Learning Relational Patterns*

Michael Geilke[1] and Sandra Zilles[2]

[1] Fachbereich Informatik, Technische Universität Kaiserslautern
D-67653 Kaiserslautern, Germany
geilke.michael@gmail.com
[2] Department of Computer Science, University of Regina
Regina, SK, Canada S4S 0A2
zilles@cs.uregina.ca

Abstract. Patterns provide a simple, yet powerful means of describing formal languages. However, for many applications, neither patterns nor their generalized versions of typed patterns are expressive enough. This paper extends the model of (typed) patterns by allowing relations between the variables in a pattern. The resulting formal languages are called *Relational Pattern Languages* (RPLs). We study the problem of learning RPLs from positive data (text) as well as the membership problem for RPLs. These problems are not solvable or not efficiently solvable in general, but we prove positive results for interesting subproblems.
We further introduce a new model of learning from a restricted pool of potential texts. Probabilistic assumptions on the process that generates words from patterns make the appearance of some words in the text more likely than that of other words. We prove that, in our new model, a large subclass of RPLs can be learned with high confidence, by effectively restricting the set of likely candidate patterns to a finite set after processing a single positive example.

1 Introduction

After Angluin [1] introduced the pattern languages, they became a popular object of study in algorithmic learning theory. Patterns are strings consisting of constant and variable symbols; substituting variables by strings of constant symbols generates words in the corresponding pattern languages. Thus patterns provide a simple and intuitive, yet powerful means of describing formal languages.

One focus has been on learning pattern languages in the limit from positive data: a learner is given access to a stream of all and only the words in the target language L and is supposed to generate a sequence of patterns that eventually stabilizes on a pattern generating L [3, 2, 9]. Two central concerns are here

(a) the apparent trade-off between the expressiveness of pattern languages and the existence of algorithms for learning such languages from positive data [10],

(b) the lack of efficient algorithms for fundamental tasks involved in many intuitive learning procedures, like solving the membership problem for pattern

* This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

languages [1]. Any learning algorithm that uses membership tests to construct patterns consistent with the known data will suffer from the NP-hardness of the membership problem.

The first concern is best illustrated when comparing non-erasing pattern languages [1] to erasing pattern languages [13], the latter ones differing from the former ones only in the detail that they allow to replace variables in a pattern with the empty string. This little additional detail makes patterns more expressive, but at the same time, in general, non-learnable in the limit from positive data [10]. Furthermore, even erasing pattern languages are often not expressive enough to model interesting real-world applications. To this end, Wright [16] and Koshiba [6] introduced an extension of pattern languages, called typed pattern languages. Typed patterns restrict the set of allowed substitutions separately for each variable, so as to model languages in which, *e.g.*, the variable x_3 in the pattern "`author:` x_1 `title:` x_2 `year:` x_3" should be replaced only by 4-digit strings, whereas x_1 and x_2 can be replaced by strings containing letters. Unfortunately, little is known about general conditions under which typed pattern languages are learnable. Moreover, for many applications, neither pattern languages nor typed pattern languages are sufficient to model the complex structure in textual data. Below we give examples of bioinformatics applications in which it is obvious that (typed) pattern languages lack the ability to express that the substitutions for two or more distinct variables are dependent on each other.

This paper extends the model of (typed) pattern languages by allowing that certain variables in a pattern are in a particular relation with each other. The resulting formal languages are called *Relational Pattern Languages*; both classical pattern languages and typed pattern languages are special cases of relational pattern languages. Moreover, relational pattern languages overcome the limitations observed in terms of expressiveness of (typed) pattern languages.

We study relational pattern languages both with respect to their learnability in the limit from positive data (text) and with respect to the complexity of the membership problem. Our contributions along these lines are as follows:

(1) Considering Gold's model of learning in the limit from arbitrary texts [3], relational pattern languages can be learned as long as the set of allowed relations between variables is finite and no variable can be replaced by the empty string. The conditions are essential for learnability.

(2) The membership problem for relational pattern languages is NP-complete in general, but we show a number of interesting sub-problems that can be solved efficiently. Most notably, we prove that the membership problem for relational patterns over finitely many polynomial-time decidable relations is solvable in polynomial time if the words for which to test membership are bounded in length. This is not only a very interesting sub-problem from an application point of view, but also not trivial, since we deal with potential empty substitutions.[3]

Considering practical applications, Gold's model of learning in the limit can be criticized: often there is no step in the learning process after which the set

[3] If only non-empty substitutions for variables are allowed, the membership problem restricted to a finite set of words becomes trivial.

of candidate patterns can be reduced to a finite set, thus forcing the learner to make a best guess within an infinite version space—this might make it difficult for a user to decide when to interrupt the infinite learning process. Despite allowing the learning algorithm this limit behaviour, there is no general positive learnability result for the case that empty substitutions are allowed, *cf.* [10]. This is partly due to the fact that a learning algorithm in Gold's model is required to be successful on any text for the target language. As a step towards a practical model for learning a large class of (erasing) relational pattern languages, we make the following contributions:

(3) We introduce a new model of learning from a restricted pool of potential texts, since in practice not all texts (data streams) are equally likely. We assume that there are probability distributions over the strings that can be substituted for pattern variables, thus making the appearance of some words in the text more likely than that of other words. As we will explain below, our model differs from previous approaches that were based on a similar motivation [5, 12]. We refer to the underlying patterns as *Probabilistic Relational Patterns*.

(4) We prove that in our new model, a large class of (erasing) probabilistic relational pattern languages can be learned with high confidence, by effectively restricting the set of likely candidate patterns to a finite set after processing only a single positive example.

(5) Our model results in a simple but potentially practical method for testing membership correctly with high confidence, for all probabilistic relational patterns using a finite set of recursive relations.

