Patterns with Bounded Treewidth

Daniel Reidenbach and Markus L. Schmid *

Department of Computer Science, Loughborough University,
Loughborough, Leicestershire, LE11 3TU, UK
{D.Reidenbach,M.Schmid}@lboro.ac.uk

Abstract. 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 to a constant, allows the membership problem for pattern languages to be solved in polynomial time. Furthermore, we identify a new such parameter, called the scope coincidence degree.

Keywords: Pattern Languages, Membership Problem, Treewidth, Extended Regular Expressions

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 [1] 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 [18]. The original motivation for pattern languages (cf. Angluin [1]) 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 [1], Shinohara [18], Lange and Wiehagen [8], Rossmanith and Zeugmann [16], Reidenbach [11,12] and, for a survey, Ng and Shinohara [10]). On the other hand, due to their simple definition, pattern languages have connections to many other areas of theoretical computer science and their general properties have been investigated in various contexts (for a survey, see, e. g., Mateescu and A. Salomaa [9]). For example, there exist several versions of regular expressions

* Corresponding author.

that are extended in such a way that pattern languages can be defined (see, e. g., Bordihn et al. [3]). The problem to decide 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, i. e., the *membership problem* for pattern languages, has been shown to be NP-complete by Angluin [1].

Besides the theoretical importance of pattern languages, the concept of patterns also finds practical application in so-called *extended regular expressions with backreferences* (*REGEX* for short) (see, e. g., Câmpeanu et al. [5]). 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 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 matchtest 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 matchtest for extended regular expressions with backreferences.

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. Nevertheless, the number of variables is a central parameter of patterns and important results about the learnability of pattern languages (see Angluin [1] and Reischuk and Zeugmann [15]) as well as recent results about the inclusion problem of pattern languages (see Bremer and Freydenberger [4]) are concerned with patterns with a restricted number of variables. The membership problem for pattern languages given by patterns with only one occurrence per variable (introduced by Shinohara [18]) is also solvable in polynomial time, simply because these patterns describe regular languages; hence, they are called *regular* patterns.

The arguably first nontrivial restriction of patterns that allow a polynomial time membership problem are Shinohara's *non cross* patterns [19], 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 [14], Shinohara's result has been extended to an infinite hierarchy of classes of pattern languages with a polynomial time membership problem. The idea in [14] 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 [14] it is demonstrated that the membership problem is solvable in time $\mathrm{O}(|\alpha|^3 \times |w|^{(\mathrm{vd}(\alpha)+4)})$, 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 any parameter of patterns that is an upper bound for the treewidth of the corresponding relational structures, if restricted to a constant, allows the membership problem to be solved in polynomial time. In this new framework, we can restate the known results about the complexity of the membership problem mentioned above, as well as identifying a new parameter that is stronger than the variable distance. Therefore, we provide a convenient way to treat the membership problem for pattern languages, which, as shall be pointed out by our results, has still potential for further improvements.

Note that, due to space constraints, all proofs are omitted.

2 Preliminaries

Let $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$. 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 string $v \in A^*$ is a *factor* of a string $w \in A^*$ if there are $u_1, u_2 \in A^*$ such that $w = u_1 \cdot v \cdot u_2$. The notation $|K|$ stands for the size of a set K or the length of a string K. The term $\mathrm{alph}(w)$ denotes the set of all symbols occurring in w. If we wish to refer to the symbol at a certain position j, $1 \leq j \leq n$, in a string $w = \mathtt{a}_1 \cdot \mathtt{a}_2 \cdot \cdots \cdot \mathtt{a}_n$, $\mathtt{a}_i \in A$, $1 \leq i \leq n$, we use $w[j] := \mathtt{a}_j$. Furthermore, for each j, j', $1 \leq j < j' \leq |w|$, let $w[j, j'] := \mathtt{a}_j \cdot \mathtt{a}_{j+1} \cdot \cdots \cdot \mathtt{a}_{j'}$ and $w[j, -] := w[j, |w|]$. If $j > |w|$, we define $w[j, -] = \varepsilon$.

Pattern Languages and the Scope Coincidence Degree

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 nonempty string over $\Sigma \cup X$, a *terminal-free pattern* is a nonempty 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 $\mathrm{var}(\alpha)$.

A morphism $h : (\Sigma \cup X)^* \to \Sigma^*$ is called a *substitution* if $h(a) = a$ for every $a \in \Sigma$. We define the *E-pattern language* of a pattern α by $L_{\mathrm{E},\Sigma}(\alpha) := \{h(\alpha) \mid h : (\Sigma \cup X)^* \to \Sigma^* \text{ is a substitution}\}$. The *NE-pattern language* $L_{\mathrm{NE},\Sigma}(\alpha)$ of α is defined analogously, just with respect to nonerasing substitutions. Since in our work the impact of the choice of the alphabet Σ is negligible, we mostly denote pattern languages by $L_{\mathrm{E}}(\alpha)$ and $L_{\mathrm{NE}}(\alpha)$.

