Theoretical Computer Science 518 (2014) 80–95

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Regular and context-free pattern languages over small alphabets ☆

Daniel Reidenbach [a], Markus L. Schmid [b,*]

[a] Department of Computer Science, Loughborough University, Loughborough, Leicestershire, LE11 3TU, United Kingdom
[b] Fachbereich 4 – Abteilung Informatik, Universität Trier, D-54296 Trier, Germany

ARTICLE INFO

Article history:
Received 13 November 2012
Accepted 31 July 2013
Communicated by J. Karhumaki

Keywords:
Pattern languages
Regular languages
Context-free languages

ABSTRACT

Pattern languages are generalisations of the copy language, which is a standard textbook example of a context-sensitive and non-context-free language. In this work, we investigate a counter-intuitive phenomenon: with respect to alphabets of size 2 and 3, pattern languages can be regular or context-free in an unexpected way. For this regularity and context-freeness of pattern languages, we give several sufficient and necessary conditions and improve known results.

1. Introduction

Within the scope of this paper, a *pattern* is a finite sequence of terminal symbols and variables, taken from two disjoint alphabets Σ and X. We say that such a pattern α generates a word w if w can be obtained from α by substituting arbitrary words of terminal symbols for all variables in α, where, for any variable, the substitution word must be identical for all of its occurrences in α. More formally, a substitution is therefore a *terminal-preserving* morphism, i.e., a morphism $\sigma : (\Sigma \cup X)^* \to \Sigma^*$ that satisfies $\sigma(a) = a$ for every $a \in \Sigma$. The *pattern language* $L(\alpha)$ is then simply the set of all words that can be obtained from α by arbitrary substitutions. For example, the language generated by $\alpha_1 := x_1 x_1 \mathtt{aba} x_2$ (where $\Sigma := \{\mathtt{a}, \mathtt{b}\}$ and $X \supset \{x_1, x_2\}$) is the set of all words over $\{\mathtt{a}, \mathtt{b}\}$ that have any square as a prefix, an arbitrary suffix and the factor $\mathtt{aba}$ in between. Hence, e.g., $w_1 := \mathtt{abbabbabaaa}$ and $w_2 := \mathtt{bbaba}$ are included in $L(\alpha_1)$, whereas $w_3 := \mathtt{abbababb}$ and $w_4 := \mathtt{bbbabaaa}$ are not.

Pattern languages were introduced by Angluin [1] in 1980 in order to formalise the process of computing commonalities of words in some given set. Her original definition disallows the substitution of the empty word for the variables, and therefore these languages are also referred to as *nonerasing* pattern languages (or *NE*-pattern languages for short). This notion of pattern languages was soon afterwards extended by Shinohara [20], who included the empty word as an admissible substitution word, leading to the definition of *extended* or *erasing* pattern languages (or *E*-pattern languages for short). Thus, in the above example, w_2 is contained in the E-pattern language, but not in the NE-pattern language of α_1. As revealed by numerous studies, the small difference between the definitions of NE- and E-pattern languages entails substantial differences between some of the properties of the resulting (classes of) formal languages (see, e.g., Mateescu and Salomaa [14] for a survey).

Pattern languages have not only been intensively studied within the scope of inductive inference (see, e.g., Lange and Wiehagen [12], Rossmanith and Zeugmann [19], Reidenbach [17] and, for a survey, Ng and Shinohara [15]), but their

☆ A preliminary version [18] of this paper was presented at the conference DLT 2012.

* Corresponding author.
E-mail addresses: D.Reidenbach@lboro.ac.uk (D. Reidenbach), MSchmid@uni-trier.de (M.L. Schmid).

0304-3975/$ – see front matter
http://dx.doi.org/10.1016/j.tcs.2013.07.035

properties are closely connected to a variety of fundamental problems in computer science and discrete mathematics, such as for (un-)avoidable patterns (cf. Jiang et al. [10]), word equations (cf. Mateescu and Salomaa [13]), the ambiguity of morphisms (cf. Freydenberger et al. [7]), equality sets (cf. Harju and Karhumäki [8]) and extended regular expressions (cf. Câmpeanu et al. [4]). Therefore, quite a number of basic questions for pattern languages are still open or have been resolved just recently (see, e.g., Freydenberger and Reidenbach [6], Bremer and Freydenberger [3]).

If a pattern contains each of its variables once, then this pattern can be interpreted as a regular expression, and therefore its language is regular. In contrast to this, if a pattern has at least one variable with multiple occurrences, then its languages is a variant of the well-known *copy language* $\{xx \mid x \in \Sigma^*\}$, which for $|\Sigma| \geqslant 2$ is a standard textbook example of a context-sensitive and non-context-free language. Nevertheless, there are some well-known example patterns of the latter type that generate regular languages. For instance, the NE-pattern language of $\alpha_2 := x_1x_2x_2x_3$ is regular for $|\Sigma| = 2$, since squares are unavoidable for binary alphabets, which means that the language is co-finite. Surprisingly, for terminal alphabets of size 2 and 3, there are even certain E- and NE-pattern languages that are context-free but not regular. This recent insight is due to Jain et al. [9] and solves a longstanding open problem.

It is the purpose of our paper to further investigate this counter-intuitive existence of languages that appear to be variants of the copy language, but are nevertheless regular or context-free. Thus, we wish to establish criteria where the seemingly high complexity of a pattern does not translate into a high complexity of its language. Since, as demonstrated by Jain et al., this phenomenon does not occur for E-pattern languages if the pattern does not contain any terminal symbols or if the size of the terminal alphabet is at least 4, our investigations focus on patterns with terminal symbols and on small alphabets of size 2 or 3.