2 Learning (typed) pattern languages

Languages are defined with respect to a non-empty *alphabet* Σ. A *word* w is a finite, possibly empty, sequence of symbols from Σ the length of which is denoted by $|w|$.[4] ϵ refers to the empty word, *i.e.*, the word of length 0. The set of all words over Σ is denoted by Σ^*, and the set of all non-empty words over Σ by Σ^+; hence $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. A *language* L is a subset of Σ^*. By $w_1 \circ w_2$ we denote the concatenation of two words w_1 and w_2 (where, for ease of presentation, we allow w_1 and/or w_2 to be written as $\sigma \in \Sigma$ rather than a word (σ) of length 1). In what follows, we always assume Σ to be a finite set of cardinality at least 2. We denote the set of all non-zero natural numbers by $\mathbb{N}_+$.

In Gold's model of learning in the limit from positive data [3], a class of languages is learnable if there is a learner that "identifies" every language in the class from any of its texts, where a text for a language L is an infinite sequence $\tau(0), \tau(1), \tau(2), \ldots$ of words such that $\{\tau(i) \mid i \in \mathbb{N}\} = L$.

Definition 1 (Gold [3]) *Let $\mathcal{L}$ be a class of languages. $\mathcal{L}$ is* learnable in the limit from positive data *if there is a hypothesis space $\{L_i \mid i \in \mathbb{N}\} \supseteq \mathcal{L}$ and a partial recursive mapping S such that, for any $L \in \mathcal{L}$ and any text $(\tau(i))_{i\in\mathbb{N}}$ for*

[4] "Word" refers to a finite sequence of symbols exclusively from Σ, whereas "string" refers to any other sequence of symbols.

L, $S(\tau(0), \ldots, \tau(n))$ is defined for all $n \in \mathbb{N}$ and there is a $j \in \mathbb{N}$ with $L_j = L$ and $S(\tau(0), \ldots, \tau(n)) = j$ for all but finitely many n.

A class of languages that has been studied in the formal language theory community as well as in the learning theory community is Angluin's class of non-erasing pattern languages [1], defined as follows. Let $X = \{x_1, x_2, \ldots\}$ be a countable set of symbols called variables; we require that X be disjoint from Σ. A *pattern* is a non-empty finite string over $\Sigma \cup X$. The set of all patterns over $\Sigma \cup X$ will be denoted by Pat_Σ. A *substitution* is a string homomorphism $\theta : Pat_\Sigma \to \Sigma^*$ that is the identity when restricted to Σ. The set of all substitutions with respect to Σ is denoted by Θ_Σ. The non-erasing language $L_{NE}(p)$ of a pattern $p \in (\Sigma \cup X)^+$ is defined by $L_{NE}(p) = \{w \in \Sigma^* \mid \exists \theta \in \Theta_\Sigma \, [\theta(p) = w \wedge \forall x \in X \, [\theta(x) \neq \epsilon]]\}$, *i.e.*, it consists of all words that result from substituting all variables in p by non-empty words over Σ^*. The set $\mathcal{L}_{\Sigma,NE}$ of non-erasing pattern languages is given by $\mathcal{L}_{\Sigma,NE} = \{L_{NE}(p) \mid p \in Pat_\Sigma\}$. Angluin showed that $\mathcal{L}_{\Sigma,NE}$ is learnable in the limit from positive data [1].

Shinohara extended pattern languages by permitting the substitution of variables by ϵ [13]. We denote the erasing pattern language of a pattern p by $L_E(p)$, where $L_E(p) = \{w \in \Sigma^* \mid \exists \theta \in \Theta_\Sigma \, [\theta(p) = w]\}$, and refer to the class of erasing pattern languages by $\mathcal{L}_{\Sigma,E}$. For $|\Sigma| \in \{2, 3, 4\}$, $\mathcal{L}_{\Sigma,E}$ is not learnable in the limit from positive data [10]. Wright [16] studied a subclass of erasing pattern languages under restricted substitutions, leading to Koshiba's typed pattern languages [6]. In Koshiba's model, each variable x in a pattern p is assigned a particular *type* $T(x) = t$, where $t \subseteq \Sigma^*$ is recursive. (p, T) is then called a typed pattern. The words generated by (p, T) are formed by substituting any variable of type t in p only with words from t, resulting in the typed pattern language $L(p, T)$.[5]

Types make pattern languages more expressive and more suitable for applications. For example, a system for entering bibliographic data as described by Shinohara [13] might use patterns like $p = \texttt{author:}\ x_1\ \texttt{title:}\ x_2\ \texttt{year:}\ x_3$. One would expect x_3 to be substituted only by certain two or four digit integers—a property that becomes expressible when using types.

In fact, every recursive language L can trivially be written as a typed pattern language, generated by the pattern $p = x_1$ where the type of x_1 is L. Thus, a typed pattern is not always a useful description of a language, from an application point of view. Ideally, one would like to keep the types themselves simple, to make the pattern understandable by humans.

Consider for example patterns describing RNA sequences formed out of bases A, C, G, U. The secondary structure of molecules contains information about bonds between base pairs in the sequence; C

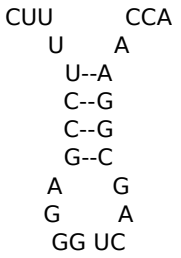

Fig. 1. RNA sequence with bonds.

[5] Actually, Koshiba did not allow substituting variables by the empty string, but we relax this condition here. We also deviate slightly from his notation.

can bond with G, A with U. For example, Fig. 1 shows potential intramolecular bonds for the sequence *CUUU UCCG AGGG UCAG CGGA ACCA*. Obviously, the bonding partners towards one end of the sequence must appear in reverse order of the corresponding bases towards the opposite end of the sequence. To express this with a typed pattern would require the whole subsequence *UCCG AGGG UCAG CGGA* to be the substitution for a single variable and thus an element of a complex type.

A formally simpler example is the language $L_1 = \{a^n b^n \mid n \geq 1\}$, which is context-free but not regular. To express L_1 as a typed pattern language requires "complex" types; regular types are not sufficient.

Proposition 2 *Let (p, T) be a typed pattern. If $L(p, T) = L_1$ then there is some $x \in X$ occurring in p such that $T(x)$ is not a regular language.*

Proof. Let (p, T) be a typed pattern generating L_1, and k be the number of variables that occur more than once in p. Assume $T(x)$ was regular for all $x \in X$ occurring in p. We deduce a contradiction by induction on k.