The problem to decide for a given pattern α and a given word $w \in \Sigma^*$ whether $w \in L_{\mathrm{E}}(\alpha)$ (or $w \in L_{\mathrm{NE}}(\alpha)$) is called the *membership problem for* E*-pattern languages* (or NE*-pattern languages*, respectively). For every class $C \subseteq (\Sigma \cup X)^*$ and every $Z \in \{\mathrm{E}, \mathrm{NE}\}$, Z-PATMem(C) denotes the membership problem for Z-pattern languages where the patterns are restricted to the class C.

The concept of the scope coincidence degree has already been introduced in [13]. However, here we shall define it in a slightly different (yet equivalent) way. Let α be a pattern. For every $y \in \mathrm{var}(\alpha)$, the *scope of* y *in* α is defined by $\mathrm{sc}_\alpha(y) := \{i, i+1, \ldots, j\}$, where i is the leftmost and j the rightmost occurrence of y in α. The scopes of $y_1, y_2, \ldots, y_k \in \mathrm{var}(\alpha)$ *coincide in* α if and only if $\bigcap_{1 \leq i \leq k} \mathrm{sc}_\alpha(y_i) \neq \emptyset$. Finally, the *scope coincidence degree* of α ($\mathrm{scd}(\alpha)$) is the maximum number of variables in α such that their scopes coincide. Let $\Sigma := \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ and let the pattern $\beta \in (\Sigma \cup X)^*$ be given by $\beta := x_1\mathtt{b}x_2\mathtt{a}x_1x_3x_2\mathtt{ab}x_3x_4x_2x_4x_5\mathtt{bcb}x_1x_4x_5$. It is easy to see that each set $\{x_1, x_2, x_3\}$, $\{x_1, x_2, x_4\}$ and $\{x_1, x_4, x_5\}$ contain variables the scopes of which coincide, but there does not exist a set of more than 3 variables with the same property. Hence, $\mathrm{scd}(\beta) = 3$. It is straightforward to see that the scope coincidence degree can be computed in time linear in the length of the pattern.

Relational Structures, Treewidth and Homomorphism Problem

For the sake of completeness, we repeat the following standard definitions very briefly, and for a comprehensive reference, the reader is referred to Chapters 4, 11 and 13 of Flum and Grohe [6].

A *(relational) vocabulary* τ is a finite set of relation symbols. Every relation symbol $R \in \tau$ has an *arity* $\mathrm{arity}(R) \geq 1$. A *τ-structure* (or simply *structure*), comprises a finite set A called the *universe* and, for every $R \in \tau$, an *interpretation* $R^{\mathcal{A}} \subseteq A^{\mathrm{arity}(R)}$. For example, every graph can be given as a relational structure over a vocabulary with one binary relation symbol representing the edges. Let $\mathcal{A}$ and $\mathcal{B}$ be structures of the same vocabulary τ with universes A and B, respectively. A *homomorphism* from $\mathcal{A}$ to $\mathcal{B}$ is a mapping $h : A \to B$ such that for all $R \in \tau$, of arity r, and for all $a = (a_1, a_2, \ldots, a_r) \in A^r$, $a \in R^{\mathcal{A}}$ implies $h(a) \in R^{\mathcal{B}}$, where $h(a) = (h(a_1), h(a_2), \ldots, h_r(a_r))$.

Next, we introduce the concept of a *treewidth* of a graph $\mathcal{G}$, denoted by $\mathrm{tw}(\mathcal{G})$. We omit the standard definition of the treewidth, that makes use of the concept of tree decompositions of graphs (see, e. g., Bodlaender [2]). Instead, we apply an alternative characterisation in terms of a game due to Seymour and Thomas [17].

In the *robber-cops-game* (called *jump searching* in [17]), a number of cops try to catch a robber on a graph. Let $\mathcal{G} := (V, E)$ be a graph. A *position (of the robber-cops-game)* is a pair (C, R), where $C \subseteq V$ and R is a connected subgraph of $\mathcal{G}$ that does not contain any vertex of C. The set C contains the vertices currently occupied by cops. The set R, on the other hand, is the position of the robber. Since the robber can move with infinite speed we can interpret it to occupy all vertices of the connected subgraph R at the same time. The initial position of the game is (C_0, R_0), where $C_0 = \emptyset$ and R_0 is some connected subgraph chosen by the robber. At the start of the i^{th} step of the game we have position (C_{i-1}, R_{i-1}). Now all the cops are removed from the graph and then again placed on some vertices $C_i \subseteq V$. After that, the robber chooses (if possible) a new connected subgraph R_i that does not contain any vertex from C_i and *touches* R_{i-1}, i. e., R_{i-1} and R_i have a common vertex or an edge connects a vertex of R_{i-1} with a vertex of R_i. If in any step i of the robber cops game, the cops could choose a set of vertices C_i such that there does not exists a position (C_i, R_i), i. e., for every possible connected subgraph R_i that touches R_{i-1}, $R_i \subseteq C_i$ is satisfied, then the cops win the robber-cops-game. We say that $k \in \mathbb{N}$ *cops can catch a robber on* $\mathcal{G}$ if the robber-cops-game can be won by the cops such that, for every position (C_i, R_i) in the game, $|C_i| \leq k$.

