Minimization of Tree Pattern Queries

Wojciech Czerwiński*
University of Warsaw

Wim Martens
Universität Bayreuth

Matthias Niewerth†
Universität Bayreuth

Paweł Parys*
University of Warsaw

ABSTRACT

We prove that minimization for tree patterns is Σ_2^P-complete and thus solve a problem first attacked by Flesca, Furfaro, and Masciari in 2003. We first provide an example that shows that tree patterns cannot be minimized by deleting nodes. This example shows that the M-NR conjecture, which states that minimality of tree patterns is equivalent to their nonredundancy, is false. We then show how the example can be turned into a gadget that allows us to prove Σ_2^P-completeness.

Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: General; H.2.3 [**Database Management**]: Languages—*query languages*

Keywords

XPath, XML, trees, tree patterns, graph databases, optimization, complexity

1. INTRODUCTION

Tree pattern queries are a fundamental tool for node-selection and querying in tree- and graph-structured data. They are present in most query languages for tree-structured data, most notably, XPath [31]. In fundamental research they appear in a wide range of topics. For example, they form a basis for conjunctive queries over trees [20], for models of XML with incomplete information [5], and for the closely related pattern-based XML queries [18]. They are used for specifying guards in Active XML systems [1] and for specifying schema mappings in XML data exchange [4].

Beyond trees, they are a natural language for querying graph databases [25, 14].

Optimization of tree pattern queries is therefore a very natural question. Not only do new results in this direction give us new insights on query optimization in many practical languages, they can also give us a better understanding of the foundations of the above mentioned models which are based on tree patterns. Tree pattern query optimization already attracted significant attention in the form of *query containment* [27, 29, 14], *satisfiability* [6], and *minimization* [3, 13, 16, 23, 30, 33].

In this paper we study the minimization problem for tree patterns that use the *child relation*, *descendant relation*, *label tests* and *wildcards* (henceforth: tree patterns). These tree patterns are widely studied [15, 27, 34, 16, 23, 1, 8, 4, 32] but their minimization problem remained elusive. In particular, the complexity of minimization is unknown.

It is believed that a key in understanding tree pattern minimization lies in understanding the relationship between *minimality* (M) and *nonredundancy* (NR) [16, 23]. Here, a tree pattern is minimal if it has the smallest number of nodes among all equivalent tree patterns. It is nonredundant if none of its leaves or branches can be deleted while remaining equivalent. The question is if minimality and nonredundancy are the same:

> M-NR Conjecture (page 35 of [16], rephrased): A tree pattern is minimal if and only if it is nonredundant.

Clearly, every minimal tree pattern is nonredundant, so one direction of the M-NR conjecture trivially holds. The opposite direction is much less clear. If it would be true, it means that, for a given tree pattern p, a minimal tree pattern is always a substructure of p. It would also mean that tree pattern minimization can be solved by tree pattern containment. Indeed, one would be able to minimize tree patterns p by iteratively removing leaves and testing if the obtained tree pattern p' is still contained in p. If no leaf can be removed anymore, the remaining tree pattern would be nonredundant and therefore minimal. Since testing containment of tree patterns is coNP-complete [27], this would mean that one could solve minimization with a polynomial-time algorithm with a coNP oracle and that the minimization problem for tree patterns is coNP-complete.

The minimization problem for tree patterns is not entirely uncontroversial. It was claimed to be coNP-complete in 2003 [15] but the algorithm relied on the M-NR conjecture. Kimelfeld and Sagiv [23] proved that, in contrast to claims in [15], the M-NR conjecture is open and, as a consequence, the algorithms from [15]

*Supported by Poland's National Science Centre grant no. UMO-2013/11/D/ST6/03075.

†Supported by grant number MA 4938/2–1 from the Deutsche Forschungsgemeinschaft (Emmy Noether Nachwuchsgruppe).

were revised in [16]. The updated work [16] proves that the M-NR conjecture holds for tree patterns in which *every wildcard node has at most one child* and presents a coNP algorithm for this case. Although [16, 23] contain a wealth of valuable results on tree patterns and their minimization (many of which we use here), the most central questions, that is, the status of the M-NR conjecture and the question of the complexity of tree pattern minimization remained open.

