Distinguishing Relational Pattern Languages With a Small Number of Short Strings

Robert C. Holte RHOLTE@UALBERTA.CA
University of Alberta, Amii

S. Mahmoud Mousawi MOUSAWI.S.M@GMAIL.COM
University of Regina

Sandra Zilles ZILLES@CS.UREGINA.CA
University of Regina

Editors: Sanjoy Dasgupta and Nika Haghtalab

Abstract

This paper studies the equivalence problem for relational pattern languages, where a relation imposes dependencies between the two strings with which two variables in a pattern can be replaced simultaneously. Our focus is on the question whether the non-equivalence of two relational patterns is witnessed by short strings, namely those generated by replacing variables in the patterns by strings of length bounded by some (small) number z.

After establishing a close connection between this problem and the study of the notions of *teaching dimension* and *no-clash teaching dimension*, we investigate specific classes of relational pattern languages. We show that the smallest number z that serves as a bound for testing equivalence is 2 when the relation between variable substitutions is that of equal string length, and the alphabet size it at least 3. This has interesting implications on the size and form of non-clashing teaching sets for the corresponding languages. By contrast, not even $z = 3$ is sufficient when the constraints require two substituted strings to be the reversal of one another, for alphabets of size 2. We conclude with a negative result on erasing pattern languages.

Keywords: pattern languages, equivalence problem, teaching dimension

1. Introduction

Pattern languages (Angluin, 1980a) have been studied in computational learning theory for over 40 years, due to their appealingly simple definition, their interesting language-theoretic properties, as well as their applications in areas such as bioinformatics (Arikawa et al., 1993), program synthesis (Nix, 1985), database theory (Barceló et al., 2012), and pattern matching (Clifford et al., 2009).

In the original definition (Angluin, 1980a), a pattern is a non-empty finite string of terminal symbols and variables, and the language it generates consists of all strings obtained when substituting variables with non-empty finite strings of terminal symbols. If a variable occurs more than once, all its occurrences have to be replaced by the same string. Various extensions to this model have been studied, for instance erasing pattern languages (Shinohara, 1982) (in which variables may be replaced by the empty string) and typed pattern languages (Geilke and Zilles, 2012; Koshiba, 1995; Wright, 1990) (in which each variable has its own *type*, i.e., set of strings permitted in substitutions).

Repetitions of variables in a pattern can be viewed as a constraint imposed by setting variable positions in *equality* relation, in the sense that the strings to be substituted for these variable positions must be in *equality* relation. In principle, one could allow any other relation between variables x and x' as well, e.g., x must be substituted by the reversal of the string substituted for x', or by a

string shorter than that substituted for x', or by a string containing more occurrences of the terminal symbol a than that substituted for x', etc. This motivated the study of relational pattern languages (Geilke and Zilles, 2011), in which any set of variables in a pattern can impose any kind of relation on the strings to be substituted for those variables.

The learnability of pattern languages and some of their variants has been studied extensively in the literature; this includes studies on learning in the limit, e.g., (Angluin, 1980a; Reidenbach, 2008), learning from queries, e.g., (Angluin, 1988; Lange and Zilles, 2003), and learning from helpful teachers, e.g., (Bayeh et al., 2020; Gao et al., 2017).

A problem that is of relevance to the design and analysis of learning algorithms in this context is the so-called *equivalence problem*: given two patterns, decide whether or not they are equivalent, i.e., whether or not they generate the same language. For Angluin's pattern languages, this problem is decidable in linear time (Angluin, 1980a), while for erasing pattern languages it is not yet known whether the equivalence problem is decidable at all (Freydenberger and Reidenbach, 2010).

This paper studies the equivalence problem for relational pattern languages, with a focus on standardizing the procedure for testing equivalence. In particular, we ask whether it is sufficient to test the membership of (a small number of) *short* strings generated by either of two patterns in order to decide their equivalence. For any pattern π and any $z \in \mathbb{N}$, denote by $L^{(z)}(\pi)$ the set of all strings generated from π by replacing each variable in π by a string of length $\leq z$. Given z, we ask if

> two patterns π and π' are equivalent as soon as all strings in $L^{(z)}(\pi)$ are generated by π' *and* all strings in $L^{(z)}(\pi')$ are generated by π. (*)

This question is of interest for multiple reasons. For example, testing equivalence with a small number of short (i.e., "simple") strings has implications on the computational complexity of deciding equivalence and could also be of relevance for applications in program synthesis, where testing a program against a specification of a few simple strings could result in highly efficient systems. From a formal language point of view, the success of a standardized test of the form (*) sheds light on structural properties of various classes of pattern languages. From a learning-theoretic point of view, our question is closely related to the study of machine teaching, as we will see below.

Our contributions are as follows. Firstly, Section 3 demonstrates that the existence of an integer z such that (*) is fulfilled is equivalent to the existence of so-called non-clashing teaching sets, made up out of "simple" strings, for the underlying class of (relational) pattern languages. Non-clashing teaching was shown by Kirkpatrick et al. (2019) to be the optimal model of machine teaching that satisfies the widely accepted notion of collusion-freeness proposed by Goldman and Mathias (1996). We also prove implications of property (*) on the classical teaching dimension (Goldman and Kearns, 1995; Shinohara and Miyano, 1991).

Secondly, noting that Angluin's pattern languages satisfy (*) already for $z = 1$, Section 4 shows that (*) is *not* fulfilled with $z = 1$ for two natural classes of relational pattern languages, namely those generated by patterns for which two related variables are replaced (i) by strings of *equal length*; (ii) by strings w, w' such that w is the *reversal* of w'.

The main contribution of Section 5 is to prove that patterns with the *equal-length* relation fulfill (*) with $z = 2$ when the alphabet over which the languages are defined has size at least 3. Moreover, we show that not *all* strings generated by substituting variables with strings of length 2 have to be considered for testing equivalence; a set of size linear in the length of the underlying pattern suffices. This yields non-clashing teaching sets consisting of only linearly many strings, where each string has length at most twice the length of the pattern to be taught.

By contrast, Section 6 shows that (*) does not even hold with $z = 3$ for the *reversal* relation, assuming the alphabet is of size 2. We conjecture though that (*) holds with $z = 2$ for the *reversal* relation for alphabets of size at least 5.

Finally, Section 7 shows that no constant value of z has property (*) for Shinohara's class of erasing pattern languages (Shinohara, 1982) (with *equality* relation), for alphabets of size 2, 3, or 4.

2. Preliminaries

In what follows, the term "string" refers to a finite string, exclusively. For any string s, $|s|$ denotes the length of s. Let Σ be a fixed finite alphabet of size at least 2. The elements of Σ are usually denoted by σ, a, or b. A word is a string over Σ. Σ^* denotes the set of all words over Σ, while Σ^+ denotes its subset of all non-empty words and Σ^t the set of all words of length t, for $t \in \mathbb{N}$.

Let X be an infinite set of variables such that $\Sigma \cap X = \emptyset$. The elements of X are usually denoted by x_i ($i \geq 1$), x, or y. A *pattern* is then a non-empty string over $\Sigma \cup X$, in which no variable occurs twice and the variables occurring in p are $x_1, x_2, \ldots, x_N$, for some N, read from left to right. For example, $p = bax_1x_2ax_3bbx_4x_5x_6$ is a pattern over $\Sigma = \{a, b\}$, but $bax_1x_2ax_3bbx_2x_5x_6$ is not since the variable x_2 is repeated.

A relational pattern is a pair (p, R), where p is a pattern and R is a binary relation over the set of variables in p. A *substitution of* p is any mapping θ that assigns to each variable occurring in p a non-empty word.[1] Then $\theta(p)$ is the word obtained from p after replacing each variable x in p by $\theta(x)$, in particular, θ can be seen as a homomorphism wrt concatenation over symbols from $\Sigma \cup X$ that leaves symbols in Σ unchanged.

A *word relation*, usually denoted r, is any decidable subset of $\Sigma^+ \times \Sigma^+$.[2] Word relations now define which substitutions are "legal" by giving semantics to the relation $R \subseteq X \times X$.

Definition 1 *Let r be a word relation and (p, R) a relational pattern. A substitution θ for p is* legal *for (p, R) wrt r, if $R(x, y)$ implies $r(\theta(x), \theta(y))$ for all variables x, y occurring in p. The language generated by (p, R) under r, denoted $L_r(p, R)$, is the set of words obtained from p by applying substitutions to p that are legal for (p, R) wrt r, i.e.,*

$$L_r(p, R) = \{\theta(p) \mid \theta \text{ is a legal substitution for } (p, R) \text{ wrt } r\}.$$

Definition 2 *Let θ be any substitution of a pattern p and $z \in \mathbb{N}$. If $|\theta(x)| \leq z$ for all variables x in p, then θ is called an ℓ_z-substitution (of p). The (finite) language $L_r^{(z)}(p, R)$ is the set of all the words $\theta(p)$ produced by ℓ_z-substitutions θ that are legal for (p, R) wrt r.*

For example, $L_r^{(1)}(p, R)$ is the set of all and only the words in $L_r(p, R)$ of length $|p|$.