For example, one cop is not enough to catch a robber even on a single path, since the robber can always outrun the cop as soon as it changes to another vertex. Two cops can catch a robber on a path and also on arbitrary trees. To catch a robber on a simple circle three cops are required: one is permanently placed on some vertex, which turns the cycle in a path, and then the other two can corner the robber in one of the two dead ends. The next theorem establishes the relation between the robber-cops-game and the treewidth of a graph.

Theorem 1 (Seymour and Thomas [17]). *Let $\mathcal{G}$ be an arbitrary graph. Then $k \in \mathbb{N}$ cops can catch a robber on $\mathcal{G}$ if and only if* $\operatorname{tw}(\mathcal{G}) \leq k - 1$.

In order to define the treewidth of relational structures, we need the concept of the *Gaifman graph* of a τ-structure $\mathcal{A}$, which is the graph that has the universe A of $\mathcal{A}$ as vertices and two vertices are connected if and only if they occur together in some relation (see Chapter 11 of Flum and Grohe [6]). Now we can state the definition of the treewidth that shall be used for our results:

Definition 2. *The treewidth of a structure equals $k-1$, where k is the minimum number of cops that are sufficient to catch a robber on its Gaifman graph.*

The *homomorphism problem HOM* is the problem to decide, for given structures $\mathcal{A}$ and $\mathcal{B}$, whether there exists a homomorphism from $\mathcal{A}$ to $\mathcal{B}$. For any set of structures C, by $\text{HOM}(C)$ we denote the homomorphism problem, where the left hand structure is restricted to be from C. If C is a class of structures with bounded treewidth, then $\text{HOM}(C)$ can be solved in polynomial time. This is a classical result that has been first achieved in terms of *constraint satisfaction problems* by Freuder [7] (see also Chapter 13 of Flum and Grohe [6]).

Theorem 3 (Freuder [7]). *Let C be a set of structures with bounded treewidth. HOM(C) is solvable in polynomial time.*

3 Patterns and Words as Relational Structures

In this section, we introduce a way to represent patterns and terminal words as relational structures. Our overall goal is to reduce the membership problem for pattern languages to the homomorphism problem for relational structures.

Representing words as relational structures is a common technique when mathematical logic is applied to language theory (see, e. g., Thomas [20] for a survey). However, our representations of patterns and words by structures differ from the standard technique, since our approach is tailored to the homomorphism problem of structures and, furthermore, we want to exploit the treewidth.

In order to encode patterns and terminal words, i. e., an instance of the membership problem for pattern languages, we use the relational vocabulary $\tau_\Sigma := \{E, S, L, R\} \cup \{D_a \mid a \in \Sigma\}$, where E, S are binary relations and L, R, D_a, $a \in \Sigma$, are unary relations. The vocabulary depends on Σ, the alphabet under consideration. In order to represent a pattern α by a τ_Σ-structure, we interpret the set of positions of α as the universe. The roles of S, L, R and D_a, $a \in \Sigma$, are straightforward: S relates adjacent positions, L and R are singletons that contain the leftmost and rightmost position, respectively, and, for every $a \in \Sigma$, the relation D_a contains the positions in α where the terminal symbol a occurrs. For the encoding of the variables, we do not explicitly store their positions in the pattern, which seems impossible, since the number of different variables can be arbitrarily large. Instead, we use the relation E in order to record pairs of positions where the same variable occurs and, furthermore, this is done in a "sparse" way. More precisely, the relation E relates *some* positions with the same variable, i. e., positions i, j with $\alpha[i] = \alpha[j]$, in such a way that the symmetric transitive closure of E contains *all* pairs (i, j) with $\alpha[i] = \alpha[j]$ and $\alpha[i] \in X$. This way of interpreting relation E is crucial for our results.

We now state the formal definition and shall illustrate it afterwards.