2. Definitions and known results

Let $\mathbb{N} := \{1, 2, 3, \ldots\}$ and let $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. For an arbitrary alphabet A, a *string* (*over* A) is a finite sequence of symbols from A, and ε stands for the *empty string*. The notation A^+ denotes the set of all non-empty strings over A, and $A^* := A^+ \cup \{\varepsilon\}$. For the *concatenation* of two strings w_1, w_2 we write $w_1 \cdot w_2$ or simply w_1w_2. We say that a string $v \in A^*$ is a *factor* of a string $w \in A^*$ if there are $u_1, u_2 \in A^*$ such that $w = u_1 \cdot v \cdot u_2$. If u_1 or u_2 is the empty string, then v is a *prefix* (or a *suffix*, respectively) of w. The notation $|K|$ stands for the size of a set K or the length of a string K.

If we wish to refer to the symbol at a certain position j, $1 \leqslant j \leqslant n$, in a string $w = \mathtt{a}_1 \cdot \mathtt{a}_2 \cdot \ldots \cdot \mathtt{a}_n$, $\mathtt{a}_i \in A$, $1 \leqslant i \leqslant n$, then we use $w[j] := \mathtt{a}_j$ and if the length of a string is unknown, then we denote its last symbol by $w[-] := w[|w|]$. Furthermore, for each j, j', $1 \leqslant j < j' \leqslant |w|$, let $w[j, j'] := \mathtt{a}_j \cdot \mathtt{a}_{j+1} \cdot \ldots \cdot \mathtt{a}_{j'}$ and $w[j, -] := w[j, |w|]$.

For any alphabets A, B, a *morphism* is a function $h : A^* \to B^*$ that satisfies $h(vw) = h(v)h(w)$ for all $v, w \in A^*$; h is said to be *nonerasing* if and only if, for every $a \in A$, $h(a) \neq \varepsilon$. Let Σ be a finite alphabet of so-called *terminal symbols* and X a countably infinite set of *variables* with $\Sigma \cap X = \emptyset$. We normally assume $X := \{x_1, x_2, x_3, \ldots\}$. A *pattern* is a non-empty string over $\Sigma \cup X$, a *terminal-free pattern* is a non-empty string over X and a *word* is a string over Σ. For any pattern α, we refer to the set of variables in α as $\mathrm{var}(\alpha)$ and for any $x \in \mathrm{var}(\alpha)$, $|\alpha|_x$ denotes the number of occurrences of x in α. A morphism $h : (\Sigma \cup X)^* \to \Sigma^*$ is called a *substitution* if $h(a) = a$ for every $a \in \Sigma$.

Definition 1. Let $\alpha \in (\Sigma \cup X)^*$ be a pattern. The *E-pattern language* of α is defined by $L_{\mathrm{E},\Sigma}(\alpha) := \{h(\alpha) \mid h : (\Sigma \cup X)^* \to \Sigma^* \text{ is a substitution}\}$. The *NE-pattern language* of α is defined by $L_{\mathrm{NE},\Sigma}(\alpha) := \{h(\alpha) \mid h : (\Sigma \cup X)^* \to \Sigma^* \text{ is a nonerasing substitution}\}$.

We denote the class of *regular* languages, *context-free* languages, E-*pattern* languages over Σ and NE-*pattern* languages over Σ by REG, CF, E-PAT$_\Sigma$ and NE-PAT$_\Sigma$, respectively. We use regular expressions as they are commonly defined (see, e.g., Yu [22]) and for any regular expression r, $L(r)$ denotes the language described by r.

We recapitulate regular and block-regular patterns as defined by Shinohara [21] and Jain et al. [9]. A pattern α is a *regular* pattern if, for every $x \in \mathrm{var}(\alpha)$, $|\alpha|_x = 1$. Every factor of variables of α that is delimited by terminal symbols is called a variable block. More precisely, for every i, j, $1 \leqslant i \leqslant j \leqslant |\alpha|$, $\alpha[i, j]$ is a *variable block* if and only if $\alpha[k] \in X$, $i \leqslant k \leqslant j$, $\alpha[i-1] \in \Sigma$ or $i = 1$ and $\alpha[j+1] \in \Sigma$ or $j = |\alpha|$. A pattern α is *block-regular* if in every variable block of α there occurs at least one variable x with $|\alpha|_x = 1$. Let $\mathrm{Z} \in \{\mathrm{E}, \mathrm{NE}\}$. The class of Z-pattern languages defined by regular patterns and block-regular patterns are denoted by Z-PAT$_{\Sigma,\mathrm{reg}}$ and Z-PAT$_{\Sigma,\mathrm{b\text{-}reg}}$, respectively. To avoid any confusion, we explicitly mention that the term regular pattern always refers to a pattern with the syntactical property of being a regular pattern and a regular E- or NE-pattern language is a pattern language that is regular, but that is not necessarily given by a regular pattern.

In order to prove some of the technical claims in this paper, the following two versions of the pumping lemma for regular languages as stated by Yu [22] shall be used.

Pumping Lemma 1. *Let $L \subseteq \Sigma^*$ be a regular language. Then there is a constant n, depending on L, such that for every $w \in L$ with $|w| \geqslant n$ there exist $x, y, z \in \Sigma^*$ such that $w = xyz$ and*

1. $|xy| \leqslant n$,