannot have less than these n words. Also, for each $n = 1, 2, \cdots$, the language $C_n = \{a, a^2, a^3, \ldots, a^n\}$ is a $SPPL$. Then $LAx(C_n) = n$, for any $SPPG\ G$ generating C_n should have a^n in the axiom set A and no other word larger in length. Hence Ax, LAx are connected.

Likewise, for each $n = 1, 2, \cdots$, the language $D_n = \{a^n\} \cup \{\lambda\}$ is generated by the $SPPG$ $G = (\{a\}, \{\lambda\}, \{a^n\})$. Note that λ cannot be a pattern. Then $Pat(D_n) = LPat(D_n) = n$. Hence $Pat, LPat$ are also connected. □

4 Non-synchronized Pure Pattern Grammars

In this section pure pattern grammar working in a non-synchronized mode is introduced. The two modes of working: synchronized and non-synchronized, are then compared.

Definition 4. *A non-synchronized pure pattern grammar ($NSPPG$) is a triple $G = (\Sigma, A, P)$ where Σ is an alphabet $A \subseteq \Sigma^*$ is a finite nonempty set of elements of Σ^* called axioms and P is a finite nonempty subset of Σ^+ called the set of patterns. The difference in the working of a $NSPPG$ is that at the rth step, each letter of a pattern is replaced by words from $\bigcup_{i=0}^{r-1} P^i(A)$ unlike in $SPPG$ where at the rth step each letter of the pattern is replaced by words from $P^{r-1}(A)$. In other words we start with the axiom set A and use the words of the axiom set in the replacement of symbols in a pattern in P to obtain $P(A)$. We then use the words in $A \cup P(A)$ to obtain $P(A \cup P(A))$ and the process is continued. Thus the languge of the $NSPPG\ G$ is $L(G) = P \cup A \cup P(A) \cup P(A \cup P(A)) \cup P(A \cup P(A) \cup P(A \cup P(A))) \cup \ldots$.*

We denote the family of languages generated by $NSPPG$ by $NSPPL$.

Remark 2. *Note that there is no difference in the components of a $SPPG$ and a $NSPPG$. The difference lies only in the working.*

Example 4. *We give some $NSPPGs$ and describe the corresponding languages generated.*

i) Consider the $NSPPG$ $G_1 = (\{a\}, \{a\}, \{aa\})$, $L(G_1) = \{a^{2^n}/n = 0, 1, \cdots\}$. In the first step, the axiom a replaces each of the two $a's$ in the pattern aa to yield a^2. In the next step the words a, a^2 are used and the process repeated to yield a^2, a^4. The process continues. Note that this grammar in the synchronising mode generates the same language.

ii) Consider the $NSPPG$ $G_2 = (\{a, b\}, \{\lambda, a, b\}, \{ab\})$. $L(G_2) = \Sigma^*$. Note that starting with $\{\lambda, a, b\}$ and working in the non-synchronising mode we obtain $\{\lambda, a, b, ab, ba, aa, bb\}$. In the subsequent step we obtain the words
$\{\lambda, a, b, ab, ba, aa, bb, aab, aba, aaa, abb, bab, bba, baa, bbb, abab, abba, abaa, abbb,$
$baab, baba, baaa, babb, aaab, aaba, aaaa, aabb, bbab, bbba, bbaa, bbbb\}$.
Here again there is no difference in the language obtained while working in the synchronized mode.

iii) Consider the $NSPPG$ $G_3 = (\{a, b\}, \{a\}, \{abb\})$. $L(G_3) = \{abb\} \cup \{a^{2n-1}/n = 1, 2, 3, \dots\}$. In the first step, using a we obtain a^3 and in the next step we obtain a^3, a^5, a^7, a^9. Note that if we work in the synchronized mode we can obtain only a^9. In the subsequent step in the non-synchronized mode we obtain $a^{11}, a^{13}, a^{15}, a^{17}, a^{19}, a^{21}, a^{23}, a^{25}, a^{27}$. Again note that the languge obtained in the synchronized mode in this grammar is $\{a^{3^n}/n = 1, 2, \cdots\}$.

Remark 3. *Note that Lemma 1 holds in the case of non-synchronized pure pattern languages as well.*

Lemma 4. *If the patterns are over a single letter then the families of $NSPPL$ and $SPPL$ are the same.*

Proof. Let $G = (\Sigma, A, P)$ be a pure pattern grammar with patterns in P over a single letter. While working in the synchronized mode or non-synchronized mode, the same word replaces all the letters in a pattern as the pattern involves only the same symbol. So in the non-synchronized mode using the words obtained in a prior step does not make a difference as each of these can only be independently used. But this amounts to working in synchronized mode as well. Therefore $SPPL$ coincides with $NSPPL$. □

Lemma 5. *If a pure pattern grammar G does not include λ as an axiom and if a pattern consists of different symbols, then the languages obtained from the non-synchronized mode could be different from the synchronising mode.*

This result is noticed from example 4(iii).