Our main contributions are the following:

- We prove that the M-NR conjecture is false by providing a tree pattern that is nonredundant but not minimal.
- We prove that tree pattern minimization is Σ_2^P-complete. This means that, unless $\Sigma_2^P = \text{coNP}$, we have that tree pattern minimization *cannot* be solved by a polynomial-time algorithm with an oracle for tree pattern containment.
- Interestingly, our counterexample and our gadgets in the Σ_2^P-hardness proof use only two wildcard nodes with two children, which is only barely beyond the fragment for which the M-NR conjecture is known to hold.

The Bigger Picture.

This paper fits naturally in a line of research that originated in the early days of database theory. Query minimization and optimization by removing redundant parts goes back to the seminal work of Chandra and Merlin [11] and has since then been successfully adopted for many types of queries, in various data models such as relations and trees (see, e.g., [3, 10, 11, 16, 30, 33]). In this section we highlight a few parallels and differences between conjunctive queries and acyclic conjunctive queries over relational data and over tree-structured data.

Conjunctive Queries over Relations. Minimization, containment, and evaluation of conjunctive queries over relations are well known to be NP-complete [11]. For *acyclic* conjunctive queries, which have been extensively studied (see, e.g., [12, 21, 35]), the complexity of these problems is in polynomial time.

Conjunctive Queries over Trees. Conjunctive queries over trees [20] are different from conjunctive queries over relations in two respects. First, the underlying model is based on trees instead of relations and second, conjunctive queries over trees use different built-in relations. Most notably, apart from the child relation, they can use the descendant relation and therefore have the power to reason about certain transitive closures. They therefore query a more restricted data model than their counterpart over relations but in return they have more powerful reasoning. The original definition of conjunctive queries over trees allows for many built-in relations (child, descendant, nextsibling, following-sibling, etc. [20]); we only discuss the two most basic ones, child and descendant, in the following.

Conjunctive queries over trees have an NP-complete evaluation problem [20] just like conjunctive queries over relations. Their containment problem, however, is Π_2^P-complete [7] and the complexity of their minimization problem is unknown.

Tree patterns can be seen as acyclic variants of conjunctive queries over trees. Their evaluation problem is in polynomial time [19] and their containment problem is coNP-complete [27].

In this paper, we aim at completing this picture by showing that minimization of tree patterns is Σ_2^P-complete. We are currently working on adapting our proof for conjunctive queries over trees and think that it can be used to prove that their minimization problem is Σ_3^P-complete.

Tree Patterns as a Graph Query Language.

Due to their modal nature, tree patterns and XPath-like languages are also a suitable language for querying graph databases [9, 22, 25, 2, 26, 17]. In fact, the complexity of tree pattern containment does not depend on whether they are evaluated over trees or over graphs, see [27, Section 5.3] and [14, Section 7]. The same is true for the minimization problem. Therefore, the complexity results in this paper can be extended to tree patterns over graphs as well. We present all results in terms of trees because it makes proofs considerably simpler.

The problems for trees can even be extended to *data graphs* [25] and patterns that compare data values with constants (such comparisons are essentially the same as the label tests of tree patterns). However, as soon as data value *comparisons* enter the picture, such a straightforward extension to graphs does not work anymore, see, e.g. [24].

Outline.

We present the counterexample to the M-NR conjecture in Section 3. We also show in Section 3 that minimal tree patterns are not unique (up to adornments). In Section 4 we prove that tree pattern minimization is Σ_2^P-complete. We discuss implications on k-ary queries and further outlooks in Section 5. We postpone a very technical proof (Lemma 4.2) to the Appendix.

2. PRELIMINARIES

We are interested in finite, labeled, unordered trees.[1] A *labeled unordered tree* is a triple (V, E, lab), where V is a finite set of *nodes*, E is a set of *edges* $(u, v) \in V \times V$ and $\text{lab} : V \to \Sigma$ is a *labeling function* assigning to every node its label coming from an infinite set of labels Σ. If $(u, v) \in E$ then we say that u is the *parent* of v and v is a *child* of u. For every node v there is at most one $(u, v) \in E$, so the parent is uniquely determined. There is a unique node without parent, which we denote $\text{root}(t)$ and call the *root* of t. The *descendant* and *ancestor* relations are transitive closures of the child and parent

[1]However, we don't require trees to be unordered. Our results are the same for ordered trees.

relations, respectively. We say that a child of a node u is a 1-*descendant* of u and a child of a k-*descendant* of u is a $(k+1)$-*descendant* of u for any $k \in \mathbb{N}$. We define k-*ancestors* similarly. A node has *depth* k if it is k-descendant of the root. In the sequel we just use the term *trees* for referring to labeled unordered trees.