For $k = 1$, $L(p, T)$ would be the concatenation of regular languages and as such regular—a contradiction.

For $k \geq 2$, let x be a variable that occurs l times in p with $l \geq 2$. W.l.o.g., we may assume that $L(x, T) \neq \{\epsilon\}$. Since $L(p, T) = L_1$ and x occurs multiple times, one obtains $L(x, T) \subseteq \{a^i \mid i \in \mathbb{N}_+\}$ or $L(x, T) \subseteq \{b^i \mid i \in \mathbb{N}_+\}$. For $L(x, T) \subseteq \{a^i \mid i \in \mathbb{N}_+\}$, let y be a variable not occurring in p, $T'(y) = \{a^{i \cdot l} \mid a^i \in L(x, T)\}$, and $T'(z) = T(z)$ for all variables z occurring p. Then $L(p', T') = L(p, T) = L_1$, where p' is the pattern that results when removing all occurrences of x from p and adding y as a prefix. The inductive hypothesis yields the required contradiction. The case $L(x, T) \subseteq \{b^i \mid i \in \mathbb{N}_+\}$ is analogous. □

3 Relational patterns

In order to model interdependencies between the substitutions of variables, we introduce relations between variables into the definition of patterns.

Definition 3 *Let R be a set of relations over Σ^*. Then, for any $n \in \mathbb{N}_+$, R_n denotes the set of n-ary relations in R. A* relational pattern *with respect to Σ and R is a pair (p, v_R) where p is a pattern over Σ and $v_R \subseteq \{(r, y_1, \ldots, y_n) \mid n \in \mathbb{N}_+,\ r \in R_n,\ \text{and } y_1, \ldots, y_n \text{ are variables in } p\}$. The set of relational patterns with respect to R will be denoted by $Pat_{\Sigma,R}$.*

The set of all possible substitutions for (p, v_R) is denoted by $\Theta_{(p,v_R),\Sigma}$. It contains all substitutions $\theta \in \Theta_\Sigma$ that fulfill, for all $n \in \mathbb{N}_+$:

$$\forall r \in R_n\ \forall y_1, \ldots, y_n \in X\ \left[(r, y_1, \ldots, y_n) \in v_R \Rightarrow (\theta(y_1), \ldots, \theta(y_n)) \in r\right].$$

The language of (p, v_R), denoted by $L(p, v_R)$, is defined as $\{w \in \Sigma^* \mid \exists \theta \in \Theta_{(p,v_R),\Sigma} : \theta(p) = w\}$. The set of all languages of relational patterns with respect to R will be denoted by $\mathcal{L}_{\Sigma,R}$.

For instance, $r = \{(w_1, w_2) \mid w_1, w_2 \in \Sigma^* \wedge |w_1| = |w_2|\}$ is a binary relation, which, applied to two variables x_1 and x_2 in a relational pattern (p, v_R), ensures that the substitutions of x_1 and x_2 generating words from p always have the same length. Formally, this is done by including (r, x_1, x_2) in v_R.

We assume, without loss of generality, that for every variable x occurring in a relational pattern (p, v_R), there is exactly one $r \in R_1$ such that $(r, x) \in v_R$. In fact, this unary relation r represents the type of variable x. If there is no $r \in R_1$ with $(r, x) \in v_R$, we can include (r_*, x) in (p, v_R), where $w \in r_* \leftrightarrow w \in \Sigma^*$. If R_1 contains several r_i (for i in some index set I) with $(r_i, x) \in v_R$, we can replace them by the single relation $(r_{\cap I}, x)$ where $w \in r_{\cap I} \leftrightarrow \forall i \in I\ [w \in r_i]$. We will use the terms "type" and "unary relation" interchangeably. Similarly, without loss of generality, each set of n variables is included in v_R with at most one n-ary relation. We further assume that relational patterns do not contain any variable twice. This is no restriction, since repetition of variables can be expressed by an equality relation between two distinct variables.

We are only going to consider the case that R is finite, which seems sufficient for many practical applications. It is easy to see that $\mathcal{L}_{\Sigma,NE}$ and $\mathcal{L}_{\Sigma,E}$, as well as the class of typed pattern languages over finitely many types, are subclasses of $\mathcal{L}_{\Sigma,R}$, for respective suitably defined finite sets R.

The gain in expressiveness shows for example in L_1, which, by Proposition 2, cannot be expressed as a typed pattern language using only regular type languages. Using relational patterns, regular types are sufficient to describe L_1.

Proposition 4 *There is a finite set R of relations such that R_1 contains only regular languages and $L_1 \in \mathcal{L}_{\Sigma,R}$.*

Proof. If $R = \{r_1, r_2, r\}$, $r_1 = \{a^i \mid i \geq 1\}$, $r_2 = \{b^i \mid i \geq 1\}$, $r = \{(w_1, w_2) \mid |w_1| = |w_2|\}$, and $v_R = \{(r_1, x_1), (r_2, x_2), (r, x_1, x_2)\}$ then $L(x_1 x_2, v_R) = L_1$. □

Since erasing pattern languages can be expressed as relational pattern languages, Reidenbach's non-learnability results for erasing pattern languages [10] immediately yield the following theorem.

Theorem 5 *There is a finite alphabet Σ and finite set R of recursive relations such that $\mathcal{L}_{\Sigma,R}$ is not learnable in the limit from positive data.*

However, if we disallow empty substitutions, we get positive learnability results for any set of recursive relations.

Theorem 6 *Let R be a finite set of recursive relations with $r \subseteq \Sigma^+$ for all $r \in R_1$. Then $\mathcal{L}_{\Sigma,R}$ is learnable in the limit from positive data.*

To prove this, we use a well-known result due to Angluin [2], according to which every *indexable class* of languages that has *finite thickness* is learnable in the limit from positive data. A class $\mathcal{L}$ of languages is indexable if there is an enumeration $(L_i)_{i\in\mathbb{N}}$ with $\mathcal{L} = \{L_i \mid i \in \mathbb{N}\}$ and an effective procedure d that decides, given any $i \in \mathbb{N}$ and $w \in \Sigma^*$, whether or not $w \in L_i$ [2]. $(L_i)_{i\in\mathbb{N}}$

is then called an indexing for $\mathcal{L}$. An indexable class $\mathcal{L}$ has finite thickness if, for every $w \in \Sigma^*$, the set of languages in $\mathcal{L}$ that contain w is finite. One can establish both indexability and finite thickness to prove Theorem 6. The proofs of both these properties use standard techniques and are omitted because of space constraints. It should be noted though that our results show in particular, that Theorem 6 can be witnessed by a learner that returns relational patterns as its hypotheses—a desirable feature from an application point of view, since relational patterns provide an intuitive representation of a language.

