Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Patterns with bounded treewidth [☆]

Daniel Reidenbach, Markus L. Schmid [*]

Department of Computer Science, Loughborough University, Loughborough, Leicestershire, LE11 3TU, United Kingdom

ARTICLE INFO

Article history:
Received 24 July 2013
Available online 20 August 2014

Keywords:
Pattern languages
Membership problem
Treewidth
Extended regular expressions
Parameterised pattern matching

ABSTRACT

A pattern is a string consisting of variables and terminal symbols, and its language is the set of all words that can be obtained by substituting arbitrary words for the variables. The membership problem for pattern languages, i.e., deciding on whether or not a given word is in the pattern language of a given pattern is NP-complete. We show that any parameter of patterns that is an upper bound for the treewidth of appropriate encodings of patterns as relational structures, if restricted, allows the membership problem for pattern languages to be solved in polynomial time. Furthermore, we identify new such parameters.

1. Introduction

A *pattern* α is a finite string that consists of *variables* and *terminal symbols* (taken from a fixed alphabet Σ), and its language is the set of all words that can be derived from α when substituting arbitrary words over Σ for the variables. For example, the language L generated by the pattern $\alpha := x_1 \mathtt{a} x_2 \mathtt{b} x_1$ (where x_1, x_2 are variables and $\mathtt{a}, \mathtt{b}$ are terminal symbols) consists of all words with an arbitrary prefix u, followed by the letter $\mathtt{a}$, an arbitrary word v, the letter $\mathtt{b}$ and a suffix that equals the prefix u. Thus, $w_1 := \mathtt{aaabbaa}$ is contained in L, whereas $w_2 := \mathtt{baaba}$ is not.

Patterns provide a compact and natural way to describe formal languages. In their original definition given by Angluin [3] variables can only be substituted by non-empty words; hence, the term *nonerasing pattern languages* (or, for short, *NE-pattern languages*) is used. *Extended* or *erasing pattern languages* (or, for short, *E-pattern languages*) where variables can also be substituted by the empty word have been introduced by Shinohara [23]. The original motivation of pattern languages (cf. Angluin [3]) is derived from *inductive inference*, i.e., the task of inferring a pattern from any given sequence of all words in its pattern language, for which numerous results can be found in the literature (see, e.g., Angluin [3], Shinohara [23], Lange and Wiehagen [13], Rossmanith and Zeugmann [22], Reidenbach [16,17] and, for a survey, Ng and Shinohara [15]). On the other hand, due to their simple definition, pattern languages have connections to many areas of theoretical computer science, e.g., formal language theory, learning theory, combinatorics on words and pattern matching, and their general properties have been investigated in various contexts (for a survey, see, e.g., Mateescu and A. Salomaa [14]).

With respect to practical applications, their membership problem, i.e., the problem of deciding, for a given word w and a pattern α, whether or not the variables of α can be substituted in such a way that w is obtained, is probably the most important aspect of pattern languages. This is mainly due to the connection to so-called *extended regular expressions with backreferences* (*REGEX* for short) (see, e.g., Câmpeanu et al. [7]). REGEX can roughly be considered as classical regular expressions that are equipped with the possibility to define backreferences, i.e., to require factors to be repeated at several defined positions in the word; hence, backreferences correspond to the variables in patterns. While backreferences

[☆] A preliminary version of this work was presented at LATA 2012.
[*] Corresponding author.
E-mail addresses: D.Reidenbach@lboro.ac.uk (D. Reidenbach), MSchmid@uni-trier.de (M.L. Schmid).

http://dx.doi.org/10.1016/j.ic.2014.08.010
0890-5401/© 2014 Elsevier Inc. All rights reserved.

dramatically increase the expressive power of classical regular expressions, they are also responsible for the membership problem of this language class to become NP-complete. This is particularly worth mentioning as today's text editors and programming languages (such as Perl, Python, Java, etc.) all provide so-called *REGEX engines* that compute the solution to the membership problem for any language given by a REGEX and an arbitrary string. Hence, despite its theoretical intractability, algorithms that perform the match test for REGEX are a practical reality. While pattern languages merely describe a proper subset of REGEX languages, they cover what is computationally hard, i.e., the concept of backreferences. Hence, investigating the membership problem for pattern languages helps to improve algorithms solving the match test for extended regular expressions with backreferences.