For a tree $t = (V, E, \text{lab})$ and a node $v \in V$ we denote by t^v the subtree of t rooted in node v. By $t \setminus v$ we denote the tree obtained from t by deleting the subtree rooted at v (including node v itself).

A *tree pattern* is intended to describe a set of trees. It is a special type a of tree; its set of edges is divided into two disjoint sets: *child edges* and *descendant edges* (we draw descendant edges using double lines). Its labeling function provides every node with a label from Σ or a special label $*$ which we assume not to be in Σ and call *wildcard*. We denote $\Sigma_* = \Sigma \cup \{*\}$. The intended meaning of the wildcard is not to specify any particular label.

For a tree pattern $p = (V_p, E_p, \text{lab}_p)$ and a tree $t = (V, E, \text{lab})$, a function $\pi : V_p \to V$ is a *strong embedding*[2] of p in t if it fulfils all the following conditions:

(1) if $\text{lab}_p(v) \neq *$ for $v \in V_p$ then $\text{lab}_p(v) = \text{lab}(\pi(v))$,

(2) if $(u, v) \in E_p$ is a child edge then $\pi(u)$ is a parent of $\pi(v)$ in the tree t,

(3) if $(u, v) \in E_p$ is a descendant edge then $\pi(u)$ is an ancestor of $\pi(v)$ in the tree t, and

(4) $\pi(\text{root}(p)) = \text{root}(t)$.

We say that p *strongly embeds* in t if there exists a strong embedding of p in t. We say that π is a *weak embedding* of p in t and p *weakly embeds* in t if the above conditions (1)–(3) are fulfilled, but not necessarily $\pi(\text{root}(p)) = \text{root}(t)$. Figure 1 contains examples of strong and weak embeddings. Notice that we do not require embeddings to be injective (Figure 1(c)).

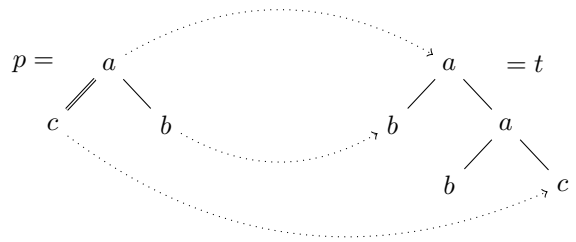
Equivalence, Containment, and Minimality.

The *(strong) language of a tree pattern* p, denoted by $L_S(p)$, is the set of trees in which p strongly embeds. A tree pattern p_1 is *(strongly) contained* in a tree pattern p_2 if $L_S(p_1) \subseteq L_S(p_2)$, which we denote by $p_1 \subseteq_S p_2$. If $p_1 \subseteq_S p_2$ and $p_1 \supseteq_S p_2$ then we say that the tree patterns p_1 and p_2 are *(strongly) equivalent* and we write $p_1 \equiv_S p_2$.

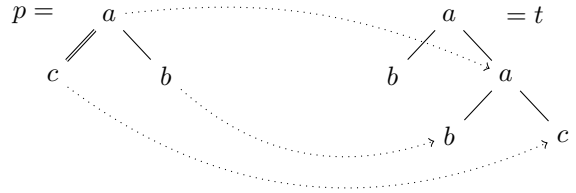
We call a tree pattern p *redundant* if one of its nodes can be removed without changing its language. More formally, p is redundant if it is strongly equivalent to $p \setminus v$ for a node v of p. In this case, v is a *redundant node*. If p is not redundant we say that it is *nonredundant*. It is known that a pattern is redundant if and only if it has a redundant leaf [23, Proposition 3.3].

The *size* of a tree pattern p, denoted $\text{size}(p)$, is the number of its nodes. A tree pattern p is said to be *minimal* if there is no tree pattern p' that is equivalent to p but has strictly smaller size.