Lemma 7 *Let R be a finite set of recursive relations with $r \subseteq \Sigma^+$ for all $r \in R_1$. There exists an effective enumeration $f : \mathbb{N} \to Pat_{\Sigma,R}$ of all relational patterns over R such that $(L(f(i)))_{i\in\mathbb{N}}$ is an indexing for $\mathcal{L}_{\Sigma,R}$.*

Lemma 8 *Let R be a finite set of recursive relations with $r \subseteq \Sigma^+$ for all $r \in R_1$. Then $\mathcal{L}_{\Sigma,R}$ has finite thickness.*

In fact, Lemma 8 can be strengthened. An indexable class $\mathcal{L}$ is said to have recursive finite thickness [8] if there is an indexing $(L_i)_{i\in\mathbb{N}}$ for $\mathcal{L}$ and a recursive procedure c such that, for any $w \in \Sigma^*$, $c(w)$ is a finite subset of $\mathbb{N}$ fulfilling $[w \in L_i \leftrightarrow \exists j \in c(w)\ [L_j = L_i]]$, *i.e.*, for every word w a finite list of indices for all languages in $\mathcal{L}$ containing w can be effectively determined.

Theorem 9 *Let R be a finite set of recursive relations with $r \subseteq \Sigma^+$ for all $r \in R_1$. Then $\mathcal{L}_{\Sigma,R}$ has recursive finite thickness.*

The proof is omitted due to space constraints. Theorem 9 has some nice consequences, which follow immediately from the literature on recursive finite thickness [8, 7]. For the following corollary, note that an iterative learner [14] is restricted to learn without access to prior data at any point in time. Its hypothesis on a text segment $(\tau(0), \ldots, \tau(n))$ is determined only by $\tau(n)$ and its hypothesis generated on $(\tau(0), \ldots, \tau(n-1))$ (or a dummy hypothesis in case $n = 0$).

Corollary 10 *Let R be a finite set of recursive relations with $r \subseteq \Sigma^+$ for all $r \in R_1$. Let $k \in \mathbb{N}$. Then the class of all unions of up to k languages from $\mathcal{L}_{\Sigma,R}$ is learnable in the limit from positive data using an iterative learner.*

4 The membership problem

Many algorithms for learning formal languages make a hypothesis only if the corresponding language is proven to be consistent with the observed data. This typically requires solving several instances of the *membership problem*. If $\mathcal{P} \subseteq Pat_\Sigma$ is a set of patterns and $W \subseteq \Sigma^*$ a set of words, then the erasing (non-erasing) membership problem for $(\mathcal{P}, W)$ is decidable if there is an effective procedure that, given any $p \in \mathcal{P}$ and any $w \in W$, decides whether or not $w \in L_{NE}(p)$ ($w \in L_E(p)$, resp.). Similarly, we can define the membership problem for relational pattern languages.

The fact that both the non-erasing and the erasing membership problem for (Pat_Σ, Σ^*) are NP-complete [1, 4] is an obstacle for the design of efficient learning algorithms for pattern languages. In this section, we study the complexity of subproblems of the memberhip problem for relational pattern languages.

The following consequence of Angluin's result is straightforward.

Theorem 11 *Let R be a finite set of recursive relations. Then the membership problem for $(Pat_{\Sigma,R}, \Sigma^*)$ is NP-hard.*

However, when the number of variables occurring in a relational pattern is bounded a priori, we get a positive result, which generalizes Angluin's result that the non-erasing membership problem is polynomial-time decidable if the patterns contain at most k variables, for some k [1].

Theorem 12 *Let R be a set of finite relations, each of which is decidable in polynomial time. Let $k \in \mathbb{N}$. Then the membership problem for $(\{(p, v_R) \in Pat_{\Sigma,R} \mid p$ contains at most k distinct variables$\}, \Sigma^*)$ is decidable in polynomial time.*

Proof. Given a relational pattern (p, v_R) over R and a word w, the following procedure decides whether $w \in L(p, v_R)$ in time polynomial in $|p|$, $|v_R|$, and $|w|$.

1. Let $z \leq k$ be the number of distinct variables in p. List all tuples $(w_1, \ldots, w_z)$ of up to z many substrings of w, for which $w_1 \circ \ldots \circ w_z$ is a subsequence of w. (Note: as k is constant, the number of such tuples is polynomial in $|w|$.)
2. For each tuple $(w_1, \ldots, w_z)$ thus listed, define a substitution θ by substituting the z variables in p in order with the words $w_1, \ldots, w_z$; then test whether *(i)* $\theta \in \Theta_{(p,v_R),\Sigma}$ and *(ii)* $\theta(p) = w$. (Note: these tests can be done in polynomial time, since all relations in R can be decided in polynomial time.)
3. If there is one tuple $(w_1, \ldots, w_z)$ for which the test on both *(i)* and *(ii)* is positive, return *yes*, otherwise return *no*.

The correctness and efficiency of the procedure follow immediately. □

In contrast to this, in the context of relational patterns, it is impossible to get an equivalent of Shinohara's [13] result stating that the (non-)erasing membership problem is polynomial-time decidable when restricted to the class of all patterns in which no variable occurs twice, so called *regular patterns*. Since relational patterns can always be expressed equivalently without using any variable twice, Theorem 11 yields NP-hardness of the membership problem for relational patterns with recursive relations and without repetition of variables.