In Angluin's original work (Angluin, 1980a), as well as in most subsequent studies of pattern languages, the only word relation that was considered was *equality*, i.e., $r(w, w')$ iff $w = w'$. For example, the pattern $q = aax_1bx_2aabx_3x_4$ could be constrained by "equating" variables x_3 and x_4, resulting in the relational pattern (q, R) with $q = aax_1bx_2aabx_3x_4$ and $R = \{(x_3, x_4)\}$. $L(q, R)$ would then consist of all words of the form $aawbw'aabw''w''$, where $w, w', w'' \in \Sigma^+$. Geilke and

1. The *erasing* case, when variables can be substituted by the empty string, has also been studied (Shinohara, 1982). We focus on non-empty substitutions here, i.e., the *non-erasing* case.
2. For the purpose of this paper, it suffices to consider only binary word relations, but our formal framework could easily be extended to relations of any arity.

Zilles (2011) loosened Angluin's definition by allowing any kind of word relation, not just *equality*. One of the word relations we consider in this paper is the *equal-length* relation, which requires that two variables x_i and x_j in relation must be substituted by words of equal length, i.e., $r(w, w')$ iff $|w| = |w'|$. If the pattern q above is constrained by setting x_3 and x_4 in *equal-length* relation, it generates all words of the form $aawbw'aabw''$, where $w, w', w'' \in \Sigma^+$ and w'' has even length.

If (p, R) is a relational pattern, then the set of variables in p is partitioned into *groups*. Two distinct variables x, y in p are in the same group if $[R](x, y)$, where $[R]$ is the transitive closure of the symmetric and reflexive closure of R (note here that the domain of R is the set of variables occurring in p). Intuitively, in any legal substitution for (p, R), the words substituted for the variables in a single group are dependent on one another. The group of a variable x in the relational pattern (p, R) is then the set $\{y \in X \mid [R](x, y)\}$. Note that x is always a member of the group of x.

Any pair (Π, r) where Π is a class of relational patterns is associated with a class $\mathcal{L}(\Pi, r)$ of relational pattern languages via $\mathcal{L}(\Pi, r) = \{L_r(\pi) \mid \pi \in \Pi\}$. In this paper, we will always assume Π to be the class of *all relational patterns* over a fixed alphabet Σ, but we will vary the word relation over which to interpret such patterns. In particular, we will study

1. the *equality* relation r_{eq} with $r_{eq}(w, w')$ iff $w = w'$, which induces the class of Angluin's pattern languages;
2. the *equal length* relation r_{len} with $r_{len}(w, w')$ iff $|w| = |w'|$;
3. the *reversal* relation r_{rev} with $r_{rev}(w, w')$ iff w is the reversal of w'.

Wlog, let Π be indexed in a way that membership of any word w in any language $L(\pi)$, $\pi \in \Pi$ can be decided effectively (such indexing obviously exists). Let r be any word relation. Clearly, for any word $w \in \Sigma^+$, there are only finitely many languages in $\mathcal{L}(\Pi, r)$ that contain w. This is because the length of p, for any pattern $(p, R) \in \Pi$ generating w, is bounded from above by the length of w. Moreover, since word relations are decidable, one can effectively construct all relational patterns $(p, R) \in \Pi$ for which $w \in L_r(p, R)$. Thus, $\mathcal{L}(\Pi, r)$ is a uniformly recursive family with *recursive finite thickness*. The latter property was introduced by Koshiba (1995). According to his definition, a uniformly recursive family $(L_i)_{i \in \mathbb{N}}$ of languages has recursive finite thickness, if there exists an effective procedure that, given any word $w \in \Sigma^*$, produces a set $I \subseteq \mathbb{N}$ of indices such that (i) $w \in L_i$ for all $i \in I$, and (ii) for each j with $w \in L_j$ there exists an $i \in I$ such that $L_i = L_j$.

3. Learning-theoretic Implications of the Decidability of the Equivalence Problem

In this paper, we focus on the question whether the equivalence problem is decidable for specific word relations r, i.e., whether there is an effective procedure that, given any two relational patterns $(p, R), (p', R') \in \Pi$, decides whether or not $L_r(p, R) = L_r(p', R')$. Answering this question has several learning-theoretic implications, some of which are due to the fact that (Π, r) induces a uniformly recursive family of non-empty languages that has the recursive finite thickness property.

Firstly, being able to decide the equivalence between two hypotheses in learning can be of practical interest, for example in program synthesis, in situations when one wants to assess the correctness of a synthesized program on the spot, or generally in similar situations where a learner formulates an equivalence query between a proposed hypothesis and the target hypothesis.

Secondly, it was shown by Koshiba (1995) that any uniformly recursive family of languages that has both (i) recursive finite thickness, and (ii) a decidable equivalence problem, is *conservatively*

learnable in the limit from positive data in Gold's model of learning in the limit (Gold, 1967). Conservativeness is the intuitively desirable, yet non-trivial property that a learner, which sequentially reads words belonging to the unknown target language, does not change its current hypothesis as long as the latter is consistent with the words observed.

Thirdly, decidability of the equivalence problem has implications for the effective construction of so-called teaching sets for relational pattern languages in $\mathcal{L}(\Pi, r)$, due to our first formal result (Theorem 3 below). This result connects our study to the study of the notion of teaching sets as introduced by Goldman and Kearns (1995); Shinohara and Miyano (1991). A teaching set for a hypothesis H in a hypothesis class $\mathcal{H}$ over a universe U[3] is a set $T \subseteq U \times \{0,1\}$ of labelled examples such that H is the only hypothesis in $\mathcal{H}$ that is consistent with T. Consistency of a hypothesis H' with a set $T' \subseteq U \times \{0,1\}$ means that, for all $(u,l) \in T'$, we have $l = 1$ if $u \in H'$, while $l = 0$ if $u \notin H'$. One then writes $H'(u) = l$ and calls l the label of u in H'.

One variation of teaching sets that was introduced recently, namely non-clashing teaching sets, will also play a role in our paper. Non-clashing teaching uses less stringent conditions on teaching sets so as to allow smaller teaching set sizes without making teacher and learner succumb to "collusion". Collusion-freeness, as defined by Goldman and Mathias (1996), stipulates that the learner, when correctly identifying H from a set T of examples labelled consistently with H, will also identify H from any superset of T that is consistent with H. This condition is used to prevent teacher and learner from using unfair coding tricks. Kirkpatrick et al. (2019) showed that the smallest worst-case "teaching set" size that can be obtained in collusion-free teaching is the size of what they call non-clashing teaching sets. Suppose each hypothesis H is assigned a consistent set T_H of labelled examples. The system of these sets is non-clashing iff there are no two distinct hypotheses H and H' in $\mathcal{H}$ such that *both* H is consistent with $T_{H'}$ *and* H' is consistent with T_H. The non-clashing property was also used by de la Higuera (1997) in the notion of characteristic sets.

Conditions under which small teaching sets exist, and how they make machine learning more data-efficient, are the subject of many recent studies in various subfields of machine learning and AI, for example (Alanazi et al., 2020; Cicalese et al., 2020; Dasgupta et al., 2019; Kirkpatrick et al., 2019; Mansouri et al., 2019); see also (Zhu et al., 2018) for a partial overview. Specifically, in the context of pattern languages with *equality* relation, the existence of small teaching sets has been studied both for the non-erasing case (Gao et al., 2017) and for the erasing case (Bayeh et al., 2020).

Theorem 3 *Let $(L_i)_{i\in\mathbb{N}}$ be any uniformly recursive family of non-empty languages, with the recursive finite thickness property. Then the following two statements are equivalent.*

1. *The equivalence problem for $(L_i)_{i\in\mathbb{N}}$ is decidable.*
2. *There is an algorithm that, given $j \in \mathbb{N}$, constructs a finite teaching set for L_j wrt $(L_i)_{i\in\mathbb{N}}$.*

Proof 2 implies 1, since one can trivially test equivalence of L_i and L_j by checking whether the teaching set for L_i is consistent with L_j (or vice versa.)

To see that 1 implies 2, note that a teaching set for L_k wrt $(L_i)_{i\in\mathbb{N}}$ can be constructed as follows. First, one finds a word $w \in L_k$, which is possible since $L_k \neq \emptyset$ and L_k has decidable membership. Second, one exploits recursive finite thickness to construct a set I of indices such that (i) $w \in L_i$ for all $i \in I$, and (ii) for each j with $w \in L_j$ there exists an $i \in I$ such that $L_i = L_j$. Using decidable equivalence, one can remove all indices for L_k from I; let us denote the resulting subset

3. In our context, a hypothesis is a language, a hypothesis class a class of languages, and U the set of words over Σ.

of I by I'. For each $i \in I'$, one can determine a word w_i in the symmetric difference of L_i and L_k, since membership is uniformly decidable. The set of these words w_i, each paired with its label $L_k(w_i) \in \{0, 1\}$, now forms a teaching set for L_k wrt $(L_i)_{i\in\mathbb{N}}$. ■

As a consequence, decidability of the equivalence problem for a set Π of relational patterns is equivalent to the existence of an effective procedure for constructing finite teaching sets for the language family $\mathcal{L}(\Pi, r)$, where r is any word relation according to our definition. For practical reasons, the size of teaching sets as well as the complexity of constructing them are of interest. To this end, we consider the following three questions, which turn out to be closely connected:

1. In order to decide whether $L_r(p, R) = L_r(p', R')$, is it enough to test a small set of short strings in these two languages, specifically those obtained by legal ℓ_z-substitutions, for some small z? In other words, does the following condition hold for some fixed z?
   Condition EQ. $L_r(p, R) = L_r(p', R')$ iff $(L_r^{(z)}(p, R) \subseteq L_r(p', R')$ *and* $L_r^{(z)}(p', R') \subseteq L_r(p, R))$, for all $(p, R), (p', R') \in \Pi$.