Definition 4. *Let α be a pattern and let $\mathcal{A}_\alpha$ be a τ_Σ-structure. $\mathcal{A}_\alpha$ is an* α-structure *if it has universe $P_\alpha := \{1, 2, \ldots, |\alpha|\}$ and $S^{\mathcal{A}_\alpha} := \{(i, i+1) \mid 1 \leq i \leq |\alpha| - 1\}$, $L^{\mathcal{A}_\alpha} := \{1\}$, $R^{\mathcal{A}_\alpha} := \{|\alpha|\}$, for every $a \in \Sigma$, $D_a^{\mathcal{A}_\alpha} := \{i \mid \alpha[i] = a\}$, and $E^{\mathcal{A}_\alpha}$ is such that, for all $i, j \in P_\alpha$,*

- *$(i, j) \in E^{\mathcal{A}_\alpha}$ implies $\alpha[i] = \alpha[j]$ and $i \neq j$,*
- *$\alpha[i] = \alpha[j]$ implies that (i, j) is in the symmetric transitive closure of $E^{\mathcal{A}_\alpha}$.*

Since τ_Σ contains only unary and binary relation symbols, it is straightforward to derive the Gaifman graph from an α-structure, which is simply a graph with two different kinds of edges due to $S^{\mathcal{A}_\alpha}$ and $E^{\mathcal{A}_\alpha}$. Hence, we shall switch between these two models at our convenience without explicitly mentioning it. In the previous definition, the universe as well as the interpretations for the relation symbols S, L, R and D_a, $a \in \Sigma$, are uniquely defined for a fixed pattern α, while there are several possibilities to define an interpretation of E. Intuitively, a valid interpretation of E is created by connecting different occurrences of the same variable by edges in such a way that all the occurrences of some variable describe a connected component. The simplest way to do this is to add an edge between

every two occurrences of the same variable, i. e., $E^{\mathcal{A}_\alpha} := \{(i,j) \mid \alpha[i] = \alpha[j]\}$. However, we shall see that for our results the interpretation of E is crucial and using the one just mentioned is not advisable. Another example of a valid interpretation of E is the following one. For every $x \in \mathrm{var}(\alpha)$, let l_x be the leftmost occurrence of x in α. Defining $E^{\mathcal{A}_\alpha} := \bigcup_{x \in \mathrm{var}(\alpha)} \{(l_x, i) \mid l_x < i \leq |\alpha|, \alpha[i] = x\}$ yields another possible α-structure.

Next, we define a canonical α-structure, i. e., the interpretation of E is such that every occurrence of a variable x at position i is connected to the next occurrence of x to the right of position i.

Definition 5. *Let α be a pattern. The* standard α-structure $(\mathcal{A}^s_\alpha)$ *is the α-structure where $E^{\mathcal{A}^s_\alpha} := \{(i,j) \mid 1 \leq i < j \leq |\alpha|, \exists\, x \in X$ such that $x = \alpha[i] = \alpha[j]$ and $\alpha[k] \neq x, i < k < j\}$.*

As an example, we consider the standard α-structure $\mathcal{A}^s_\alpha$ for the pattern $\alpha := x_1{\cdot}\mathtt{a}{\cdot}\mathtt{b}{\cdot}x_1{\cdot}\mathtt{b}{\cdot}x_2{\cdot}\mathtt{a}{\cdot}x_1{\cdot}x_2{\cdot}x_1$. The universe of $\mathcal{A}^s_\alpha$ is $P_\alpha = \{1, 2, \ldots, 10\}$ and the relations are interpreted in the following way. $S^{\mathcal{A}^s_\alpha} = \{(1,2), (2,3), \ldots, (9,10)\}$, $L^{\mathcal{A}^s_\alpha} = \{1\}$, $R^{\mathcal{A}^s_\alpha} = \{10\}$, $D^{\mathcal{A}^s_\alpha}_{\mathtt{a}} = \{2, 7\}$, $D^{\mathcal{A}^s_\alpha}_{\mathtt{b}} = \{3, 5\}$ and, finally, $E^{\mathcal{A}^s_\alpha} = \{(1,4), (4,8), (6,9), (8,10)\}$.

We continue with representing words over the terminal alphabet Σ as τ_Σ-structures. We recall that it is our goal to represent the membership problem for pattern languages as homomorphism problem for relational structures. Hence, the way we represent terminal words by τ_Σ-structures must cater for this purpose. Furthermore, we have to distinguish between the E case and the NE case. We first introduce the NE case and shall afterwards point out how to extend the constructions for the E case. We choose the universe to be the set of all possible factors of w, where these factors are represented by their unique start and end positions in w; thus, two factors that are equal but occur at different positions in w are different elements of the universe. The interpretation of L contains all prefixes and the interpretation of R contains all suffixes of w. The interpretation of S, which for patterns contains pairs of adjacent variables, contains now pairs of adjacent (non-overlapping) factors of w. The relation E is interpreted such that it contains *all* pairs of factors that are equal and non-overlapping. Finally, for every $a \in \Sigma$, D_a contains all factors of length one that equal a. This is necessary for the possible terminal symbols in the pattern.