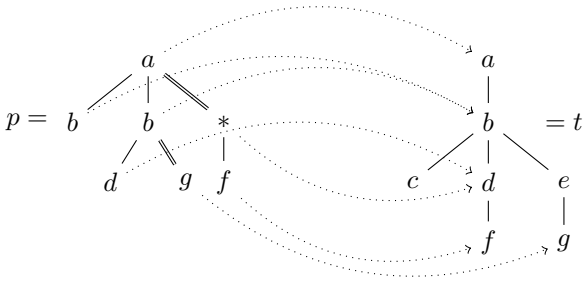
[2]A strong embedding is sometimes also called *root embedding*.



(a) A strong embedding of a tree pattern p in a tree t.



(b) A weak embedding of a tree pattern p in a tree t.



(c) A strong embedding of a tree pattern p in tree t. Embeddings are not required to be injective.

Figure 1: Examples of strong and weak embeddings.

Analogously, we define *weak language*, *weak containment*, *weak equivalence*, *weak redundancy*, and *weak nonredundancy*. The definitions are exactly the same, but use weak embeddings instead of strong embeddings. The notation for weak containment and weak equivalence is $p_1 \subseteq_W p_2$ and $p_1 \equiv_W p_2$, respectively.

It is well-known that containment of tree patterns, i.e., deciding for two given tree patterns p and q if $p \subseteq_S q$, is coNP-complete.

THEOREM 2.1 ([27]). *Containment of tree patterns is* coNP*-complete.*

We now mention a weak version of the M-NR Conjecture which was proved in [16]. We require the following definition.

DEFINITION 2.2 ($*$-NARROW PATTERN). *A tree pattern is a* $*$-narrow pattern *if every wildcard node has at most one child.*

For example, the tree patterns in Figure 1 are $*$-narrow patterns.

LEMMA 2.3 (COROLLARY 4.5 IN [16]). *If p is a $*$-narrow pattern, then p is minimal if and only if p is non-redundant.*

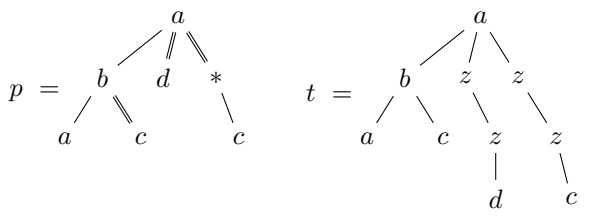


Figure 2: A tree pattern p and a canonical tree t.

Canonical trees.

Canonical trees were introduced by Miklau and Suciu [27] for studying the containment problem for tree patterns. We need them in our paper to simplify proofs.

Let $z \in \Sigma$ be a special label that does not occur in any tree pattern that we consider in the paper. (We can assume that such a label exists because Σ is infinite.) A *canonical tree* of a tree pattern p is a tree obtained from p by application of the two following steps:

- for every node v such that $\text{lab}_p(v) = *$, we relabel $\text{lab}(v) = z$,
- change every descendant edge in p to a sequence of edges in t in such a way that all newly created nodes are labeled by z.

An example is given in Figure 2. We denote by $\text{Can}(p)$ the set of all canonical trees of p.

Given a pattern p and one of its canonical trees t, there is a canonical injective embedding of the non-wildcard nodes of p into t. We sometimes use this correspondence to talk about nodes in t. That is, for a non-wildcard node u of p, we use this injective embedding to identify *the node in t corresponding to u*.

LEMMA 2.4 (PROPOSITION 3 IN [27]). *Let p_1 and p_2 be two tree patterns. Then $p_1 \subseteq_S p_2$ if and only if $Can(p_1) \subseteq L_S(p_2)$.*

A corresponding lemma for weak containment can be proved analogously. (See Lemma B.1 in Appendix B.)

Homomorphisms.

Let $p_1 = (V_{p_1}, E_{p_1}, \text{lab}_{p_1})$ and $p_2 = (V_{p_2}, E_{p_2}, \text{lab}_{p_2})$ be tree patterns. A *homomorphism* from p_1 to p_2 is a function $h : V_{p_1} \to V_{p_2}$ that fulfils the following conditions:

(1) $h(\text{root}(p_1)) = \text{root}(p_2)$,

(2) if $\text{lab}_{p_1}(v) \neq *$ for $v \in V_{p_1}$ then $\text{lab}_{p_1}(v) = \text{lab}_{p_2}(h(v))$,

(3) if $(u, v) \in E_{p_1}$ is a child edge then $(h(u), h(v)) \in E_{p_2}$ is a child edge, and

(4) if $(u, v) \in E_{p_1}$ is a descendant edge then $h(u)$ is an ancestor of $h(v)$ in p_2.