2. For some fixed z, can the set $L_r^{(z)}(p, R)$ for $L_r(p, R)$, equipped with positive labels (label 1), always be used to form a system of non-clashing teaching sets wrt $\mathcal{L}(\Pi, r)$?
   Condition NCTS. Mapping $L_r(p, R)$ to the set $\{(w, 1) \mid w \in L_r^{(z)}(p, R)\}$ for all $(p, R) \in \Pi$ yields a system of non-clashing teaching sets for $\mathcal{L}(\Pi, r)$.

3. For some fixed z, is the set of words generated from p by unconstrained ℓ_z-substitutions, equipped with the appropriate labels, always a teaching set for $L_r(p, R)$ wrt $\mathcal{L}(\Pi, r)$?
   Condition TS. For all $(p, R) \in \Pi$, the set $\{(w, L_r(p, R)(w)) \mid w \in L_r^{(z)}(p, \emptyset)\}$ is a teaching set for $L_r(p, R)$ wrt $\mathcal{L}(\Pi, r)$.[4]

Theorem 4 *Let r be a word relation and $z \in \mathbb{N}$. Then r, z fulfill Condition EQ iff r, z fulfill Condition NCTS. Moreover, if r, z fulfill Condition EQ then r, z fulfill Condition TS.*

Proof First suppose that r, z fulfill Condition EQ. Then r, z obviously fulfill Condition NCTS. To see that r, z fulfill Condition TS, suppose $(p', R') \in \Pi$ is consistent with $T = \{(w, L_r(p, R)(w)) \mid w \in L_r^{(z)}(p, \emptyset)\}$ for some $(p, R) \in \Pi$. Then all legal ℓ_z-substitutions for (p', R') generate strings $w \in L_r^{(z)}(p, \emptyset)$ that are also contained in $L_r(p, R)$. Likewise, all legal ℓ_z-substitutions for (p, R), which must have the label 1 in T, are contained in $L_r(p', R')$ due to consistency of the latter with T. By Condition EQ then $L_r(p, R) = L_r(p', R')$, i.e., T is a teaching set for $L_r(p, R)$ wrt $\mathcal{L}(\Pi, r)$.

Finally, suppose r, z fulfill Condition NCTS. Let $(p, R) \in \Pi$. Obviously, $L_r(p, R) = L_r(p', R')$ implies $(L_r^{(z)}(p, R) \subseteq L_r(p', R')$ *and* $L_r^{(z)}(p', R') \subseteq L_r(p, R))$. The opposite direction follows immediately from Condition NCTS. Thus, Condition EQ is satisfied for r, z. ■

Due to the close relation between Condition EQ and the existence of finite (non-clashing) teaching sets consisting of short (and thus "simple") strings, we devote the remainder of this paper to the study of Condition EQ for relational pattern languages. Importantly though, it will turn out that the (non-clashing) teaching sets that are implied by the constructions in our proofs of Condition EQ are typically smaller than the bounds implied by Conditions NCTS/TS suggest.

4. Recall that $H(w)$ denotes the label of w in H, so here, using $H = L_r(p, R)$, the term $L_r(p, R)(w)$ equals 1 if $w \in L_r(p, R)$, and 0 otherwise.

4. ℓ_1-Substitutions

We begin by analyzing a few examples of relations r concerning the question whether r and $z = 1$ fulfill Condition EQ, i.e., we investigate for these relations whether $L_r(p, R) = L_r(p', R')$ is equivalent to $(L_r^{(1)}(p, R) \subseteq L_r(p', R')$ and $L_r^{(1)}(p', R') \subseteq L_r(p, R))$.

Angluin's pattern languages, with equality as the only allowed relation between variables, are a prominent case in which Condition EQ is fulfilled for $z = 1$.

Proposition 5 (Angluin (1980a)) *The word relation r_{eq} fulfills Condition EQ with $z = 1$, i.e., for any two relational patterns (p, R), (p', R'), we have $L_{r_{eq}}(p, R) = L_{r_{eq}}(p', R')$ iff $(L_{r_{eq}}^{(1)}(p, R) \subseteq L_{r_{eq}}(p', R')$ and $L_{r_{eq}}^{(1)}(p', R') \subseteq L_{r_{eq}}(p, R))$.*

Angluin's argument does not require testing the full sets $L_{r_{eq}}^{(1)}(p, R)$ and $L_{r_{eq}}^{(1)}(p', R')$ in order to decide equivalence of (p, R) and (p', R'). Instead, it suffices to consider legal ℓ_1-substitutions that replace one variable x in the respective pattern (as well as all variables in the group of x) with a, while replacing all other variables with b, for some $a, b \in \Sigma$, $a \neq b$. Using the same argument as in the proof of Theorem 4, one therefore immediately obtains non-clashing teaching sets of size at most linear in the length of the underlying pattern (for the class of all non-erasing pattern languages).

By contrast with Proposition 5, one does not obtain Condition EQ for $z = 1$ when the word relation is r_{len} or r_{rev}. The intuitive reason is that (i) for r_{len}, all ℓ_1-substitutions are legal and thus they give no insight about which variables are related; (ii) for r_{rev}, ℓ_1-substitutions help to identify groups of related variables, but if the group size is 3 or larger, they cannot help identify which pairs of variables in a group are in reversal relation. For example, consider the pattern $p = x_1x_2x_3$. Using r_{len}, clearly $(L_{r_{len}}^{(1)}(p, \emptyset) \subseteq L_{r_{len}}(p, \{(x_1, x_2), (x_1, x_3)\})$ and $L_{r_{len}}^{(1)}(p, \{(x_1, x_2), (x_1, x_3)\}) \subseteq L_{r_{len}}(p, \emptyset))$. However, $L_{r_{len}}(p, \emptyset) \neq L_{r_{len}}(p, \{(x_1, x_2), (x_1, x_3)\})$, since the length of each word in $L_{r_{len}}(p, \{(x_1, x_2), (x_1, x_3)\})$ is a multiple of 3. For r_{rev}, clearly $(L_{r_{rev}}^{(1)}(p, \{(x_1, x_2), (x_2, x_3)\}) \subseteq L_{r_{rev}}(p, \{(x_1, x_2), (x_1, x_3)\})$ and $L_{r_{rev}}^{(1)}(p, \{(x_1, x_2), (x_1, x_3)\}) \subseteq L_{r_{rev}}(p, \{(x_1, x_2), (x_2, x_3)\})$. But $L_{r_{rev}}(p, \{(x_1, x_2), (x_2, x_3)\}) \neq L_{r_{rev}}(p, \{(x_1, x_2), (x_1, x_3)\})$, since *abbaba* is a word contained in $L_{r_{rev}}(p, \{(x_1, x_2), (x_1, x_3)\}$, but not in $L_{r_{rev}}(p, \{(x_1, x_2), (x_2, x_3)\}$.

Condition EQ does not hold for $z = 1$ for the word relations r_{len} and r_{rev}, only because ℓ_1-substitutions cannot distinguish between different relation sets R, R' over the same pattern string p. It is not hard to see that ℓ_1-substitutions suffice to reveal differences in the pattern string itself, when the word relation is either r_{len} or r_{rev}:

Proposition 6 *Let $r \in \{r_{len}, r_{rev}\}$. Let $(p, R), (p', R') \in \Pi$ with $L_r^{(1)}(p, R) \subseteq L_r(p', R')$ and $L_r^{(1)}(p', R') \subseteq L_r(p, R)$. Then $p = p'$.*

The proof is identical to that of Angluin's analogous result for r_{eq} (Angluin, 1980a).

5. The Equal-Length Relation

Motivated by the fact that ℓ_1-substitutions do not suffice to test the equivalence of relational patterns over the *equal-length* relation, we want to determine the smallest integer z for which r_{len}, z fulfill Condition EQ. It turns out that this integer is 2, if the alphabet Σ is of size at least 3.

Theorem 7 *Let $|\Sigma| \geq 3$. Then the word relation r_{len} fulfills Condition EQ with $z = 2$, i.e., for any two relational patterns (p, R) and (p', R'), we have $L_{r_{len}}(p, R) = L_{r_{len}}(p', R')$ iff ($L^{(2)}_{r_{len}}(p, R) \subseteq L_{r_{len}}(p', R')$ and $L^{(2)}_{r_{len}}(p', R') \subseteq L_{r_{len}}(p, R)$).*

The remainder of this section is dedicated to the proof of Theorem 7. To this end, we will first introduce some more notation and state some simple facts.