For the E case, the empty factors of w need to be represented as well. To this end, for every i, $0 \leq i \leq |w|$, we add an element i_ε to the universe denoting the empty factor between positions i and $i+1$ in w. The interpretations of S and R are extended to contain the empty prefix and the empty suffix, respectively, and relation S is extended to relate non-empty factors to adjacent empty factors and, in addition, each empty factor is also related to itself. Next, we formally define this construction for the NE case and its extension to the E case.

Definition 6. *Let $w \in \Sigma^*$ be a terminal word. The* standard-NE-w-structure $(\mathrm{NE}-\mathcal{A}^{\mathrm{s}}_w)$ *with universe P_w is defined by*

- $P_w := \{(i,j) \mid 1 \leq i \leq j \leq |w|\}$,

- $E^{\mathrm{NE}-\mathcal{A}^s_w} := \{((i,j),(i',j')) \mid j < i' \text{ or } j' < i, w[i,j] = w[i',j']\}$,
- $S^{\mathrm{NE}-\mathcal{A}^s_w} := \{((i,j),(j+1,j')) \mid 1 \leq i \leq j, j+1 \leq j' \leq |w|\}$,
- $L^{\mathrm{NE}-\mathcal{A}^s_w} := \{(1,j) \mid 1 \leq j \leq |w|\}$,
- $R^{\mathrm{NE}-\mathcal{A}^s_w} := \{(i,|w|) | \mid 1 \leq i \leq |w|\}$ *and,*
- *for every* $a \in \Sigma$, $D_a^{\mathrm{NE}-\mathcal{A}^s_w} := \{(i,i) \mid w[i] = a\}$.

Let $\mathrm{NE}-\mathcal{A}^s_w$ *be the standard-NE-*w*-structure with universe* P_w*. We define the* standard-E-w-structure $(\mathrm{E}-\mathcal{A}^{\mathrm{s}}_w)$ *with universe* P^{E}_w *as follows:*

- $P^{\mathrm{E}}_w := P_w \cup \{i_\varepsilon \mid 0 \leq i \leq |w|\}$,
- $E^{\mathrm{E}-\mathcal{A}^s_w} := E^{\mathrm{NE}-\mathcal{A}^s_w} \cup \{(i_\varepsilon, j_\varepsilon) \mid 0 \leq i,j \leq |w|, i \neq j\}$,
- $S^{\mathrm{E}-\mathcal{A}^s_w} := S^{\mathrm{NE}-\mathcal{A}^s_w} \cup \{(i_\varepsilon, i_\varepsilon) \mid 0 \leq i \leq |w|\} \cup$
  $\{((i,j),j_\varepsilon)) \mid 1 \leq i \leq j \leq |w|\} \cup \{(i_\varepsilon,(i+1,j)) \mid 0 \leq i \leq j \leq |w|\}$,
- $L^{\mathrm{E}-\mathcal{A}^s_w} := L^{\mathrm{NE}-\mathcal{A}^s_w} \cup \{0_\varepsilon\}$,
- $R^{\mathrm{E}-\mathcal{A}^s_w} := R^{\mathrm{NE}-\mathcal{A}^s_w} \cup \{|w|_\varepsilon\}$ *and,*
- *for every* $a \in \Sigma$, $D_a^{\mathrm{E}-\mathcal{A}^s_w} := D_a^{\mathrm{NE}-\mathcal{A}^s_w}$.

In the following lemma we state that the membership problem for pattern languages can be reduced to the homomorphism problem for relational structures. We shall informally explain this for the case of terminal-free NE-pattern languages. If there exists a substitution that maps the pattern α to the word w, then we can construct a homomorphism g from $\mathcal{A}_\alpha$ to $\mathrm{NE}-\mathcal{A}^{\mathrm{s}}_w$ by mapping the positions of α to the factors of w according to the substitution h. If two positions in α are adjacent, then so are their images under h in w and the same holds for equal variables in α; hence, g is a valid homomorphism. If, on the other hand, there exists a homomorphism g from $\mathcal{A}_\alpha$ to $\mathrm{NE}-\mathcal{A}^{\mathrm{s}}_w$, then the elements of the universe of $\mathcal{A}_\alpha$, i. e., positions of α, are mapped onto factors of w such that a factorisation of w is described. This is enforced by the relations S, L and R. Furthermore, this mapping from α to w induced by g is a substitution, since the symmetric transitive closure of $E^{\mathcal{A}_\alpha}$ contains all pairs (i,j) with $\alpha[i] = \alpha[j]$ and $\alpha[i] \in X$. For general patterns with terminal symbols and for the E case the idea is the same, but the situation is technically more complex.