The existence of a homomorphism h from p_1 to p_2 is a sufficient for $p_2 \subseteq_S p_1$ [27]. Essentially, the reason is that, if π is a strong embedding of p_2 in a tree t, then $\pi \circ h$ is a strong embedding of p_1 in t. We make use of this fact later in the paper.

3. NONREDUNDANCY AND MINIMALITY

In this section we present a counterexample for the M-NR conjecture. We build further on this example to show that minimal tree patterns are not unique. We choose the examples in such a way that they help the reader to understand the gadgets we use in Section 4.

Nonredundancy ≠ Minimality.

We will show that the left tree pattern p in Figure 3 is nonredundant and not minimal. One thing is easy to see: the tree pattern q on the right is smaller. Seeing that the left tree pattern p is nonredundant and equivalent to the tree pattern q on the right requires more work.

First we show that the tree pattern p is non-redundant. To this end, it suffices to show that none of its leaves can be deleted while remaining equivalent [23, Proposition 3.3]. For the purpose of this proof, we order the leaves in p from left to right, that is, the *first c_1-leaf* is the one in depth 7, the *second c_1-leaf* is the leftmost leaf on depth 8, and so on.

- If the first c_1-leaf is removed, then the resulting tree pattern matches the tree in Figure 4(a) by the strong embedding π which we partly illustrated in Figure 4(a). However, the tree pattern p does not match. The reason why we cannot remove the first c_2-leaf of p is analogous (replace the first c_2-leaf in the tree in Figure 4(a) by a c_1-leaf).
- If the second c_1-leaf is removed, then the resulting tree pattern matches the tree in Figure 4(b) using the strong embedding π which we partly illustrated in Figure 4(b). However, the tree pattern p does not match. The reason why we cannot remove the first c_2-leaf of p is analogous (replace the first c_2-leaf in the tree in Figure 4(b) by a c_1-leaf).
- Finally, if any of the other c_1- or c_2-leaves would be removed, the tree pattern would match the tree in Figure 4(c) in which the corresponding (circled) leaf would be removed and this is always a tree that is not matched by p.

Finally, we show that the tree pattern p is (strongly) equivalent to the tree pattern q. To this end, observe that $q \subseteq_S p$ because q is more restrictive: it has the same requirements as p but, in addition it says that the nodes onto which the second c_1- and c_2-leaves are matched have the same parent. It therefore only remains to show that $p \subseteq_S q$. By Lemma 2.4, it suffices to prove that $\text{Can}(p) \subseteq_S L_S(q)$.

Let $t \in \text{Can}(p)$. Let π_p be a strong embedding of p in t. Consider the nodes u_A and u_B in p. We make a case distinction on the number of nodes between $\pi_p(u_A)$ and $\pi_p(u_B)$ in t and show in each case how to construct a strong embedding π_q from q in t.

- If $\pi_p(u_A)$ is the parent of $\pi_p(u_B)$, then we can define $\pi_q(v_2) := \pi_p(u_4)$. For all non-descendants of v_2, we define π_q the same as π_p.

