International Journal of Mathematics and
Computer Science, **15**(2020), no. 4, 1321–1325

Parallelisable String-Based SP-Local Languages and their properties

N. Mohana, Kalyani Desikan

School of Advanced Sciences
Vellore Institute of Technology
Chennai, India

email: mohana.n@vit.ac.in, kalyanidesikan@vit.ac.in

(Received July 7, 2020, Accepted September 15, 2020)

Abstract

Scanners are a special class of automata and considered as computational models which accept local information. A local language is defined by the factors of its words of length two and can be generalized by Locally Testable Languages. A natural computation on sequential tasks is concatenation; it models a system finishing one task and then starting another one. We have adopted parallelism to model the tasks of a sequential system because this methodology enables computational tasks to be completed using various types of multiple resources simultaneously. Parallel computing has become an important concept in computer architecture. The classification of parallel computers is based on the level at which parallelism is supported by the hardware. We have introduced parallelisable string-based SP-languages as the collection of series parallel languages and parallel series languages. Parallelisable string-based SP-local language has been defined and their acceptance or recognition on parallelisable string-based SP-regular grammar has been proved.

1 Introduction

McNaughton and Papert [1] have introduced Locally Testable Languages (LTL). A standard regular event has been introduced and stated that any

Key words and phrases: Parallelisable string-based SP-language, Parallelisable string-based SP-local language, Parallelisable stringbased SP-regular grammar.
AMS (MOS) Subject Classifications:68Q45.
ISSN 1814-0432, 2020, http://ijmcs.future-in-tech.net

regular event is a homomorphic image of a standard regular event in [2] and also the theorem statement has been generalized to a tree automata and standard regular events have been considered as strictly locally testable event. Then the term 'event' has been replaced as 'language'. The study of syntactic semigroups of locally testable languages has been discussed and characterized in [3]. The characterisation of locally testable events and k-testable events have been derived in terms of semi-groups, structural decomposition, finite automata and restricted star-free expressions in [4]. The algorithms identify a regular event is locally testable or $k+1$-testable. Also, the characterisation of generalized definite events has been studied.

Regular languages that are Star-free languages can be found irrespective of the star operator. Languages namely, finite, definite and LTL have their own properties in terms of syntactic semigroups. The syntactic monoids of family of subsets of Σ^* with idempotent and commutative conditions have been derived in [5] and generalised to locally testable languages. In this article, Section 2 depicts the preliminary notations and definitions. Section 3 renders the definition of parallelisable string-based SP-languages, SP-regular grammar and SP-local languages. Also, provides the recognition of parallelisable string-based SP-local languges on parallelisable string-based SP-regular grammar.

2 Preliminaries

Consider a finite alphabet Σ. The collection of all sequential terms over Σ is Σ^* and Σ^+ is the collection of all strings except null string over Σ. Σ^k represents the set of strings of length k and Σ_k denotes the collection of all strings of depth k over the alphabet Σ.

Now let us consider some more possible terms or words which are obtained by using both sequential and parallel operators. $\Sigma^\oplus$ is the collection of all parallel strings, Σ_p^* the set of all parallel series strings and $\Sigma_s^\oplus$ the set of all series parallel strings over an alphabet. Prefix, suffix and infix of parallelisable string w of length or depth k are denoted as $P_k(w), S_k(w)$ and $F_k(w)$.

For example, consider $u = abab$ and $v = a\|b\|b\|b$, $P_1(u) = a, P_2(u) = ab, P_3(u) = aba$, $P_1(v) = a, P_2(v) = a\|b, P_3(v) = a\|b\|b$, $S_1(u) = b, S_2(u) = ab, S_3(u) = bab$, $S_1(v) = b, S_2(v) = b\|b, S_3(v) = b\|b\|b$, $F_2(u) = \{ab, ba\}, F_3(u) = \{aba, bab\}$, $F_2(v) = \{a\|b, b\|b\}, F_3(v) = \{b\|b\|b, a\|b\|b\}$.

Consider $x = (ba)\|(bb)$ and $y = (b\|a)(b\|b)$ whose length is 2 and depth is 2.

3 Parallelisable string-based SP-local languages

Definition 3.1. *Consider* $SP(\Sigma) = \Sigma_p^* \cup \Sigma_s^{\oplus}$. $SP(\Sigma)$ *denotes the collection of all parallelisable string-based SP-strings over* Σ. $L \subseteq SP(\Sigma)$ *is said to be a parallelisable string-based SP-language.*

EXAMPLE 3.2. *Consider* $L_1 = \{(aab)^{m\oplus} : m > 0\}$, $L_2 = \{(b\|a)^m : m > 0\}$, *that is,* $L_1 = \{aab, aab\|aab, ...\}$ *is a series parallel language,* $L_2 = \{(b\|a), (b\|a)(b\|a), ...\}$ *is a parallel series language then* $L = L_1 \cup L_2$ *is a parallelisable string-based SP-language.*