Lemma 7. *Let* α *be a pattern,* $w \in \Sigma^*$ *and let* $\mathcal{A}_\alpha$ *be an* α*-structure. Then* $w \in L_{\mathrm{NE}}(\alpha)$ *(or* $w \in L_{\mathrm{E}}(\alpha)$*) if and only if there exists a homomorphism from* $\mathcal{A}_\alpha$ *to* $\mathrm{NE}-\mathcal{A}^s_w$ *(or from* $\mathcal{A}_\alpha$ *to* $\mathrm{E}-\mathcal{A}^s_w$*, respectively).*

From Lemma 7 and Theorem 3, we can conclude that, for patterns that can be encoded by α-structures with a bounded treewidth, the membership problem is solvable in polynomial time.

Theorem 8. *Let* $C \subseteq (X \cup \Sigma)^+$ *and let* g *be a mapping that, in polynomial time, maps every* $\alpha \in C$ *to an* α*-structure, such that* $\widehat{C} := \{g(\alpha) \mid \alpha \in C\}$ *has bounded treewidth. Then* NE-PATMem(C) *and* E-PATMem(C) *are decidable in polynomial time.*

Due to Theorem 8, the task of identifying parameters of patterns that, if bounded, allow a polynomial time membership problem, can now be seen from a different angle, i. e., as the problem of finding classes of patterns that can be encoded by α-structures with a bounded treewidth. The fact that we can easily rephrase known results about the complexity of the membership problem for pattern languages in terms of standard α-structures with a bounded treewidth, pointed out by the following proposition, indicates that this point of view is natural and fits with our current knowledge of the membership problem.

Proposition 9. *Let α be a pattern. If α is non-cross or regular then* $\mathrm{tw}(\mathcal{A}^s_\alpha) \leq 2$. *Furthermore,* $\mathrm{tw}(\mathcal{A}^s_\alpha) \leq |\operatorname{var}(\alpha)|$.

While Proposition 9 is trivially true, an analogous result can also be given for the variable distance of patterns (see Section 1 and [14]), which is a more subtle parameter the restriction of which is known to yield a polynomial time membership problem. This follows from the main result of the subsequent section, which shows that a stronger parameter than the variable distance, namely the already mentioned scope coincidence degree, also allows a polynomial time membership problem if restricted to a constant. The question arises why we do not simply consider the treewidth of a pattern α, i. e., $\mathrm{tw}(\alpha) := \min\{\mathrm{tw}(\mathcal{A}_\alpha) \mid \mathcal{A}_\alpha \text{ is an } \alpha\text{-structure}\}$, to be an appropriate parameter of patterns that should be bounded in order to solve the membership problem efficiently. The problem here is that, firstly, for a pattern α there exists an exponential number of α-structures and, secondly, computing the treewidth of graphs is an NP-complete problem. Consequently, it is not clear whether this parameter can be computed in polynomial time and, in order to conclude complexity theoretical results from Theorem 8, we rely on finding easily computable parameters of patterns that – ideally as tight as possible – are upper bounds for $\mathrm{tw}(\alpha)$.

4 Patterns with Restricted Scope Coincidence Degree

In this section we show that, for every pattern α, the treewidth of the standard α-structure is bounded by the scope coincidence degree of α. To this end, we shall play the robber-cops-game defined in Section 2 on the Gaifman graph of standard α-structures. For the sake of convenience, we shall not distinguish anymore between a pattern α and the Gaifman graph of its standard α-structure, i. e., we change at will between interpreting the positions in the pattern as occurrences of variables or as vertices in the Gaifman graph of the standard α-structure. Similarly, we allow some leeway with respect to the robber-cops-game and shall play it directly on a pattern α, meaning to actually playing it on the Gaifman graph of its standard α-structure. Next, we define a strategy to search a pattern in terms of the robber-cops-game.

Definition 10. *Let α be a pattern. We define the* inspector search strategy (on α). *We assume that we have an infinite number of cops, where one distinct cop is called the* inspector *and all other cops are called* constables. *Let m, $1 \leq m \leq |\alpha|$,*

be the leftmost occurrence of a variable in α. Initially, the inspector is placed on vertex m. For every i, $m \leq i \leq |\alpha| - 1$, when the inspector is located on vertex i, the following steps are executed:

1. *If $\alpha[i]$ is the leftmost occurrence of some variable x in α, then a constable is placed on vertex i. If $\alpha[i]$ is an occurrence of some variable x, but not the leftmost one, then the constable from vertex j is moved to vertex i, where $j < i$, is the rightmost occurrence of x that is currently occupied by a constable.*
2. *The inspector moves from vertex i to vertex $i + 1$.*
3. *If i is the rightmost occurrence of some variable x in α, then the constable on vertex i is removed.*

The number of constables that are required to carry out the inspector search strategy on α is called the constable number of α.