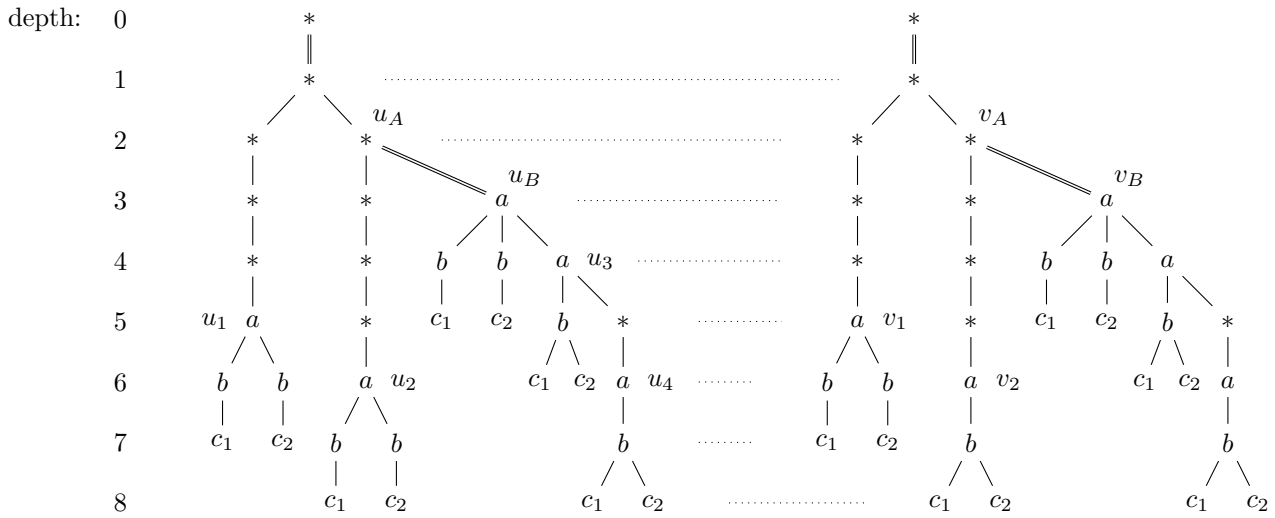


Figure 3: A non-redundant tree pattern p (left) and an equivalent tree pattern q that is smaller (right).

- If $\pi_p(u_A)$ is the 2-ancestor of $\pi_p(u_B)$ (see Figure 5(a)), then we can define $\pi_q(v_1) := \pi_q(u_2)$, and $\pi_q(v_2) := \pi_p(u_4)$, and $\pi_q(v_B) := \pi_p(u_B)$, and, since the distance between $\pi_q(v_2)$ and $\pi_q(v_A)$ is four, we map v_A to the parent of $\pi_p(u_B)$. In particular, the leftmost branch of t is not used by π_q at all.
- If $\pi_p(u_A)$ is the 3-ancestor of $\pi_p(u_B)$ (see Figure 5(b)), then we can define $\pi_q(v_1) := \pi_p(u_B)$ and $\pi_q(v_2) := \pi_p(u_3)$. In particular, the two leftmost branches of t are not used by π_q at all.
- If $\pi_p(u_A)$ is the k-ancestor of $\pi_b(u_B)$ for some $k \geq 4$, we proceed in the same way than in the previous case. Here, we move $\pi_q(v_A)$ downward on the path to $\pi_p(u_B)$ so that the distance between $\pi_q(v_2)$ and $\pi_q(v_A)$ is four.

This means that every canonical tree of p can be (strongly) matched by q. By Lemma 2.4 this means that p and q are strongly equivalent.

Putting everything together, we therefore obtained that p is non-redundant but q is equivalent and smaller, which leads to the following theorem:

THEOREM 3.1. *The M-NR conjecture is false.*

Minimal Patterns are not Unique up to Adornment.

It is well known that minimal tree patterns are not unique [27]. The underlying reason is very simple: the tree patterns in Figure 7(a) are equivalent and minimal. It is for this reason that the existence of a homomorphism between tree patterns fails to be necessary for containment [28]. To circumvent this problem, Miklau and Suciu introduced *adorned patterns* [27, Section 3.2], which allowed them to obtain a sound, polynomial time algorithm for containment that is complete in many cases. Adornment essentially rewrites paths that do not branch and consist of wildcard nodes, child edges and at least one descendant edge into a normal form. The normal form consists of considering each such path and replacing each child edge in the path with a descendant edge. Notice that any given tree pattern is equivalent to its adorned version. We do not formally define adornment here but we refer to Figure 7(b) for a simple example and to [27, Section 3.2] for the details.

Here we show that there exist equivalent but "structurally different" minimal tree patterns or, more formally, equivalent minimal tree patterns that have different adorned patterns. Consider the gadget $P(X, Y, Z)$ in Figure 8. For tree patterns p, q, and r, denote by $P(p, q, r)$ the tree pattern obtained from P by instantiating the subtrees marked X, Y, and Z by p, q, and r, respectively. Consider the tree patterns p, q_1, q_2, and r from Figure 6. Then, we claim that

$$P(p, q_1, r) \text{ and } P(p, q_2, r)$$

are equivalent and minimal, but their adorned versions are different. It is easy to see that the adorned versions of these tree patterns are different, but their equivalence and minimality are non-trivial.

We first show that the tree patterns are equivalent. To this end, we first prove a lemma that already gives insight to a central property of the gadget in Figure 8.