A *block* of variables in pattern p is any sequence $x_i x_{i+1} \dots x_{i+m}$ ($m \geq 0$) such that p is of the form $q_1 x_i x_{i+1} \dots x_{i+m} q_2$ for some $q_1, q_2 \in (\Sigma \cup X)^*$ where q_1 does not end in a variable and q_2 does not begin with a variable. That means, a block of variables in p is any maximal substring of p that consists only of variables. Adjacent variables within a block are interchangeable when it comes to satisfying *equal-length* relations. This obvious fact is useful when applied repeatedly, for example to show that $L_{r_{len}}(ax_1x_2x_3abx_4, \{(x_3, x_4)\}) = L_{r_{len}}(ax_1x_2x_3abx_4, \{(x_1, x_4)\})$.

Lemma 8 *Let $(p, R) \in \Pi$, where $p = q_1 x_i x_{i+1} q_2$ with $q_1, q_2 \in (\Sigma \cup X)^*$. Then $L_{r_{len}}(p, R) = L_{r_{len}}(p, R')$, where R' is obtained from R by swapping x_i with x_{i+1} in all relations.*

In the remainder of this section, we always use Lemma 8 implicitly, i.e., without mentioning we make use of the fact that the order of variables inside a block plays no role when testing the equivalence of two relational patterns when r_{len} is the underlying word relation.

Note that any pattern is of the form $w_0 \mathbb{B}_1 w_1 \dots \mathbb{B}_b w_b$ where $w_0, w_b \in \Sigma^*$, $w_k \in \Sigma^+$ ($k \in \{1, \dots, b-1\}$), and each $\mathbb{B}_k$ is a block of variables.

Definition 9 *Let (p, R) be a relational pattern with p having b blocks, $\mathbb{B}_1 \dots \mathbb{B}_b$, and θ a legal substitution (wrt r_{len}). The length tuple corresponding to θ is the b-tuple $\langle |\theta(\mathbb{B}_1)|, \dots, |\theta(\mathbb{B}_b)| \rangle$.*

A word $w \in L(p, R)$ matches *length tuple $\langle m_1, m_2, \dots m_b \rangle$ if there exists a substitution θ that is legal for (p, R) (wrt r_{len}) such that $\theta(p) = w$ and θ's length tuple is $\langle m_1, m_2, \dots m_b \rangle$.*[5] *$LT(p, R)$ is the set of length tuples that correspond to legal substitutions (wrt r_{len}), and $LT^{(2)}(p, R)$ is the set of length tuples that correspond to legal ℓ_2 substitutions (wrt r_{len}).*

Let n be the number of variable groups induced by R. We impose a fixed, but arbitrary order on the groups of variables in p so that the multi-set of group sizes becomes a vector $G = \langle g_1, g_2, \dots, g_n \rangle$. If p contains k variables then $\sum_{i=1}^{n} g_i = k$.

R also specifies how many variables (zero or more) from block $\mathbb{B}_k$ there are in group i. This is called the "decomposition" of $\mathbb{B}_k$ and it too will be written as a vector of length n ordered in the same way as G. If the decomposition of $\mathbb{B}$ is $\langle a_1, a_2, \dots, a_n \rangle$ then $\sum_{i=1}^{n} a_i = |\mathbb{B}|$.

Example 1 *Let $p = x_1x_2bbx_3x_4x_5bbx_6x_7x_8x_9$ and $R = \{(x_1, x_3), (x_1, x_4), (x_2, x_9), (x_5, x_6), (x_5, x_7)\}$. R defines four groups – $\{x_1, x_3, x_4\}$, $\{x_2, x_9\}$, $\{x_5, x_6, x_7\}$, and $\{x_8\}$. Ordered in that way, $G = \langle 3, 2, 3, 1 \rangle$ and the decomposition of the three blocks is as follows: (i) $\mathbb{B}_1(x_1x_2)$ has decomposition $\langle 1, 1, 0, 0 \rangle$; (ii) $\mathbb{B}_2(x_3x_4x_5)$ has $\langle 2, 0, 1, 0 \rangle$; and (iii) $\mathbb{B}_3(x_6x_7x_8x_9)$ has $\langle 0, 1, 2, 1 \rangle$.*

The sum of the values in $\mathbb{B}_k$'s decomposition is $|\mathbb{B}_k|$ and the sum of the values in the i^{th} position of the decomposition vectors is g_i, the size of group i. If the set of block decompositions is regarded as a matrix, then multiplying it by a vector $\langle \gamma_1 \dots, \gamma_n \rangle$ produces a length tuple. For example, multiplying the decompositions in this example by $\langle 2, 2, 2, 1 \rangle$ produces the length tuple $\langle 4, 6, 7 \rangle$. This represents the form of words that are produced when words of length γ_i are substituted for all the variables in group i, which here is $\sigma_{1,1}, \dots \sigma_{1,4}bb\sigma_{2,1}, \dots \sigma_{2,6}bb\sigma_{3,1}, \dots \sigma_{3,7}$, for some $\sigma_{i,j} \in \Sigma$.

5. A word can match more than one length tuple, e.g., if $p = x_1\sigma x_2$ then $w = \sigma\sigma\sigma\sigma$ matches $\langle 2, 1 \rangle$ and $\langle 1, 2 \rangle$.

Lemma 10 *Let $(p, R) \in \Pi$ with p having b blocks, $\mathbb{B}_1 \dots \mathbb{B}_b$, n the number of variable groups induced by R, and $\langle a_{k,1}, a_{k,2}, \dots, a_{k,n}\rangle$ the decomposition of block $\mathbb{B}_k$. Then $\langle m_1, m_2, \dots, m_b\rangle \in LT(p, R)$ iff for each $i \in \{1, \dots, n\}$ there exists $\gamma_i \geq 0$ such that for all $k \in \{1, \dots, b\}, m_k = |\mathbb{B}_k| + \sum_{i=1}^{n} \gamma_i a_{k,i}$.*

Proof Immediate from the definitions of length tuple and decomposition of a block. ■

Our argument for the proof of Theorem 7 can be summarized as follows. First, due to Proposition 6, we only need to prove the statement of Theorem 7 for the case $p = p'$. In particular, we only need to prove that $(L^{(2)}_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')$ and $L^{(2)}_{r_{len}}(p, R') \subseteq L_{r_{len}}(p, R))$ implies $L_{r_{len}}(p, R) = L_{r_{len}}(p, R')$. It turns out that we can prove a stronger statement, namely:

$$L^{(2)}_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R') \Rightarrow L_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')\,. \tag{1}$$

Second, to obtain statement (1), we will prove the following two claims:

1. If $|\Sigma| \geq 3$ and $L^{(2)}_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')$, then $LT^{(2)}(p, R) \subseteq LT(p, R')$. (Theorem 15.)
2. If $LT^{(2)}(p, R) \subseteq LT(p, R')$, then $L_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')$. (Theorem 14.)

Let $p = w_0\mathbb{B}_1 w_1 \dots w_{b-1}\mathbb{B}_b w_b$ be a pattern, where $w_0, w_b \in \Sigma^*$, $w_k \in \Sigma^+$ for each $k \in \{1, \dots, b-1\}$, and each $\mathbb{B}_k$ is a block. Let $(p, R), (p, R') \in \Pi$, and n and n' be the number of variable groups induced by R and R' respectively. The decomposition of block $\mathbb{B}_k$ by R is an n-tuple $\langle a_{k,1}, \dots, a_{k,n}\rangle$ and its decomposition by R' is an n'-tuple $\langle a'_{k,1}, \dots, a'_{k,n'}\rangle$.

With b blocks, length tuples are b-tuples, and $LT^{(2)}(p, R)$ will contain tuple $\langle m_1, \dots, m_b\rangle$ if and only if there exist $\gamma_1, \dots, \gamma_n$ ($\gamma_i \in \{0, 1\}$) such that for all $k \in \{1 \dots b\}$, $m_k = |\mathbb{B}_k| + \sum_{i=1}^{n} \gamma_i a_{k,i}$ (Lemma 10). In particular, for each $i \in \{1 \dots n\}$, $LT^{(2)}(p, R)$ contains length tuple $\langle |\mathbb{B}_1| + a_{1,i}, \dots, |\mathbb{B}_k| + a_{k,i}, \dots, |\mathbb{B}_b| + a_{b,i}\rangle$ (obtained by setting $\gamma_i = 1$ and $\gamma_h = 0$ for all $h \neq i$). We refer to this subset of $LT^{(2)}(p, R)$ as $LT^{(2)}_{base}(p, R)$.