Informally, the inspector search strategy on some pattern α can be described in the following way. The inspector moves through the pattern from left to right. Every occurrence of a terminal symbol is ignored and the inspector just moves on. If it enters an occurrence of a variable, it places a constable there. A new constable is required if this is the first occurrence of some variable x. If, on the other hand, there exists an earlier occurrence of x in α, then, by definition of the search strategy, a constable is located at the next occurrence of x to the left of the current inspector position. This constable is now moved forward to the position of the inspector. If the inspector reaches a rightmost occurrence of a variable, then also a constable is moved to this position, but removed immediately after the inspector moves on. However, it is important that the constable is placed there before the inspector moves on and remains there while the inspector moves to the next position. Obviously, this procedure terminates as soon as the inspector reaches position $|\alpha|$. We note that as long as there exists at least one variable in the pattern, the constable number is at least one, hence, we can assume that the constable number is always at least one.

We observe the following property of the inspector search strategy.

Proposition 11. *Let α be a pattern. Every time step 1 of the inspector search strategy on α is executed, the following condition is satisfied. Let $p_1, p_2, \ldots, p_k \in \{1, 2, \ldots, |\alpha|\}$ with $p_1 < p_2 < \ldots < p_k$ be the positions occupied by constables. Then there are k different variables $y_1, y_2, \ldots, y_k$, such that $\mathrm{var}(\alpha[p_1, p_k]) = \{y_1, y_2, \ldots, y_k\}$ and, for every i, $1 \leq i \leq k$, p_i is the rightmost occurrence of y_i in $\alpha[p_1, p_k]$.*

The next lemma describes how a certain area of the pattern can be sealed off by the constables, such that the robber cannot reach this area.

Lemma 12. *Let α be a pattern and let $p_1, p_2, \ldots, p_k \in \{1, 2, \ldots, |\alpha|\}$ with $p_1 < p_2 < \ldots < p_k$ such that $\mathrm{var}(\alpha[p_1, p_k]) = \{y_1, y_2, \ldots, y_k\}$ and, for every i, $1 \leq i \leq k$, p_i is the rightmost occurrence of y_i in $\alpha[p_1, p_k]$. If vertices $p_1, \ldots, p_k$ are occupied by constables, then the robber cannot reach a vertex t, $p_1 \leq t \leq p_k$, from a vertex s, $p_k < s$.*

The previous results can be used in order to show that, for every pattern α, a robber can be caught by applying the inspector search strategy on α.

Lemma 13. *Let α be a pattern with constable number k. If $k = 1$, then 3 cops are sufficient to catch a robber on the Gaifman graph of $\mathcal{A}^s_\alpha$, and if $k \geq 2$, then $k+1$ cops are sufficient to catch a robber on the Gaifman graph of $\mathcal{A}^s_\alpha$.*

It is worth mentioning that the special case in the above lemma concerning patterns with a constable number of 1 is caused by the fact that the patterns may contain terminals. For terminal-free patterns α with a constable number of 1, a robber can be caught on the Gaifman graph of $\mathcal{A}^s_\alpha$ by 2 cops. Next, we show that the constable number of a pattern equals its scope coincidence degree.

Lemma 14. *For every pattern α, the constable number of α equals* $\mathrm{scd}(\alpha)$.

By the previous lemmas, we can conclude that, for every pattern α with $\mathrm{scd}(\alpha) = 1$, a robber can be caught on α by using one inspector and 2 constables, and for every pattern α with $\mathrm{scd}(\alpha) = k$, $k \geq 2$, a robber can be caught on α by using one inspector and k constables. Hence, referring to Theorem 1, the treewidth of the standard α-structure is bounded by the scope coincidence degree. Furthermore, the standard α-structures of the class of patterns with restricted variable distance have a bounded treewidth as well.

Theorem 15. *Let α be a pattern. Then* $\mathrm{tw}(\mathcal{A}^s_\alpha) \leq \mathrm{scd}(\alpha) \leq \mathrm{vd}(\alpha) + 1$.

Theorems 8 and 15 imply the following complexity result.