Theorem 4. $NSPPL - SPPL \neq \phi$.

Proof. The language $L = \{abb\} \cup \{a^{2n-1}/n = 1, 2, 3, \dots\}$ is in the family $NSPPL$ as seen in example 4(iii) but it cannot be generated by any $SPPG$. In fact, if we work in the synchronized mode, λ cannot be an axiom, since λ is not in L and hence $a \in L$ has to be an axiom. But we cannot derive words such as a^5, a^7 which are in the language, as the axiom a will be used only in the first step yielding a^3 and will be unavailable for use in the subsequent steps in the synchronized mode of working. □

Theorem 5. *The family of non-synchronized pure pattern languages ($NSPPL$) is incomparable with*
i) the family of pattern languages (PL) and
ii) the family of pure context free languages (PCF).

The proof of this result is similar to the proof of Theorem 1.

Remark 4. *Closure results for $NSPPL$ can be obtained as done for $SPPL$ in Theorem 2.*

5 Conclusion

We have considered here two modes of derivation of pure pattern grammar and obtained theoretical properties such as closure and comparison results. Other properties such as decidability results remain for future study.

Acknowledgments

The authors gratefully acknowledge the comments of the reviewers. The author K.G. Subramanian gratefully acknowledges support for this research from a FRGS grant No. 203/PKOMP/6711267 of the Ministry of Higher Education (MoHE), Malaysia.

References

[1] G. Rozenberg and A. Salomaa (Eds.) *Handbook of formal languages: Vol. 1. Word, language, grammar,* Springer-Verlag, Berlin, 1997.

[2] A. Salomaa. Formal languages, Academic Press, New York, 1973.

[3] H.A. Maurer, A. Salomaa and D. Wood. Pure grammars. *Inform. Control,* 44:47–72, 1980.

[4] J. Berstel. Axel Thue's Papers on Repetitions in Words: a Translation, *Publications du LaCIM, Dpartement de mathmatiques et d'informatique 20, Universit du Qubec Montral*, 1995.

[5] G. Rozenberg and A. Salomaa. The Mathematical theory of L-systems, Academic Press, New York, 1980.

[6] A. Gabrielian. Pure grammars and pure languages. *Int. J. Comput. Math.,* 9:3–16, 1981.

[7] G. Georgescu. On the index of pure context-free grammars and languages. *In: G. Rozenberg, (Ed.): Developments in Language Theory, At the Crossroads of Mathematics, Computer Science and Biology, World Scientific, Singapore*, pages 60-69, 1994.

[8] E. Mäkinen. On Szilard languages of pure context-free grammars. *J. Inf. Process. Cybern.,* 22:527-532, 1986.

[9] M. Novotný. Construction of pure grammars. *Fundam. Inform.,* 52:345-360, 2002.

[10] K. G. Subramanian, R. M. Ali, M. Geethalakshmi, A. K. Nagar. Pure 2D picture grammars and languages. *Discrete Appl. Math.,* 157:3401-3411, 2009.

[11] J. Dassow, Gh. Păun and A. Salomaa. Grammars based on patterns. *Int. J. Found. Comp. Sci.,* 4:1–14, 1993.

[12] D. Angluin. Finding patterns common to a set of strings. J. Comp. System Sci., 21:46–62, 1980.

[13] G. Castiglione, A. Restivo and S. Salemi. Patterns in words and languages. *Discrete Appl. Math.,* 144:237–246, 2004.

[14] H. C. M. Kleijn and G. Rozenberg. On the generative power of regular pattern grammars. *Acta Inform.,* 20:391–411, 1983.

[15] Gh. Păun, G. Rozenberg and A. Salomaa. Pattern Grammars. *Journal of Automata, Languages and Combinatorics,* 1:219–235, 1996.

[16] V. Mitrana. Iterated pattern languages. *J. Autom. Lang. Comb.,* 1:305–311, 1996.

[17] V. Mitrana, Patterns and languages: An Overview. *Grammars,* 2:149–173, 1999.

[18] V. Mitrana, Gh. Păun, G. Rozenberg and A. Salomaa. Pattern systems. *Theoret. Comput. Sci.,* 154:183–201, 1996.

[19] P.J. Abisha, K.G. Subramanian and D.G. Thomas. Pure pattern grammars. *Proc. Int. Workshop on Grammar Systems*, Austria, pages 253–262, 2000.

[20] P.J. Abisha, D.G. Thomas, and S. J. Kumaar. Learning subclasses of pure pattern languages.*Proc. Int. Colloquium on Grammatical Inference (ICGI 2008), LNCS 5278* pages 280-282, 2008.

[21] S. Turaev, G. Mavlankulov, M. Othman, and M. H. Selamat. Descriptional complexity of lindenmayer systems. *App. Math. and Comp. Intel.,* 1:12–23, 2012.

[22] J. Gruska. The descriptional complexity of context-free languages. *In Proc. MFCS Symp.*, pages 71–84, 1973.