For erasing regular pattern languages, Shinohara's result can be extended:

Theorem 13 *Let $k \in \mathbb{N}$. Then the erasing membership problem for $(\{p \in Pat_\Sigma \mid$ there are at most k variables that occur multiple times in $p\}, \Sigma^*)$ is decidable in polynomial time.*

Proof. For a given $p \in Pat_\Sigma$ with at most k repeated variables and for $w \in \Sigma^*$, the number of ways in which only the repeated variables in p can be replaced by subwords of w is (loosely) upper-bounded by $\binom{|w|}{2k}$ (for each of the up to k repeated variables, one fixes a start position and an end position of the first location in w in which the variable could be substituted), which is polynomial in $|w|$. Replacing only the repeated variables by words in this way maps p to a regular pattern whose length is polynomial in $|w| + |p|$. Obviously, $w \in L_E(p)$ iff w is generated by one of these regular patterns. Since, according to Shinohara [13], the erasing membership problem for regular patterns is polynomial-time decidable, it follows that $w \in L_E(p)$ can be decided in polynomial time as well. □

To our knowledge, the literature has so far not dealt with the question of the complexity of the membership problem when the class of patterns is not severely restricted, yet the set of words is. Since many real-world applications deal with an a priori restricted set of words, it seems reasonable to focus our attention on such problems. For example, in bioinformatics applications, one often has an upper bound on the length of RNA sequences or amino acid sequences that will occur in a database, due to restrictions on either molecular size or the length of snippets collected in experiments. We hence focus on the membership problem for $(\mathcal{P}, \Sigma^{\leq m})$ for $m \in \mathbb{N}$ and for large classes $\mathcal{P}$ of (relational) patterns. Here $\Sigma^{\leq m}$ denotes the set of words of length at most m over Σ.

For classical patterns, the non-erasing membership problem for $(Pat_\Sigma, \Sigma^{\leq m})$ clearly is polynomial-time decidable, since the length of a pattern generating a word w is upper-bounded by $|w|$. However, for erasing pattern languages and for the general case of relational pattern languages, a similar statement does not follow that obviously. The following main result of this section states that, for words of length at most m, the membership problem for a very general class of relational patterns is polynomial-time decidable. Note that this does not provide practical solutions in general, since the length bound m, which occurs in the exponent in our complexity bound, might be rather large in practice.

Theorem 14 *Let R be a set of polynomial-time decidable relations and $m \in \mathbb{N}$. Then the membership problem for $(Pat_{\Sigma,R}, \Sigma^{\leq m})$ is decidable in polynomial time.*

Proof. Let $(p, v_R) \in Pat_{\Sigma,R}$ and $w \in \Sigma^{\leq m}$. Let R' be the set of relations resulting from R when every unary relation t is replaced by $t \setminus \{\epsilon\}$, *i.e.*, $R' = (R \setminus R_1) \cup \{(t \setminus \{\epsilon\} \mid t \in R_1\}$.

We say that $((p, v_R), w)$ fulfills Property $(*)$ if there is a relational pattern $(q, v_{R'}) \in Q_{(p,v_R)}$ with $|q| \leq |w|$, and a substitution $\theta_q \in \Theta_{(q,v_{R'}),\Sigma}$, such that $\theta_q(q) = w$ and $\theta \in \Theta_{(p,v_R),\Sigma}$, where θ restricted to the variables in q equals θ_q and $\theta(x) = \epsilon$ for all other variables. Here $Q_{(p,v_R)}$ is the set of all relational patterns $(q, v_{R'}) \in Pat_{\Sigma,R'}$ where

1. q results from p by removing arbitrarily many variables from p
2. $v_{R'} = \{(r, y_1, \ldots, y_n) \in v_R \mid y_i \text{ occurs in } q \text{ for all } 1 \leq i \leq n,\ n \geq 2\}] \cup \{(t \setminus \{\epsilon\}, x) \mid (t, x) \in v_R \text{ and } x \text{ occurs in } q\}$.

First, we claim that $w \in L(p, v_R)$ iff $((p, v_R), w)$ fulfills Property $(*)$: If $w \in L(p, v_R)$, then there is a substitution $\theta_p \in \Theta_{(p,v_R),\Sigma}$ such that $\theta_p(p) = w$. Let q be the pattern resulting from p after deletion of all variables x with $\theta(x) = \epsilon$. Obviously, $|q| \leq |w|$ and there is a $\theta_q \in \Theta_{(q,v_{R'}),\Sigma}$ with $\theta_q(q) = w$, where $v_{R'}$ is defined as in Property $(*)$.2. Clearly, $(q, v_{R'}) \in Q_{(p,v_R)}$. It remains to show that $\theta \in \Theta_{(p,v_R),\Sigma}$, where θ restricted to the variables in q equals θ_q and $\theta(x) = \epsilon$ for all other variables. This follows from $\theta = \theta_p$. Hence $((p, v_R), w)$ fulfills Property $(*)$. If $((p, v_R), w)$ fulfills Property $(*)$, it follows just as easily that $w \in L(p, v_R)$.

Second, we show that Property $(*)$ can be tested in polynomial time in $|p|$ and $|v_R|$: To construct a list of all $(q, v_{R'}) \in Q_{(p,v_R)}$ with $|q| \leq |w|$, it suffices to consider all sets S of at most $|w|$ many distinct variables occurring in p and to list, for each such S, the relational pattern $(\theta'(p), v_{R'})$ with $v_{R'}$ as in Property $(*)$.2, where

$$\theta'(x) := \begin{cases} x\,, & \text{if } x \in S \cup \Sigma\,, \\ \epsilon\,, & \text{otherwise}\,. \end{cases}$$

With $|S| \leq |w| \leq m$, it follows that at most

$$\sum_{i=0}^{|w|} \binom{|p|}{i} \leq \sum_{i=0}^{|w|} |p|^i \leq (m+1) \cdot |p|^m = O(|p|^m)$$

many relational patterns $(q, v_{R'})$ are listed. Theorem 12 implies that, for these listed relational patterns $(q, v_{R'})$, membership of w in $L(q, v_{R'})$ can be tested in time polynomial in m. If all these membership tests are negative, then Property $(*)$ is not fulfilled. Each positive membership test yields a substitution $\theta_q \in \Theta_{(q,v_{R'})}$ with $\theta_q(q) = w$. One then tests whether $\theta \in \Theta_{(p,v_R),\Sigma}$, where θ is defined as in Property $(*)$; each of these tests can be done in $O(|v_R| \cdot |p|)$. Property $(*)$ is fulfilled if and only if one of these tests is positive. With m being fixed, the total run-time is polynomial in $|v_R|$ and $|p|$. □