Lemma 11 *Let $(p, R), (p, R') \in \Pi$. If $LT^{(2)}_{base}(p, R) \subseteq LT(p, R')$, then $LT(p, R) \subseteq LT(p, R')$.*

Proof If $LT^{(2)}_{base}(p, R) \subseteq LT(p, R')$ then for each $i \in \{1 \dots n\}$ there exist $\delta_{i,1}, ..., \delta_{i,n'} \in \mathbb{N}$ such that, for every $k \in \{1 \dots b\}$, $|\mathbb{B}_k| + a_{k,i} = |\mathbb{B}_k| + \sum_{j=1}^{n'} \delta_{i,j} a'_{k,j}$, i.e. $a_{k,i} = \sum_{j=1}^{n'} \delta_{i,j} a'_{k,j}$.[6]

If $\langle m_1, \dots, m_b\rangle$ is any length tuple in $LT(p, R)$ then by Lemma 10 there exist $\gamma_1, \dots, \gamma_n \in \mathbb{N}$ such that, for all $k \in \{1, \dots, b\}$, $m_k = |\mathbb{B}_k| + \sum_{i=1}^{n} \gamma_i a_{k,i}$. When $LT^{(2)}_{base}(p, R) \subseteq LT(p, R')$, we can substitute $\sum_{j=1}^{n'} \delta_{i,j} a'_{k,j}$ for $a_{k,i}$ in the summation term to get $\sum_{i=1}^{n} \gamma_i (\sum_{j=1}^{n'} \delta_{i,j} a'_{k,j}) = \sum_{j=1}^{n'} \sum_{i=1}^{n} \gamma_i \delta_{i,j} a'_{k,j} = \sum_{j=1}^{n'} \beta_j a'_{k,j}$, where $\beta_j = \sum_{i=1}^{n} \gamma_i \delta_{i,j} \geq 0$. In other words, $m_k = |\mathbb{B}_k| + \sum_{j=1}^{n'} \beta_j a'_{k,j}$, and therefore (Lemma 10) $\langle m_1, \dots, m_b\rangle \in LT(p, R')$, as required. ■

The proof of the following lemma is detailed in Appendix A.

6. $1 + \delta_{i,j}$ is the length of the words substituted for the variables in R''s group j in order to produce the length tuple $\langle |\mathbb{B}_1| + a_{1,i}, \dots, |\mathbb{B}_k| + a_{k,i}, \dots, |\mathbb{B}_b| + a_{b,i}\rangle$.

Lemma 12 *Let $(p, R) \in \Pi$, where $p = w_0\mathbb{B}_1 w_1 \mathbb{B}_2 w_2 \dots \mathbb{B}_b w_b$, for some $w_0, w_b \in \Sigma^*$, $w_k \in \Sigma^+$ ($k \in \{1, \dots, b-1\}$), and for some blocks $\mathbb{B}_k$ of variables. Then*

$$L_{r_{len}}(p, R) = \bigcup_{\langle m_1, m_2, \dots m_b\rangle \in LT(p,R)} \{w_0\Sigma^{m_1} w_1 \Sigma^{m_2} w_2 \dots \Sigma^{m_b} w_b\}.$$

Corollary 13 *Let $(p, R), (p, R') \in \Pi$. If $LT(p, R) \subseteq LT(p, R')$ then $L_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')$.*

Proof Immediate from Lemma 12. ■

The next result follows directly from the combination of Lemma 11 and Corollary 13.

Theorem 14 *Let $(p, R), (p, R') \in \Pi$. If $LT^{(2)}_{base}(p, R) \subseteq LT(p, R')$ then $L_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')$.*

The only step that remains to establish Theorem 7 is proving the following theorem.

Theorem 15 *Suppose $|\Sigma| \geq 3$, and let $(p, R), (p, R') \in \Pi$. If $L^{(2)}_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')$ then $LT^{(2)}_{base}(p, R) \subseteq LT(p, R')$.*

Proof Let $p = w_0\mathbb{B}_1 w_1 \mathbb{B}_2 w_2 \dots \mathbb{B}_b w_b$, for some $w_0, w_b \in \Sigma^*$, $w_k \in \Sigma^+$ ($k \in \{1, \dots, b-1\}$), and for some blocks $\mathbb{B}_k$ of variables. Suppose $L^{(2)}_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')$ and $\langle m_1, \dots, m_b\rangle \in LT^{(2)}_{base}(p, R)$. We need to show that $\langle m_1, \dots, m_b\rangle \in LT(p, R')$.

To this end, fix a legal ℓ_2-substitution θ for (p, R) wrt r_{len}, with the length tuple $\langle m_1, \dots, m_b\rangle$. Specifically, choose θ in a way that, for all k, the block $\mathbb{B}_k$ is replaced with $a_k^{m_k}$, where $a_k \in \Sigma$ is any symbol different from the first symbol in w_k as well as from the last symbol in w_{k-1}. This is possible since $|\Sigma| \geq 3$. Let $w = \theta(p)$. Since $w \in L^{(2)}_{r_{len}}(p, R)$, we have $w \in L_{r_{len}}(p, R')$. Hence, there is a substitution θ' that is legal for (p, R') wrt r_{len} and satisfies $\theta'(p) = w$.

First, note that $w_0 a_1^{m_1}$ must be a prefix of $\theta'(w_0\mathbb{B}_1)$. This is because a_1 is different from the first symbol in w_1, which is the symbol following $\theta'(w_0\mathbb{B}_1)$ in w. Second, we argue that $w_0 a_1^{m_1} w_1 a_2^{m_2}$ is a prefix of $\theta'(w_0\mathbb{B}_1 w_1 \mathbb{B}_2)$. To see this, note that $w_0 a_1^{m_1} w_1$ is a prefix of $\theta'(w_0\mathbb{B}_1 w_1)$. So, the replacement θ' makes for $\mathbb{B}_2$ begins somewhere after $w_0 a_1^{m_1} w_1$ in w. Then it cannot end before the end of $w_0 a_1^{m_1} w_1 a_2^{m_2}$ in w, since otherwise w_2 would start with a_2. We apply the same argument inductively to obtain that $w_0 a_1^{m_1} w_1 a_2^{m_2} \dots w_{k-1} a_k^{m_k}$ is a prefix of $\theta'(w_0\mathbb{B}_1 w_1 \mathbb{B}_2 \dots w_{k-1}\mathbb{B}_k)$ for all $k \in \{1, \dots, b\}$. Analogously, one can argue that $a_k^{m_k} w_k \dots a_b^{m_b} w_b$ is a suffix of $\theta'(\mathbb{B}_k w_k \dots \mathbb{B}_b w_b)$ for all $k \in \{1, \dots, b\}$. Together, these statements imply that $\theta'(\mathbb{B}_k) = \theta(\mathbb{B}_k)$ for all $k \in \{1, \dots, b\}$. Therefore $\langle m_1, \dots, m_b\rangle$ is also the length tuple associated with θ', and thus $\langle m_1, \dots, m_b\rangle \in LT(p, R')$ as required. ■

A closer inspection of our proofs shows that not all legal ℓ_2-substitutions for two relational patterns (p, R), (p, R') are relevant for testing their equivalence wrt r_{len}. In fact, one can associate with each group of variables a word of length 2 and a word of length 1. Then one only ever needs to check legal substitutions that replace (i) variables *inside* a group with its associated word of length 2, and (ii) variables *outside* that group with its associated word of length 1. The number of such substitutions is only linear in the number of groups in p, and thus linear in the length of p. If all the resulting words for (p, R) belong to $L_{r_{len}}(p, R')$, then $L_{r_{len}}(p, R) \subseteq L_{r_{len}}(p, R')$. To sum up:

Corollary 16 *Let $|\Sigma| \geq 3$. Then there is a computable mapping that assigns every relational pattern (p, R) a set $S(p, R) \subseteq L^{(2)}_{r_{len}}(p, R)$ such that the following properties hold.*

1. *For any $(p, R) \in \Pi$, the cardinality $|S(p, R)|$ is linear in $|p|$.*
2. *For any $(p, R), (p', R') \in \Pi$, we have $L_{r_{len}}(p, R) = L_{r_{len}}(p', R')$ iff $S(p, R) \subseteq L_{r_{len}}(p', R')$ and $S(p', R') \subseteq L_{r_{len}}(p, R)$.*

Using the same argument as in the proof of Theorem 4, we obtain the following corollary, which states intuitively that relational pattern languages over the *equal-length* relation can be learned in the non-clashing teaching model using *linear-size teaching sets* consisting of *"short" words*.

Corollary 17 *Let $|\Sigma| \geq 3$. Then the class $\{L_{r_{len}}(p, R) \mid (p, R)$ is a relational pattern$\}$ has a system of non-clashing teaching sets such that the non-clashing teaching set assigned to a language $L(p, R)$ is of size $O(|p|)$ and contains only words of length at most $2|p|$.*

6. The Reversal Relation

As explained in Section 4, the word relation r_{rev} does not satisfy Condition EQ with $z = 1$. The question for the smallest z such that r_{rev}, z fulfill Condition EQ is not resolved in our paper. We conjecture that $z = 2$ suffices when the underlying alphabet Σ has at least 5 symbols. By contrast, if $|\Sigma| = 2$, not even $z = 3$ is sufficient, as the next theorem shows.