The membership problem for pattern languages, that was first shown to be NP-complete by Angluin [3] in 1980, can also be considered as some kind of pattern matching task, since we have to decide whether or not a given word satisfies a given pattern. In fact, this pattern matching aspect of pattern languages, independently from Angluin's work, has recently been rediscovered in the pattern matching community in terms of so-called *parameterised pattern matching*, where a text is not searched for all occurrences of a specific factor, but for all occurrences of factors that satisfy a given pattern with parameters (i.e., variables). In the original version of parameterised pattern matching introduced by Baker [4], variables in the pattern can only be substituted by single symbols and, furthermore, the substitution must be injective, i.e., different variables cannot be substituted by the same symbol. Amir et al. [2] then generalised this problem by dropping the injectivity condition and Amir and Nor [1] added the possibility of substituting variables by words instead of single symbols and they also allowed "don't care" symbols to be used in addition to variables. In 2009, Clifford et al. [8] considered parameterised pattern matching as introduced by Amir and Nor, but without "don't care" symbols, which led to patterns as introduced by Angluin. In [1], motivations for the membership problem of pattern languages can be found from such diverse areas as software engineering, image searching, DNA analysis, poetry and music analysis, or author validation.

Our main research task is to identify parameters of patterns that, if restricted to a constant, allow a polynomial time membership problem. The benefit of finding such parameters is twofold. Firstly, we can learn what properties of a pattern are actually responsible for the complexity of the membership problem, i.e., we achieve a refined complexity analysis of this problem. Secondly, restricting these parameters is likely to lead to improved algorithms for the membership problem of pattern languages. The first such parameter that comes to mind is the number of different variables in a pattern. Its restriction constitutes a trivial way to obtain a polynomial time membership problem, since the brute force algorithm that simply enumerates all possibilities to substitute the variables by terminal words in order to check whether the input word can be obtained is exponential in the number of variables (for a detailed complexity analysis see Ibarra et al. [12]). Nevertheless, the number of variables is a central parameter of patterns and important results about the learnability of pattern languages (see Angluin [3] and Reischuk and Zeugmann [21]) as well as recent results about the inclusion problem of pattern languages (see Bremer and Freydenberger [6]) are concerned with patterns with a restricted number of variables. Moreover, Stephan et al. [25] investigate the parameterised complexity of the membership problem for pattern languages and they show that it is fixed parameter intractable, if parameterised by the number of variables. The membership problem for pattern languages given by patterns with only one occurrence per variable (introduced by Shinohara [24]) is solvable in polynomial time, simply because these patterns describe regular languages; hence, they are called *regular* patterns. If the patterns are unrestricted, then the membership problem can still be solved in polynomial time provided that the length of the input word is bounded (see Geilke and Zilles [11]).

The arguably first nontrivial restriction of patterns that allow a polynomial time membership problem are Shinohara's *non-cross* patterns [24], i.e., patterns where between any two occurrences of the same variable x no other variable different from x occurs. However, this result does not provide a structural parameter of patterns that can be considered to contribute to the complexity of the membership problem. Recently, in [19], an automata based approach has been used in order to extend Shinohara's result to an infinite hierarchy of classes of pattern languages with a polynomial time membership problem. The idea in [19] is to restrict a rather subtle parameter, namely the *distance* several occurrences of any variable x may have in a pattern (i.e., the maximum number of different variables separating any two consecutive occurrences of x). This parameter is called the *variable distance* vd of a pattern α, and in [19] it is demonstrated that the membership problem is solvable in time $O(|\alpha|^3 \times |w|^{(\mathrm{vd}(\alpha)+4)})$, so it is exponential only in the variable distance.

In this work, we approach the problem of identifying such parameters in a novel and quite general way. More precisely, we encode patterns and words as relational structures and, thus, reduce the membership problem to the homomorphism problem for relational structures. Our main result is that for any parameter of patterns that is an upper bound for the treewidth of the corresponding relational structures, we obtain a polynomial time algorithm for the membership problem if the parameter is bounded by a constant. In this new framework, we can restate the known results about the complexity of the membership problem mentioned above, as well as identifying new and, compared to the old results, rather large classes of patterns with a polynomial time membership problem. Therefore, we provide a convenient way to study the membership problem for pattern languages, which, as will be pointed out by our results, has still potential for further improvements.

2. Preliminaries

Let $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$ denote the set of all natural numbers. 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 nonempty 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 $w_1 w_2$. We say that a