5 Probabilistic relational patterns

We have identified two problems that are impossible or hard to solve for complex classes of relational patterns: *(i)* learning such patterns from text and *(ii)* the membership problem for such patterns for all words $w \in \Sigma^*$. There is reason to assume that real-world instantiations of these problems can be solved more easily. To model realistic scenarios more closely, we assume that certain words in a (relational) pattern language are more likely than others, and thus introduce the class of *probabilistic relational patterns*. Our approach addresses the two issues above in the following way.

Learning from text. In applications, not all words are equally likely to occur in a text for a target language. Hence it seems unnecessary to demand from a learner to be successful on all texts of a target language. In this section, we modify Gold's success criterion by considering probability distributions over types in the pattern language and demanding learnability only from texts that are

"sufficiently likely." Studying the learnability of *probabilistic* relational pattern languages in our new model, we obtain upper bounds on the number of relational patterns that have to be considered to identify the target language.

The membership problem. As a side-effect of our model of probabilistic relational patterns, we obtain a simple and potentially practical mechanism for testing membership of words of arbitrary length.

We choose to model probabilistic relational pattern languages via probability distributions on *types*, *i.e.*, for any type r and any subset $A \subseteq r$, a variable of type r is substituted by a word from subset A with a certain probability. We have to make sure though that the collection of words substituted for distinct variables does not violate any of the higher-order relations in the pattern.

Definition 15 *Let R be a set of relations. A* probabilistic relational pattern *over R is a triple (p, v_R, pr_{R_1}), where (p, v_R) is a relational pattern over R and, for each $r \in R_1$, pr_{R_1} contains exactly one probability distribution on r. For $r \in R_1$ and $A \subseteq r$, the probability of A with respect to r is then denoted by $pr_r(A)$. A probabilistic text for (p, v_R, pr_{R_1}), if existent, is an infinite sequence of words in $L(p, v_R)$, each of which is formed by the following stochastic process:*

1. *For each variable x in p, for $(r, x) \in v_R$, draw a substitution $\theta(x) \in r$ according to pr_r.*
2. *If, for all $n \in \mathbb{N}_+$, all $r' \in R_n$, and all $(r', x_{i_1}, \ldots, x_{i_n}) \in v_R$, the substitutions drawn fulfill $(\theta(x_{i_1}), \ldots, \theta(x_{i_n})) \in r'$, then return the word resulting from the substitutions drawn in step 1. Otherwise, repeat from step 1.*

Unlike Gold's definition, we define the success criterion with respect to a class of patterns rather than pattern languages. This is necessary since two relational patterns could describe the same language L but differ in the probability distribution defined on L.

Definition 16 *Let $\mathcal{P}$ be a class of probabilistic relational patterns and $\delta \in (0, 1)$. $\mathcal{P}$ is* δ-learnable in the limit from likely texts *if there is a partial recursive mapping S such that, for any $(p, v_R, pr_{R_1}) \in \mathcal{P}$, if $(\tau(i))_{i \in \mathbb{N}}$ is any probabilistic text for (p, v_R, pr_{R_1}), then, $S(\tau(0), \ldots, \tau(n))$ is defined for all n and, with probability at least $1 - \delta$, there is a relational pattern (q, v'_R) such that $L(q, v'_R) = L(p, v_R)$ and $S(\tau(0), \ldots, \tau(n)) = q$ for all but finitely many n.*

The parameter δ determines a confidence level with which the learner S is supposed to identify a pattern from a probabilistic text, if this text is generated according to Definition 15. Learning is still a limit process, but success is only required on a portion of all possible texts.

Our model has some similarities with stochastic finite learning [11, 12]. In the latter model though, learning is a finite process and the data stream is generated from a probability distributions over the learning domain (which is in our case Σ^*) rather than over types. Kearns and Pitt [5] studied PAC learning of pattern languages generated by patterns with a bounded number of variables, restricting the class of possible distribution of words by considering only a particular kind

of substitution. The way they restrict substitutions differs from our model, and their learner expects both positive and negative examples to learn from.

Prior research on (efficient) learning of subclasses of erasing pattern languages mostly studied patterns in which the number of variables is a priori upper-bounded [11, 12, 15]. However, in many applications, the number of variables in target patterns is huge, but the arity of relations is limited. For instance, in our simplified bioinformatics model, the maximal arity of relations in relational patterns that describe the two-dimensional structure of RNA sequences is 2. Therefore, we study relational patterns in which the arity of relations is bounded and the number of distinct variables is unbounded.

Definition 17 *Let $k \in \mathbb{N}$, R a finite set of recursive relations, and (p, v_R) a relational pattern. (p, v_R) is called* k-simple, *if*

1. *$R_i = \emptyset$ for all $i > k$,*
2. *for every $x \in X$, $|\{(r, y_1, \ldots, y_{i-1}, x, y_{i+1}, \ldots, y_n) \in v_R \mid n \geq 2,\ 1 \leq i \leq n\}| \leq 1$, and*
3. *if $r \in R$, $(w_1, \ldots, w_i) \in r$, and $w_j = \epsilon$ for some $j \in \{1, \ldots, i\}$ then $w_1 = \ldots = w_i = \epsilon$.*

A k-simple relational pattern contains at most 1 multi-variable relation per variable, with an arity of at most k. These relations, as well as all types in a k-simple pattern, are recursive. Erasing pattern languages generated by patterns with a bounded number of variable repetitions can also be generated by k-simple relational patterns.[6] In addition, relational patterns generating RNA sequences as shown in Figure 1 are 2-simple, if each base participates in at most one bond.