Theorem 18 *Suppose $|\Sigma| = 2$. Then there exists a pattern p and relations R, R' such that $L^{(3)}_{r_{rev}}(p, R) \subseteq L_{r_{rev}}(p, R')$ and $L^{(3)}_{r_{rev}}(p, R') \subseteq L_{r_{rev}}(p, R)$, yet $L_{r_{rev}}(p, R) \neq L_{r_{rev}}(p, R')$.*

Proof Let $\Sigma = \{a, b\}$ and let $p = v_1\, aa\, x\, aa\, v_2\, aa\, y\, a\, v_3 z v_4$, where $x, y, z, v_i \in X$, $a \in \Sigma$, $R = \{(x, z), (z, y)\}$, and $R' = \{(x, y), (z, y)\}$. Legal substitutions for (p, R) replace x and y with the same word and z with its reversal, while those for (p, R') replace x and z with the same word and y with its reversal. It remains to verify that (i) $L^{(3)}_{r_{rev}}(p, R) \subseteq L_{r_{rev}}(p, R')$, and (ii) $L^{(3)}_{r_{rev}}(p, R') \subseteq L_{r_{rev}}(p, R)$, while (iii) $L_{r_{rev}}(p, R) \neq L_{r_{rev}}(p, R')$. The details are given in Appendix B. ■

By Proposition 6, two relational patterns (p, R), (p', R') are non-equivalent under r_{rev} if $p \neq p'$; and this is already verified by testing ℓ_1-substitutions for (p, R) and (p', R'). So, as in the case of the *equal-length* relation, also for the *reversal* relation the only non-trivial cases of the equivalence problem are those with two relational patterns using the same pattern string p but two different relation sets R and R'. The two relational patterns (p, R) and (p, R') used in the proof of Theorem 18 have the interesting property that the relation sets R and R' have exactly the same groups of variables: both have the group $\{x, y, z\}$ as well as singleton groups for the remaining variables. It turns out that cases in which the two relational patterns have identical groups are the only ones that will require more than ℓ_1-substitutions to decide equivalence wrt r_{rev}. In particular, under the relation r_{rev}, two relational patterns (p, R) and (p, R') induce exactly the same groups of variables unless their semantic difference is already witnessed by legal ℓ_1-substitutions:

Proposition 19 *Let (p, R) and (p, R') be two relational patterns. Let r_{rev} be the underlying word relation. Then $(L^{(1)}_{r_{rev}}(p, R) \subseteq L_{r_{rev}}(p', R')$ and $L^{(1)}_{r_{rev}}(p', R') \subseteq L_{r_{rev}}(p, R))$ implies that every group of variables in (p, R) is a group of variables in (p, R') and vice versa.*

The proof is straightforward and given in Appendix C.

7. A Note on Erasing Pattern Languages

In this paper, we studied only non-erasing relational pattern languages, i.e., substitutions that replace some variables with the empty string are forbidden. When loosening this constraint, one obtains a negative result on the analog of Condition EQ for the erasing case. This is observed already when using *equality* as the underlying word relation r. For this purpose, we extend the definition of substitution to map to Σ^* instead of just Σ^+, and we denote the corresponding (erasing) language of a relational pattern (p, R) over word relation r by $L_{*,r}(p, R)$. Likewise the notion of ℓ_z-substitution is extended to that of an ℓ_z^*-substitution. The latter replaces each variable by a word whose length is in $\{0, 1, \ldots, z\}$ rather than just $\{1, \ldots, z\}$. The set of words generated from (p, R) by legal ℓ_z^*-substitutions (wrt r) is denoted $L_{*,r}^{(z)}(p, R)$.

Condition EQ*. $L_{*,r}(p, R) = L_{*,r}(p', R')$ iff $(L_{*,r}^{(z)}(p, R) \subseteq L_{*,r}(p', R')$ *and* $L_{*,r}^{(z)}(p', R') \subseteq L_{*,r}(p, R))$, for all $(p, R), (p, R') \in \Pi$.

Theorem 20 *Let* $|\Sigma| \in \{2, 3, 4\}$. *Then there is no* $z \in \mathbb{N}$ *such that* r_{eq}, z *fulfill Condition EQ*.*

Proof Note that the class of all languages $\{L_{*,r_{eq}}(p, R) \mid (p, R)$ is a relational pattern$\}$ is exactly the class of erasing pattern languages as introduced by Shinohara (1982). For $|\Sigma| \in \{2, 3, 4\}$, Reidenbach (2008) showed that this class does not possess a system of tell-tales.[7] That means, assuming any enumeration $(L_i)_{i\in\mathbb{N}}$ of the class of erasing pattern languages, there is no family $(T_i)_{i\in\mathbb{N}}$ such that, for all $i \in \mathbb{N}$, (i) $T_i \subseteq L_i$, and (ii) there is no $j \in \mathbb{N}$ with $T_i \subseteq L_j \subset L_i$.

Now suppose there were a $z \in \mathbb{N}$ such that r_{eq}, z fulfill Condition EQ*. We claim that mapping each $L_{*,r_{eq}}(p, R)$ to the finite set $T_{(p,R)} = L_{*,r_{eq}}^{(z)}(p, R)$ would then yield a system of tell-tales for the class of erasing pattern languages. Clearly, (i) $T_{(p,R)} \subseteq L_{*,r_{eq}}(p, R)$. To show (ii), suppose there is some (p', R') such that $T_{(p,R)} \subseteq L_{*,r_{eq}}(p', R') \subset L_{*,r_{eq}}(p, R)$. This implies $L_{*,r_{eq}}^{(z)}(p', R') = T_{(p',R')} \subseteq L_{*,r_{eq}}(p, R)$, so that we have both $L_{*,r_{eq}}^{(z)}(p, R) \subseteq L_{*,r_{eq}}(p', R')$ and $L_{*,r_{eq}}^{(z)}(p', R') \subseteq L_{*,r_{eq}}(p, R)$. By Condition EQ*, we obtain $L_{*,r_{eq}}(p, R) = L_{*,r_{eq}}(p', R')$, so that (ii) is fulfilled. Thus, there is a system of tell-tales for the class of erasing pattern languages, in contradiction to Reidenbach's result. Therefore, there is no $z \in \mathbb{N}$ such that r_{eq}, z fulfill Condition EQ*. ■

8. Conclusions

With Condition EQ, we formulated a property that is of relevance for the design of efficient standardized tests for the equivalence of relational pattern languages, for the construction of tell-tale sets in learning in the limit (see Theorem 20), as well as for the design of small teaching sets consisting of simple examples. It was shown that substitutions replacing variables with at most two characters suffice for testing the equivalence of patterns under the *equal-length* relation, for alphabets of size at least 3, while not even three characters are enough under the *reversal* relation when the alphabet is binary. The connection to tell-tales further provided a negative result on erasing pattern languages. We conjecture that the *reversal* relation satisfies Condition EQ with $z = 2$ when the alphabet has at least 5 symbols. That the alphabet size is critical for statements on pattern languages or for the simplicity of their proofs is not uncommon (Bayeh et al., 2020; Reidenbach, 2008; Nessel and Lange, 2005); our results suggest similar issues for relational patterns.

7. The notion of tell-tale was introduced by Angluin (1980b) in the context of characterizing classes of languages that are learnable in Gold's model of identification in the limit from positive data.

Acknowledgments

This research was supported through the Alberta Machine Intelligence Institute (Amii), through the NSERC Discovery Grants program, through the NSERC Canada Research Chairs program, as well as through the Canada CIFAR AI Chairs program. Moreover, the authors thank the anonymous reviewers for comments that helped improve the presentation of this paper.

References

E. Alanazi, M. Mouhoub, and S. Zilles. The complexity of exact learning of acyclic conditional preference networks from swap examples. *Artif. Intell.*, 278, 2020.

D. Angluin. Finding patterns common to a set of strings. *J. Comput. Syst. Sci.*, 21:46–62, 1980a.

D. Angluin. Inductive inference of formal languages from positive data. *Inform. Control*, 45:117–135, 1980b.

D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

S. Arikawa, S. Miyano, A. Shinohara, S. Kuhara, Y. Mukouchi, and T. Shinohara. A machine discovery from amino acid sequences by decision trees over regular patterns. *New Generation Comput.*, 11:361–375, 1993.

P. Barceló, L. Libkin, A. Widjaja Lin, and P.T. Wood. Expressive languages for path queries over graph-structured data. *ACM Trans. Database Syst.*, 37(4):31, 2012.

F. Bayeh, Z. Gao, and S. Zilles. Finitely distinguishable erasing pattern languages. *Theor. Comput. Sci.*, 808:38–73, 2020.

F. Cicalese, S. Filho, E.S. Laber, and M. Molinaro. Teaching with limited information on the learner's behaviour. In *Proc. 37th Intl. Conf. on Machine Learning (ICML)*, pages 2016–2026, 2020.

R. Clifford, A. Wettroth Harrow, A. Popa, and B. Sach. Generalised matching. In *SPIRE*, pages 295–301. Springer, 2009.

S. Dasgupta, D. Hsu, S. Poulis, and X. Zhu. Teaching a black-box learner. In *Proc. 36th Intl. Conf. on Machine Learning (ICML)*, pages 1547–1555, 2019.