EXAMPLE 3.3. $L = \{a, a\|(bb), a\|(b(a\|bb)), ...\}$ *is not a parallelisable string-based SP-language, since the strings can neither be written as series parallel strings nor parallel series strings. But it is a SP-language with respect to posets defined by Lodaya [7].*

Definition 3.4. *A parallelisable string-based SP-grammar is given by the 4-tuple* $G_{sp} = (V, T, P, S)$ *with* T *denotes a finite collection of terminals,* V *is a finite collection of variables,* $S \in V$ *is a start variable and* P *is a finite collection of production rules of the form* $\beta \to \alpha$ *where* $\beta \in V$ *and* $\alpha \in SP(T \cup V)$.

EXAMPLE 3.5. *Let* G_{sp} *be a parallelisable string-based SP-grammar with* $T = \{a, b\}$, $V = \{S, V_1, V_2\}$, S *is a start variable and* P *is the collection of productions* $S \to bV_1$, $V_1 \to a\|V_2$, $V_2 \to bV_1$, $V_1 \to a$. *The sequences* $S \Rightarrow bV_1 \Rightarrow ba$,
$S \Rightarrow bV_1 \Rightarrow ba\|V_2 \Rightarrow ba\|bV_1 \Rightarrow ba\|ba$
$S \Rightarrow bV_1 \Rightarrow ba\|V_2 \Rightarrow ba\|bV_1 \Rightarrow ba\|ba\|V_2 \Rightarrow ba\|ba\|bV_1 \Rightarrow ba\|ba\|ba$ *are some derivations of* G_{sp}.
$L(G_{sp}) = \{(ba)^{m\oplus} : m > 0\}$ *is the language which is generated by* G_{sp}.

Definition 3.6. *A grammar* $G_{sp} = (V, T, P, S)$ *is a left-linear grammar if all its production rule is* $V_i \to V_j\|x|V_jx$ *or* $V_i \to x$ *with* $V_i, V_j \in V$ *and* $x \in T$

Definition 3.7. *A grammar* $G_{sp} = (V, T, P, S)$ *is called parallelisable string-based SP-regular if it is either right-linear or left-linear.*

Definition 3.8. $L \subseteq \Sigma^*$ *is called a local language if there exists a triple* (I, C, J) *where* $I, J \subseteq \Sigma$ *and* $C \subseteq \Sigma^2$ *such that*

$$L = \{u \in \Sigma^* : P_1(u) \in I, F_2(u) \subseteq C, S_1(u) \in J\}$$

EXAMPLE 3.9. *Consider the language* $L = \{ab^n : n > 0\}$. L *is local where* $I = \{a\}$, $J = \{b\}$, $C = \{ab, bb\}$.

Definition 3.10. *$L \subseteq \Sigma_s^{\oplus}$ is called $s-$local if there exists a triple (I_s, C_s, J_s) where $I_s \subseteq \Sigma$, $J_s \subseteq \Sigma$, $C_s \subseteq \Sigma^2$ such that $L = \{u \in \Sigma_s^{\oplus} : P_1(u) \in I_s, S_1(u) = J_s, F_2(u) \subseteq C_s\}$*

Example 3.11. *Consider $L = \{(ab)^{n\oplus} : n > 0\}$ where $I_s = \{a\}$, $J_s = \{b\}$ and $C_s = \{ab\}$.*

Definition 3.12. *$L \subseteq \Sigma_p^{*}$ is called $p-$local if there exists a triple (I_p, C_p, J_p) where $I_p \subseteq \Sigma$, $J_p \subseteq \Sigma$, $C_p \subseteq \Sigma_2$ such that $L = \{u \in \Sigma_p^{*} : P_1(u) \in I_p, S_1(u) = J_p, F_2(u) \subseteq C_p\}$*

Example 3.13. *Consider $L = \{(a\|b)^m : m > 0\}$ is a local parallel series language where $I_p = \{a\}$, $J_p = \{b\}$ and $C_p = \{a\|b\}$.*

Definition 3.14. *A language $L \subseteq SP(\Sigma)$ is called $sp - local$ if there exists a triple (I_{sp}, C_{sp}, J_{sp}) where $I_{sp} \subseteq \Sigma$, $J_{sp} \subseteq \Sigma$, $C_{sp} \subseteq \Sigma^2 \cup \Sigma_2$ such that $L = \{u \in SP(\Sigma) : P_1(u) \in I_{sp}, S_1(u) \in J_{sp}, F_2(u) \subseteq C_{sp}\}$*

Example 3.15. *Consider $L = \{(ab)^{m\oplus}, (a\|b)^m : m > 0\}$ where $I_{sp} = \{a\}$, $J_{sp} = \{b\}$ and $C_{sp} = \{ab, a\|b\}$.*