Definition 18 requires additional notation. For a relational pattern (p, v_R), any n-ary relation r, and any $i \in \{1, \ldots, n\}$, $r[i]$ denotes the set $\{w \in \Sigma^* \mid \exists w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n \in \Sigma^*\ [(w_1, \ldots, w_{i-1}, w, w_{i+1}, \ldots, w_n) \in r]\}$ and

$$\text{Allowed}(x) = \bigcap_{r(y_1,\ldots,y_{i-1},x,y_{i+1},\ldots,y_n) \in v_R} r[i]\,.$$

Intuitively, Allowed(x) is the set of all words that can potentially be substituted for x in the relational pattern (p, v_R).

Definition 18 *Let $\pi \in (0, 1)$, $k \in \mathbb{N}$, R a finite set of recursive relations, and (p, v_R, pr_{R_1}) a probabilistic relational pattern. (p, v_R, pr_{R_1}) is called* (k, π)-good *if (p, v_R) is k-simple and, for all $t \in R_1$ and all x with $(t, x) \in v_R$*

1. *$pr_t(\text{Allowed}(x)) > 0$,*
2. *there is a set $A \subseteq t \setminus \{\epsilon\}$ with $\frac{pr_t(\text{Allowed}(x) \cap A)}{pr_t(\text{Allowed}(x))} > 1 - \pi$.*

[6] To our knowledge, learnability of this class has not been studied in the literature before—our positive learnability result presented below in Corollary 21 immediately yields learnability of such erasing pattern languages from likely texts.

The main result of this section shows that (k,π)-good probabilistic relational patterns are learnable with high confidence, by cutting down to a finite set of likely candidate patterns after processing only *a single positive example.*

Theorem 19 *Let $\delta, \pi \in (0,1)$, $k \in \mathbb{N}$, R a finite set of recursive relations, and $\mathcal{P}$ any class of (k,π)-good probabilistic relational patterns with respect to R. Then $\mathcal{P}$ is δ-learnable in the limit from likely texts.*

Proof. Let $(p, v_R, pr_{R_1}) \in \mathcal{P}$ be arbitrary and v be the number of *independent variables* in (p, v_R), *i.e.*, variables x such that for all $n \in \mathbb{N}$ and for all $r \in R_n$: $(r, y_1, \ldots, y_{i-1}, x, y_{i+1}, \ldots, y_n) \in v_R$ implies $i = 1$. Let τ be an arbitrary probabilistic text for $L(p, v_R)$. Since (p, v_R, pr_{R_1}) is (k,π)-good, for all $t \in R_1$ and for all variables x of type t in p, there is a set $A \subseteq t \setminus \{\epsilon\}$ fulfilling Properties 1 and 2 of Definition 18. $m_1 = |\tau(0)|$ is an upper bound for the number of independent variables in (p, v_R) that have been substituted by $w \in A$ in $\tau(0)$. Since $pr_t(\text{Allowed}(x) \cap A)/pr_t(\text{Allowed}(x)) > 1 - \pi$, each of the v many independent variables in p are substituted with probability $1 - \pi$ by a word from A.

Using Chernoff bounds with $\mu = v(1-\pi)$, it follows that the probability that fewer than $(1-\lambda_1)\mu = (1-\lambda_1)v(1-\pi)$ independent variables are replaced by a word from A is less than $\exp(-\frac{\mu\lambda_1^2}{2}) = \exp(-\frac{v(1-\pi)\lambda_1^2}{2})$ where $\lambda_1 \in (0,1)$.

First, we determine an upper bound for v with a confidence of at least $1-\delta_1$ for some $\delta_1 \in (0, \frac{\delta}{2})$. We want to upper-bound the probability $Pr[m_1 < (1-\lambda_1)v(1-\pi)]$ (which is $< \exp(-\frac{v(1-\pi)\lambda_1^2}{2})$) by δ_1. This can be achieved when $\exp(-\frac{\mu}{2} + m_1 - \frac{m_1^2}{2\mu}) \leq \delta_1$ (details are omitted), which holds when $\mu + \frac{m_1^2}{\mu} \geq 2m_1 - 2\ln\delta_1$ and in particular when $\mu \geq 2m_1 - 2\ln\delta_1$. The latter is equivalent to $v \geq \frac{1}{1-\pi}(2\cdot m_1 - 2\cdot\ln\delta_1)$. Thus, with confidence level $1-\delta_1$, $\frac{1}{1-\pi}(2\cdot m_1 - 2\cdot\ln\delta_1)$ is an upper bound for the number of independent variables in p.

Second, we compute an upper bound for the number m_2 of independent variables that are substituted by ϵ, with confidence level $1-\delta_2$ for $\delta_2 = \frac{\delta}{2}$. Using Chernoff bounds with an expected value of $\mu' = v\pi$, it follows that

$$Pr[m_2 > (1+\lambda_2)\cdot\mu'] = Pr[m_2 > (1+\lambda_2)\cdot v\cdot\pi] < \left[\frac{\exp(\lambda_2)}{(1+\lambda_2)^{(1+\lambda_2)}}\right]^{v\cdot\pi}.$$

If $\lambda_2 = \min\{\lambda > 0 \mid \left[\frac{\exp(\lambda)}{(1+\lambda)^{(1+\lambda)}}\right]^{v\pi} \leq \delta_2\}$, then $(1+\lambda_2)v\pi + 1 \geq m_2$ with confidence level $1-\delta_2$. Since (p, v_R) is k-simple, a variable can be substituted by ϵ only if it is in relation with a variable that is substituted by ϵ. As $R_i = \emptyset$ for $i > k$ and because of Property 2 in Definition 17, with confidence level $1-\delta_2$, the number of variables in p that are substituted by ϵ is at most $m_2 k$.

Thus, with confidence at least $(1-\delta_1)(1-\delta_2) > (1-\delta)$, $m_1 + m_2 k$ is an upper bound for $|p|$. Since R is finite, the set of possible candidates for the target language (with confidence at least $1-\delta$) is finite and, therefore, the target pattern can be identified in the limit from positive data with probability $1-\delta$. □

For typed pattern languages defined over a finite number of types and with a bounded number of variable repetitions, we obtain a general positive learnability result. They can be represented by a particular kind of k-simple relational

pattern, for which we can prove learnability from likely texts using Theorem 19. The proof is omitted because of space constraints. We require that every variable has a non-zero probability of being replaced with a non-empty word.