C. de la Higuera. Characteristic sets for polynomial grammatical inference. *Mach. Learn.*, 27(2): 125–138, 1997.

D. Freydenberger and D. Reidenbach. Bad news on decision problems for patterns. *Inf. Comput.*, 208:83–96, 2010.

Z. Gao, Z. Mazadi, R. Meloche, H.U. Simon, and S. Zilles. Distinguishing pattern languages with membership examples. *Inf. Comput.*, 256:348–371, 2017.

M. Geilke and S. Zilles. Learning relational patterns. In *ALT*, pages 84–98, 2011.

M. Geilke and S. Zilles. Polynomial-time algorithms for learning typed pattern languages. In *LATA*, pages 277–288, 2012.

E.M. Gold. Language identification in the limit. *Inform. Control*, 10:447–474, 1967.

S.A. Goldman and M.J. Kearns. On the complexity of teaching. *J. Comput. Syst. Sci.*, 50:20–31, 1995.

S.A. Goldman and H.D. Mathias. Teaching a smarter learner. *J. Comput. Syst. Sci.*, 52(2):255–267, 1996.

D.G. Kirkpatrick, H.U. Simon, and S. Zilles. Optimal collusion-free teaching. In *Proceedings of the Intl. Conf. on Algorithmic Learning Theory (ALT)*, pages 506–528, 2019.

T. Koshiba. Typed pattern languages and their learnability. In *EuroCOLT*, pages 367–379. Springer, 1995.

S. Lange and S. Zilles. On the learnability of erasing pattern languages in the query model. In *Proceedings of the 14th International Conference on Algorithmic Learning Theory*, pages 129–143, 2003.

F. Mansouri, Y. Chen, A. Vartanian, J. Zhu, and A. Singla. Preference-based batch and sequential teaching: Towards a unified view of models. In *Proc. 32nd Conf. on Neural Information Processing Systems (NeurIPS)*, pages 9199–9209, 2019.

J. Nessel and S. Lange. Learning erasing pattern languages with queries. *Theor. Comput. Sci.*, 348 (1):41–57, 2005.

R.P. Nix. Editing by example. *ACM Trans. Program. Lang. Syst.*, 7:600–621, 1985.

D. Reidenbach. Discontinuities in pattern inference. *Theor. Comput. Sci.*, 397:166–193, 2008.

A. Shinohara and S. Miyano. Teachability in computational learning. *New Generation Comput.*, 8: 337–347, 1991.

T. Shinohara. Polynomial time inference of extended regular pattern languages. In *RIMS Symposium on Software Science and Engineering*, pages 115–127, 1982.

K. Wright. Inductive identification of pattern languages with restricted substitutions. In *COLT*, pages 111–121, 1990.

X. Zhu, A. Singla, S. Zilles, and A.N. Rafferty. An overview of machine teaching. *ArXiv*, abs/1801.05927, 2018.

Appendix A. Proof of Lemma 12

Lemma 12 *Let $(p, R) \in \Pi$, where $p = w_0\mathbb{B}_1w_1\mathbb{B}_2w_2 \dots \mathbb{B}_bw_b$, for some $w_0, w_b \in \Sigma^*$, $w_k \in \Sigma^+$ ($k \in \{1, \dots, b-1\}$), and for some blocks $\mathbb{B}_k$ of variables. Then*

$$L_{r_{len}}(p, R) = \bigcup_{\langle m_1, m_2, \dots m_b\rangle \in LT(p,R)} \{w_0\Sigma^{m_1}w_1\Sigma^{m_2}w_2 \dots \Sigma^{m_b}w_b\}\,.$$

Proof We first show that, if $\langle m_1, m_2, \ldots m_b\rangle \in LT(p,R)$, then $\{w_0\Sigma^{m_1}w_1\Sigma^{m_2}w_2\ldots\Sigma^{m_b}w_b\} \subseteq L_{r_{len}}(p,R)$.

To this end, let w be any word in $\{w_0\Sigma^{m_1}w_1\Sigma^{m_2}w_2\ldots\Sigma^{m_b}w_b\}$. We need to show $w \in L_{r_{len}}(p,R)$, i.e. that there is a legal substitution θ of (p,R) such that $\theta(p) = w$.

Let n be the number of variable groups induced by R. Consider any block of variables $\mathbb{B}_k = x_{k,1}x_{k,2}\ldots x_{k,|\mathbb{B}_k|}$ and let $\langle a_{k,1},\ldots,a_{k,n}\rangle$ be R's decomposition of $\mathbb{B}_k$. In particular, R stipulates for each legal substitution θ under r_{len}, that $|\theta(x_{k,1})| = |\theta(x_{k,2})| = \ldots|\theta(x_{k,a_{k,1}})|$, that $|\theta(x_{k,a_{k,1}+1})| = |\theta(x_{k,a_{k,1}+2})| = \ldots|\theta(x_{k,a_{k,1}+a_{k,2}})|$, and so on. (Note that we here implicitly use Lemma 8 in order to shuffle the variables in a block into homogeneous groups without affecting the resulting language.)

Because $\langle m_1, m_2, \ldots m_b\rangle \in LT(p,R)$, there exist $\gamma_i \geq 1$ ($i \in \{1\ldots n\}$) such that, for all $k \in \{1,\ldots,b\}$, $m_k = \sum_{i=1}^{n}\gamma_i a_{k,i}$. Any substitution θ that, for all $k \in \{1,\ldots,b\}$, substitutes a word of length γ_1 for variables $x_{k,1},\ldots x_{k,a_{k,1}}$, a word of length γ_2 for variables $x_{k,a_{k,1}+1},\ldots x_{k,a_{k,1}+a_{k,2}}$, etc., will satisfy the constraints in R and have the length tuple $\langle m_1, m_2, \ldots m_b\rangle$.

All that remains to be shown is that there exists such a θ that produces w when applied to p. But this is obvious because there are no constraints on the θ other than the length constraints just described.

Because w matches length tuple $\langle m_1, m_2, \ldots m_b\rangle$ the section of w that corresponds to $\mathbb{B}_k = x_{k,1}x_{k,2}\ldots x_{k,|\mathbb{B}_k|}$ begins at position $\alpha_k = (\sum_{j=0}^{k-1}|w_j|) + (\sum_{j=1}^{k-1} m_j) + 1$. The following mapping θ therefore satisfies the constraints in R and will generate w:

- $\theta(x_{k,1}) = w_{\alpha_k}, w_{\alpha_k+1},\ldots,w_{\alpha_k+\gamma_1-1}$
- $\theta(x_{k,2}) = w_{\alpha_k+\gamma_1}, w_{\alpha_k+\gamma_1+1},\ldots,w_{\alpha_k+2\gamma_1-1}$
- $\ldots$
- $\theta(x_{k,a_{k,1}}) = w_{\alpha_k+(a_{k,1}-1)\gamma_1}, w_{\alpha_k+(a_{k,1}-1)\gamma_1+1},\ldots,w_{\alpha_k+a_{k,1}\gamma_1-1}$
- $\theta(x_{k,a_{k,1}+1}) = w_{\alpha_k+a_{k,1}\gamma_1}, w_{\alpha_k+a_{k,1}\gamma_1+1},\ldots,w_{\alpha_k+a_{k,1}\gamma_1+\gamma_2-1}$

The opposite inclusion ($L_{r_{len}}(p,R) \subseteq \bigcup_{\langle m_1,m_2,\ldots m_b\rangle\in LT(p,R)}\{w_0\Sigma^{m_1}w_1\Sigma^{m_2}w_2\ldots\Sigma^{m_b}w_b\}$) follows from the fact that $w \in L_{r_{len}}(p,Q)$ implies that w matches at least one length tuple in $LT(p,Q)$. ■

Appendix B. Proof of Theorem 18

Theorem 18 *Suppose $|\Sigma| = 2$. Then there exists a pattern p and relations R, R' such that $L^{(3)}_{r_{rev}}(p,R) \subseteq L_{r_{rev}}(p,R')$ and $L^{(3)}_{r_{rev}}(p,R') \subseteq L_{r_{rev}}(p,R)$, yet $L_{r_{rev}}(p,R) \neq L_{r_{rev}}(p,R')$.*

Proof Let $\Sigma = \{a,b\}$ and let

$$p = v_1\, aa\, x\, aa\, v_2\, aa\, y\, a\, v_3 z v_4\,,$$

where $x,y,z,v_i \in X$, $a \in \Sigma$, $R = \{(x,z),(z,y)\}$, and $R' = \{(x,y),(z,y)\}$. Note that legal substitutions for (p,R) replace x and y with the same word and z with its reversal, while legal substitutions for (p,R') replace x and z with the same word and y with its reversal.

It remains to verify that (i) $L^{(3)}_{r_{rev}}(p, R) \subseteq L_{r_{rev}}(p, R')$, and (ii) $L^{(3)}_{r_{rev}}(p, R') \subseteq L_{r_{rev}}(p, R)$, while (iii) $L_{r_{rev}}(p, R) \neq L_{r_{rev}}(p, R')$.