Corollary 16. *Let $k \in \mathbb{N}$ and $Z \in \{\mathrm{E}, \mathrm{NE}\}$. The problem Z-*PATMem*$(\{\alpha \mid \mathrm{scd}(\alpha) \leq k\})$ is solvable in polynomial time.*

We conclude this work by some questions not explicitly addressed so far. Since in Definition 4 we leave the exact definition of the relation symbol E open, there are many possible α-structures for a pattern α that all permit an application of Theorem 8. However, the standard way of encoding patterns (Definition 5) has turned out to be sufficient for all results in the present paper. Hence, it would be interesting to know whether or not, for some pattern α, there exists a better α-structure than the standard one, i. e., $\mathrm{tw}(\alpha) < \mathrm{tw}(\mathcal{A}^s_\alpha)$. This question is open, but we conjecture that it can be answered in the negative.

Another question is whether the scope coincidence degree of a pattern α is a tight upper bound for the treewidth of the standard α-structure. This question can be answered in the negative. Consider for example the pattern $\alpha := x_1 \cdot x_2 \cdot \cdots \cdot x_{k-1} \cdot x_k \cdot x_k \cdot x_{k-1} \cdot \cdots \cdot x_2 \cdot x_1$. It is easy to see that $\mathrm{scd}(\alpha) = k$. On the other hand, we can catch a robber on α with 3 cops. Thus $\mathrm{tw}(\mathcal{A}^s_\alpha) \leq 2 < \mathrm{scd}(\alpha)$. This examples also gives an indication that it might be possible to identify a parameter closer to $\mathrm{tw}(\alpha)$ still preserving polynomial time computability.

Acknowledgements

The authors wish to thank Sanjay Jain and the anonymous referees for their detailed and helpful remarks and suggestions, which will be taken into account for the full version of this paper.

References

1. Angluin, D.: Finding patterns common to a set of strings. Journal of Computer and System Sciences 21, 46–62 (1980)
2. Bodlaender, H.L.: Treewidth: Characterizations, applications, and computations. In: Graph-Theoretic Concepts in Computer Science. Lecture Notes in Computer Science, vol. 4271, pp. 1–14 (2006)
3. Bordihn, H., Dassow, J., Holzer, M.: Extending regular expressions with homomorphic replacement. RAIRO Theoretical Informatics and Applications 44, 229–255 (2010)
4. Bremer, J., Freydenberger, D.D.: Inclusion problems for patterns with a bounded number of variables. In: Proceedings of the 14th International Conference on Developments in Language Theory, DLT 2010. pp. 100–111 (2010)
5. Câmpeanu, C., Salomaa, K., Yu, S.: A formal study of practical regular expressions. International Journal of Foundations of Computer Science 14, 1007–1018 (2003)
6. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2006)
7. Freuder, E.C.: Complexity of k-tree structured constraint satisfaction problems. In: Proceedings of the 8th National Conference on Artificial Intelligence. pp. 4–9 (1990)
8. Lange, S., Wiehagen, R.: Polynomial-time inference of arbitrary pattern languages. New Generation Computing 8, 361–370 (1991)
9. Mateescu, A., Salomaa, A.: Patterns. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 1, pp. 230–242. Springer (1997)
10. Ng, Y., Shinohara, T.: Developments from enquiries into the learnability of the pattern languages from positive data. Theoretical Computer Science 397, 150–165 (2008)
11. Reidenbach, D.: A non-learnable class of E-pattern languages. Theoretical Computer Science 350, 91–102 (2006)
12. Reidenbach, D.: Discontinuities in pattern inference. Theoretical Computer Science 397, 166–193 (2008)
13. Reidenbach, D., Schmid, M.L.: Finding shuffle words that represent optimal scheduling of shared memory access. In: Proc. 5th International Conference on Language and Automata Theory and Applications, LATA 2011. Lecture Notes in Computer Science, vol. 6638, pp. 465–476 (2011)
14. Reidenbach, D., Schmid, M.L.: A polynomial time match test for large classes of extended regular expressions. In: Proc. 15th International Conference on Implementation and Application of Automata, CIAA 2010. Lecture Notes in Computer Science, vol. 6482, pp. 241–250 (2011)
15. Reischuk, R., Zeugmann, T.: An average-case optimal one-variable pattern language learner. Journal of Computer and System Sciences 60, 302–335 (2000)
16. Rossmanith, P., Zeugmann, T.: Stochastic finite learning of the pattern languages. Machine Learning 44, 67–91 (2001)
17. Seymour, P.D., Thomas, R.: Graph searching and a min-max theorem for tree-width. Journal of Combinatorial Theory, Series B 58, 22–33 (1993)
18. Shinohara, T.: Polynomial time inference of extended regular pattern languages. In: Proc. RIMS Symposia, Kyoto. LNCS, vol. 147, pp. 115–127 (1982)
19. Shinohara, T.: Polynomial time inference of pattern languages and its application. In: Proc. 7th IBM MFCS. pp. 191–209 (1982)
20. Thomas, W.: Languages, automata, and logic. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 3, chap. 7, pp. 389–455. Springer (1997)