LEMMA 3.2. *Let α, β_1, β_2, and γ be tree patterns that do not use labels in $\{a, b\}$ and such that $\alpha \subseteq_S \beta_1 \subseteq_S \gamma$ and $\alpha \subseteq_S \beta_2 \subseteq_S \gamma$. Then $P(\alpha, \beta_1, \gamma) \equiv_S P(\alpha, \beta_2, \gamma)$.*

PROOF SKETCH. The proof follows the same lines as the proof showing that the two tree patterns in Figure 3 are equivalent. Given a tree t and embedding π_1 of $P(\alpha, \beta_1, \gamma)$ in t, the corresponding embedding π_2 of $P(\alpha, \beta_2, \gamma)$ in t can be constructed using the same case distinction as for tree patterns in Figure 3 and in an analogous manner. Proving that $P(\alpha, \beta_2, \gamma) \subseteq_S P(\alpha, \beta_1, \gamma)$ is analogous. □

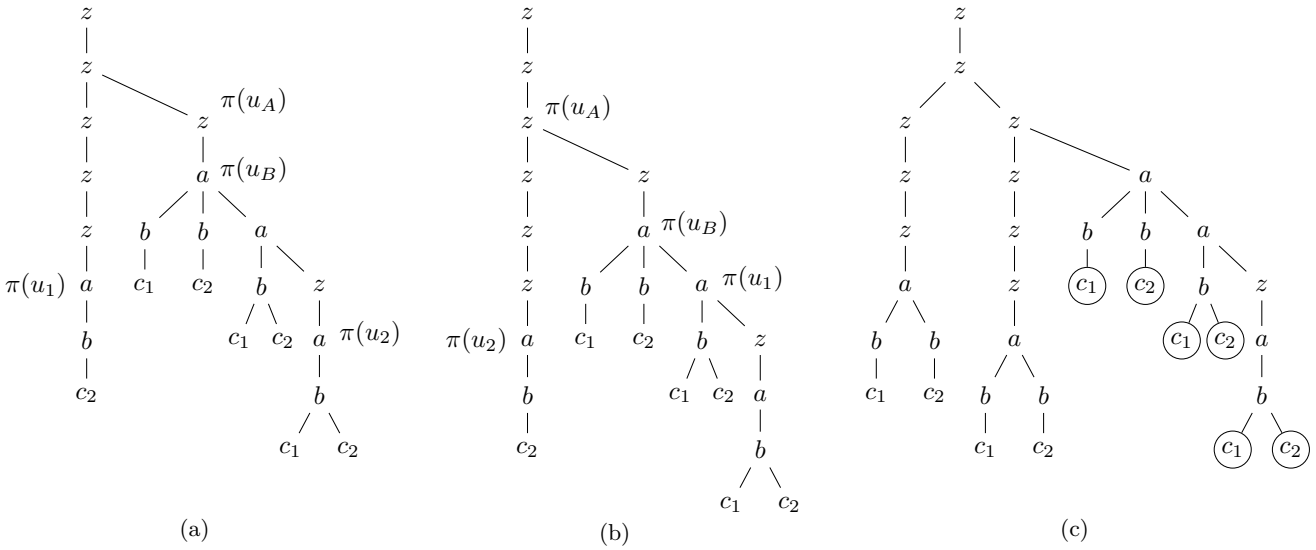


Figure 4: Trees for proving non-redundancy of p in Figure 3.