We begin by showing part (i). Any legal ℓ_3-substitution θ for (p, R) that replaces x with a palindrome is clearly also legal for (p, R'), so the only non-trivial cases are $\theta(x) = abb$, $\theta(x) = aab$, $\theta(x) = baa$, $\theta(x) = bba$, $\theta(x) = ab$, and $\theta(x) = ba$.

When $\theta(x) = abb$, $\theta(p, R) = w = w_1aa\boxed{abb}aaw_2aa\boxed{abb}aw_3\boxed{bba}w_4$, where $\theta(v_i) = w_i$, and the boxes indicate the locations of $\theta(x)$, $\theta(y)$, and $\theta(z)$ in w. A substitution θ' that is legal for (p, R') and produces w is the following: $\theta'(v_1) = w_1a$, $\theta'(x) = bb$, $\theta'(v_2) = w_2a$, $\theta'(y) = bb$, $\theta'(v_3) = w_3$, $\theta'(z) = bb$, and $\theta'(v_4) = aw_4$.

When $\theta(x) = aab$, $\theta(p, R) = w = w_1aa\boxed{aab}aaw_2aa\boxed{aab}aw_3\boxed{baa}w_4$, where $\theta(v_i) = w_i$. A substitution θ' that is legal for (p, R') and produces w is the following: $\theta'(v_1) = w_1aa$, $\theta'(x) = b$, $\theta'(v_2) = w_2aa$, $\theta'(y) = b$, $\theta'(v_3) = w_3$, $\theta'(z) = b$, and $\theta'(v_4) = aaw_4$.

When $\theta(x) = baa$, $\theta(p, R) = w = w_1aa\boxed{baa}aaw_2aa\boxed{baa}aw_3\boxed{aab}w_4$, where $\theta(v_i) = w_i$. A substitution θ' that is legal for (p, R') and produces w is the following: $\theta'(v_1) = w_1$, $\theta'(x) = b$, $\theta'(v_2) = aaw_2$, $\theta'(y) = b$, $\theta'(v_3) = aaw_3aa$, $\theta'(z) = b$, and $\theta'(v_4) = w_4$.

When $\theta(x) = bba$, $\theta(p, R) = w = w_1aa\boxed{bba}aaw_2aa\boxed{bba}aw_3\boxed{abb}w_4$, where $\theta(v_i) = w_i$. A substitution θ' that is legal for (p, R') and produces w is the following: $\theta'(v_1) = w_1$, $\theta'(x) = bb$, $\theta'(v_2) = aw_2$, $\theta'(y) = bb$, $\theta'(v_3) = aw_3a$, $\theta'(z) = bb$, and $\theta'(v_4) = w_4$.

When $\theta(x) = ab$, $\theta(p, R) = w = w_1aa\boxed{ab}aaw_2aa\boxed{ab}aw_3\boxed{ba}w_4$, where $\theta(v_i) = w_i$. A substitution θ' that is legal for (p, R') and produces w is the following: $\theta'(v_1) = w_1a$, $\theta'(x) = b$, $\theta'(v_2) = w_2a$, $\theta'(y) = b$, $\theta'(v_3) = w_3$, $\theta'(z) = b$, and $\theta'(v_4) = aw_4$.

When $\theta(x) = ba$, $\theta(p, R) = w = w_1aa\boxed{ba}aaw_2aa\boxed{ba}aw_3\boxed{ab}w_4$, where $\theta(v_i) = w_i$. A substitution θ' that is legal for (p, R') and produces w is the following: $\theta'(v_1) = w_1$, $\theta'(x) = b$, $\theta'(v_2) = aw_2$, $\theta'(y) = b$, $\theta'(v_3) = aw_3a$, $\theta'(z) = b$, and $\theta'(v_4) = w_4$.

Part (ii) is verified in a similar way. Any legal ℓ_3-substitution θ' for (p, R') that replaces x with a palindrome is clearly also legal for (p, R), so the only non-trivial cases are $\theta'(x) = abb$, $\theta'(x) = aab$, $\theta'(x) = baa$, $\theta'(x) = bba$, $\theta'(x) = ab$, and $\theta'(x) = ba$.

When $\theta'(x) = abb$, $\theta'(p, R') = w = w_1aa\boxed{abb}aaw_2aa\boxed{bba}aw_3\boxed{abb}w_4$, where $\theta'(v_i) = w_i$, and the boxes indicate the locations of $\theta'(x)$, $\theta'(y)$, and $\theta'(z)$ in w. A substitution θ that is legal for (p, R) and produces w is the following: $\theta(v_1) = w_1a$, $\theta(x) = bb$, $\theta(v_2) = w_2$, $\theta(y) = bb$, $\theta(v_3) = aw_3a$, $\theta(z) = bb$, and $\theta(v_4) = w_4$.

When $\theta'(x) = aab$, $\theta'(p, R') = w = w_1aa\boxed{aab}aaw_2aa\boxed{baa}aw_3\boxed{aab}w_4$, where $\theta'(v_i) = w_i$. A substitution θ that is legal for (p, R) and produces w is the following: $\theta(v_1) = w_1aa$, $\theta(x) = b$, $\theta(v_2) = w_2$, $\theta(y) = b$, $\theta(v_3) = aaw_3aa$, $\theta(z) = b$, and $\theta(v_4) = w_4$.

When $\theta'(x) = baa$, $\theta'(p, R') = w = w_1aa\boxed{baa}aaw_2aa\boxed{aab}aw_3\boxed{baa}w_4$, where $\theta'(v_i) = w_i$. A substitution θ that is legal for (p, R) and produces w is the following: $\theta(v_1) = w_1$, $\theta(x) = b$, $\theta(v_2) = aaw_2aa$, $\theta(y) = b$, $\theta(v_3) = w_3$, $\theta(z) = b$, and $\theta(v_4) = aaw_4$.

When $\theta'(x) = bba$, $\theta'(p, R') = w = w_1aa\boxed{bba}aaw_2aa\boxed{abb}aw_3\boxed{bba}w_4$, where $\theta'(v_i) = w_i$. A substitution θ that is legal for (p, R) and produces w is the following: $\theta(v_1) = w_1$, $\theta(x) = bb$, $\theta(v_2) = aw_2a$, $\theta(y) = bb$, $\theta(v_3) = w_3$, $\theta(z) = bb$, and $\theta(v_4) = aw_4$.

When $\theta'(x) = ab$, $\theta'(p, R') = w = w_1aa\boxed{ab}aaw_2aa\boxed{ba}aw_3\boxed{ab}w_4$, where $\theta'(v_i) = w_i$. A substitution θ that is legal for (p, R) and produces w is the following: $\theta(v_1) = w_1a$, $\theta(x) = b$, $\theta(v_2) = w_2$, $\theta(y) = b$, $\theta(v_3) = aw_3a$, $\theta(z) = b$, and $\theta(v_4) = w_4$.

When $\theta'(x) = ba$, $\theta'(p, R') = w = w_1aa\boxed{ba}aaw_2aa\boxed{ab}aw_3\boxed{ba}w_4$, where $\theta'(v_i) = w_i$. A substitution θ that is legal for (p, R) and produces w is the following: $\theta(v_1) = w_1$, $\theta(x) = b$, $\theta(v_2) = aw_2a$, $\theta(y) = b$, $\theta(v_3) = w_3$, $\theta(z) = b$, and $\theta(v_4) = aw_4$.

Part (iii) is witnessed by the word $w = baa\boxed{babb}aabaa\boxed{babb}ab\boxed{bbab}b \in L_{r_{rev}}(p, R)$, created by $\theta(v_i) = b$, $\theta(x) = \theta(y) = babb$, and $\theta(z) = bbab$. To see that there does not exist a θ' such that $\theta'(p) = w$ and θ' is legal for (p, R'), note that both p and w contain exactly three occurrences of the substring aa. Therefore, any substitution generating w from p must replace x with $babb$, but then cannot replace y with the reversal $bbab$. ■

Appendix C. Proof of Proposition 19

Proposition 19 *Let (p, R) and (p, R') be two relational patterns. Let r_{rev} be the underlying word relation. Then $(L^{(1)}_{r_{rev}}(p, R) \subseteq L_{r_{rev}}(p', R')$ and $L^{(1)}_{r_{rev}}(p', R') \subseteq L_{r_{rev}}(p, R))$ implies that every group of variables in (p, R) is a group of variables in (p, R') and vice versa.*

Proof By contradiction. Wlog, suppose x, y are variables in p such that x and y are in the same group in R but not in R'. Then there is a legal ℓ_1-substitution θ for (p, R') that replaces x with a and y with b, for some $a, b \in \Sigma$, $a \neq b$. Because the length of $w = \theta(p)$ is $|p|$, θ is the only substitution of variables in p that generates w from p. Since θ is not legal for (p, R), we obtain that $w \in L^{(1)}_{r_{rev}}(p, R') \setminus L_{r_{rev}}(p, R)$. ■