Theorem 3.16. *If L be a parallelisable string-based SP-local language then there exists a parallelisable string-based SP-regular grammar G_{sp} such that $L = L(G_{sp})$.*

Proof. Let L be a parallelisable string-based SP-local language. Then there exists a triple (I_{sp}, J_{sp}, C_{sp}) where $C_{sp} \subseteq \Sigma^2 \cup \Sigma_2$ and $I_{sp}, J_{sp} \subset \Sigma$. Consider a parallelisable string-based SP-regular grammar $G_{sp} = (V, T, P, S)$ with $T = \Sigma$, $V = \{S, V_1, V_2\}$, $S \in V$ is a start variable and P is the collection of productions $P : \{S \rightarrow x\|V_1|xV_1, x \in I_{sp} \ \cup V_1 \rightarrow yV_2, V_2 \rightarrow x\|V_1$ if $x\|y \in C_{sp} \ \cup V_1 \rightarrow y\|V_2, V_2 \rightarrow xV_1$ if $x.y \in C_{sp} \ \cup V_1 \rightarrow y, y \in J_{sp}\}$ where $x, y \in T$. We prove that $L = L(G_{sp})$. Let $u \in L \subseteq SP(\Sigma)$. If $u \in \Sigma_s^{\oplus}$ then it can be generated by the productions $S \rightarrow xV_1$, $V_1 \rightarrow y\|V_2$, $V_2 \rightarrow xV_1$, $V_1 \rightarrow y$. If $u \in \Sigma_p^{*}$ then it can be generated by the productions $S \rightarrow x\|V_1$, $V_1 \rightarrow yV_2$,$V_2 \rightarrow x\|V_1$, $V_1 \rightarrow y$. This implies $L \subseteq L(G_{sp})$. Assume $w \in L(G_{sp})$ then its production rules are as follows: $S \rightarrow xV_2|x\|V_2$, $V_1 \rightarrow xV_2|x\|V_2$, $V_2 \rightarrow y$ From the above construction of grammar, we have $L = \{w \in L(G_{sp}) : P_1(w) \in I_{sp}, S_1(w) \in J_{sp}$ and $F_2(w) \subset C_{sp}\}$. This implies $L(G_{sp}) \subseteq L$. Hence $L = L(G_{sp})$. □

Theorem 3.17. *If G_{sp} is a parallelisable string-based SP-grammar then $L(G_{sp})$ need not be a parallelisable string-based SP-local language.*

Proof. Consider a parallelisable string-based SP-regular grammar $G_{sp} = (V, T, P, S)$ with $T = \{a, b\}$, $V = \{S, A, B, C\}$, $S \in V$ is a start variable and P contains productions $S \to b\|A$, $A \to b\|B$, $B \to a\|C$, $C \to b.S$, $C \to b$. The sequences $S \Rightarrow b\|A \Rightarrow b\|b\|B \Rightarrow b\|b\|a\|C \Rightarrow b\|b\|a\|b$, $S \Rightarrow b\|A \Rightarrow b\|b\|B \Rightarrow b\|b\|a\|C \Rightarrow b\|b\|a\|b.S \Rightarrow b\|b\|a\|b.b\|A \Rightarrow b\|b\|a\|b.b\|b\|B \Rightarrow b\|b\|a\|b.b\|b\|a\|C \Rightarrow b\|b\|a\|b.b\|b\|a\|b$ are some derivations of G_{sp}. $L(G_{sp}) = \{(b\|b\|a\|b)^m : m > 0\}$ where $I_{sp} = \{b\}$, $J_{sp} = \{b\}$ and $C_{sp} = \{b\|b, b\|a, a\|b\}$ is not local, since the triple (I_{sp}, J_{sp}, C_{sp}) is same for another parallelisable string-based SP-language $L = \{(b\|a\|b\|b)^m : m > 0\}$. Therefore, a language generated by a parallelisable string-based SP-regular grammar need not be local.

□

References

[1] R. McNaughton, S. A. Papert, , Counter-Free Automata (MIT research monograph No. 65), *MIT Press*, 1971.

[2] J. W. Thatcher, Generalized sequential machine maps, *Journal of Computer and System Sciences*, **4,** no. 4, (1970), 339–367.

[3] Y. Zalcstein, Locally testable languages, *Journal of Computer and System Sciences*, **6,** no. 2, (1972), 151–167.

[4] J. A. Brzozowski, I. Simon, Characterizations of locally testable events, *Discrete Mathematics*, 4 (3), 243-271, 1973.

[5] J. A. Brzozowski, F. E. Fich, On generalized locally testable languages, *Discrete Mathematics*, **50,** (1984), 153–169.

[6] R. McNaughton, Algebraic decision procedures for local testability, *Mathematical systems theory*, **8,** no. 1, 60–76.

[7] K. Lodaya, P. Weil, Series-Parallel languages and the bounded-width property, *Theoretical Computer Science*, **237,** nos. 1-2, (2000), 347–380.