Theorem 20 *Let $k \in \mathbb{N}$ and let $\mathcal{L}$ be the class of all typed pattern languages generated using finitely many types and patterns with at most k repetitions per variable. Then there is a set R of relations such that*

1. *$\mathcal{L} \subseteq \mathcal{L}_{\Sigma,R}$, and*
2. *for all $\delta, \pi \in (0,1)$, the class $\{(p, v_R, pr_{R_1}) \mid (p, v_R) \in \mathcal{L}_{\Sigma,R}$ and $\forall t \in R_1\ [pr_t(t \setminus \{\epsilon\}) > 1 - \pi]\}$ is δ-learnable from likely texts.*

Corollary 21 *Let $k \in \mathbb{N}$ and let $\mathcal{L}$ be the class of all erasing pattern languages generated by patterns with at most k repetitions per variable. Then there is a set R of relations such that*

1. *$\mathcal{L} \subseteq \mathcal{L}_{\Sigma,R}$, and*
2. *for all $\delta, \pi \in (0,1)$, the class $\{(p, v_R, pr_{R_1}) \mid (p, v_R) \in \mathcal{L}_{\Sigma,R}$ and $\forall t \in R_1\ [pr_t(t \setminus \{\epsilon\}) > 1 - \pi]\}$ is δ-learnable from likely texts.*

The model of probabilistic relational patterns also yields a straightforward method for testing membership correctly with high confidence.

Observation 22 *Let R be a finite set of recursive relations and let pr_{R_1} contain a probability distribution for each $r \in R_1$. Let $\delta, \pi \in (0,1)$. Let $\mathcal{A}$ be the following algorithm, given a probabilistic relational pattern (p, v_R, pr_{R_1}) and $w \in \Sigma^*$:*

1. *Let W be the set of words obtained from $2 - 2\ln(\delta)/\pi$ independent draws from $L(p, v_R)$ as in Definition 15.*
2. *If $w \in W$, return* yes*, else return* no*.*

Then, for any (p, v_R) and any $w \in L(p, v_R)$ whose probability of being generated in (p, v_R, pr_{R_1}) by the process in Definition 15 is at least π, $\mathcal{A}$ detects the membership of w in $L(p, v_R)$ with probability at least $1 - \delta$. For any (p, v_R) and any $w \notin L(p, v_R)$, $\mathcal{A}$ declares non-membership of w in $L(p, v_R)$.

The proof is a simple application of Chernoff bounds that implies that, after $2 - 2\ln(\delta)/\pi$ independent trials, an event with probability at least π occurs with probability at least $1 - \delta$. For example, to detect membership with a confidence of 0.9 for all words whose probability is at least $\pi = 0.00001$, a set W of about 660,518 words would be generated. How efficient such a method would be in practice depends on the efficiency of generating words from relational patterns.

6 Conclusion

We extended the model of (typed) pattern languages by introducing relational pattern languages, whose expressiveness seems more apt for many applications. We studied relational pattern languages both with respect to their learnability

from positive data and with respect to the complexity of the membership problem. We identified interesting sub-problems of the membership problem that can be solved efficiently, in particular the sub-problem of bounded word length, which, to our knowledge, has not even been studied for classical patterns yet.

Our probabilistic version of relational patterns, and the corresponding model of learning from likely texts provide an adaptation for more realistic text mining and bioinformatics scenarios than Gold's original model can deal with. In our new model, a large class of (erasing) probabilistic relational pattern languages can be learned with high confidence. After seeing just one positive example, the learner can effectively restrict the set of likely candidate patterns to a finite set. Finally, we proposed a simple yet potentially efficient method for testing membership for probabilistic relational patterns correctly with high confidence.

References

1. D. Angluin. Finding patterns common to a set of strings. *J. Comput. Syst. Sci.*, 21:46–62, 1980.
2. D. Angluin. Inductive inference of formal languages from positive data. *Inform. Control*, 45:117–135, 1980.
3. E.M. Gold. Language identification in the limit. *Inform. Control*, 10:447–474, 1967.
4. T. Jiang, E. Kinber, A. Salomaa, K. Salomaa, and S. Yu. Pattern languages with and without erasing. *Int. J. Comput. Math.*, 50:147–163, 1994.
5. M. Kearns and L. Pitt. A polynomial-time algorithm for learning k-variable pattern languages from examples. In *COLT*, pages 57–71, 1989.
6. T. Koshiba. Typed pattern languages and their learnability. In *EuroCOLT*, pages 367–379, 1995.
7. S. Lange. Algorithmic learning of recursive languages. Habilitationsschrift, University of Leipzig, 2000.
8. S. Lange and T. Zeugmann. Incremental learning from positive data. *J. Comput. Syst. Sci.*, 53:88–103, 1996.
9. S. Lange, T. Zeugmann, and S. Zilles. Learning indexed families of recursive languages from positive data: A survey. *Theor. Comput. Sci.*, 397:194–232, 2008.
10. D. Reidenbach. Discontinuities in pattern inference. *Theor. Comput. Sci.*, 397:166–193, 2008.
11. R. Reischuk and T. Zeugmann. An average-case optimal one-variable pattern language learner. *J. Comput. Syst. Sci.*, 60:302–335, 2000.
12. P. Rossmanith and T. Zeugmann. Stochastic finite learning of the pattern languages. *Mach. Learn.*, 44:67–91, 2001.
13. T. Shinohara. Polynomial time inference of extended regular pattern languages. In *RIMS Symposium on Software Science and Engineering*, pages 115–127, 1982.
14. R. Wiehagen. Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. *Elektronische Informationsverarbeitung und Kybernetik*, 12:93–99, 1976.
15. K. Wright. Identification of unions of languages drawn from an identifiable class. In *COLT*, pages 328–333, 1989.
16. K. Wright. Inductive identification of pattern languages with restricted substitutions. In *COLT*, pages 111–121, 1990.