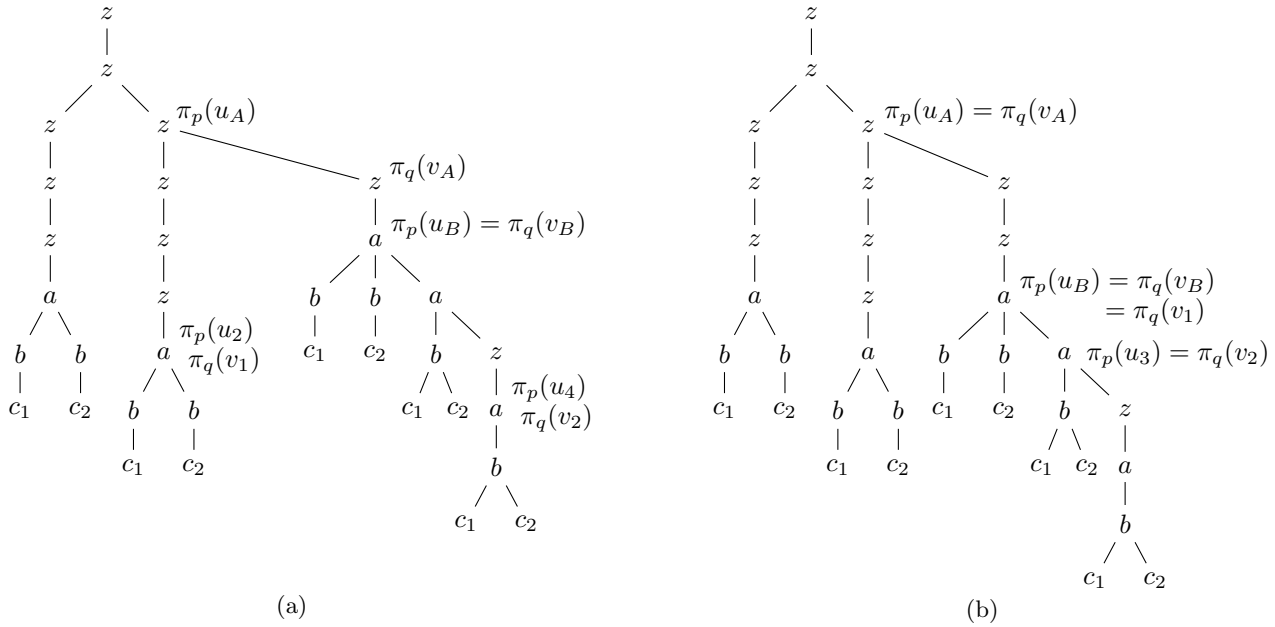


Figure 5: Canonical trees for proving that p and q in Figure 3 are equivalent.

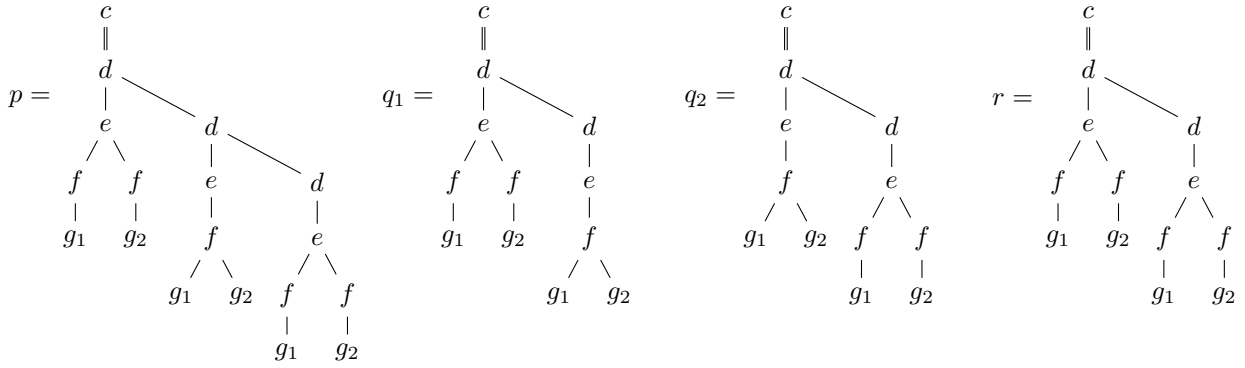


Figure 6: Tree patterns used to show that minimal tree patters can be structurally different.

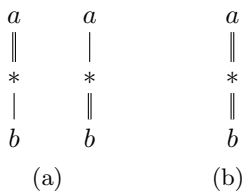


Figure 7: Two minimal but different tree patterns and their equivalent adorned pattern.

We are now ready to show the equivalence between the tree patterns.

PROPOSITION 3.3. *$P(p, q_1, r)$ is strongly equivalent to $P(p, q_2, r)$.*

PROOF SKETCH. The inclusions $q_1 \subseteq_S r$ and $q_2 \subseteq_S r$ are trivial: q_1 and q_2 restrict r by additionally requiring that two nodes labeled g_1 and g_2 should have the same parent. The inclusion $p \subseteq_S q_1$ holds since q_1 is a strict subpattern of p and the inclusion $p \subseteq_S q_2$ holds since there is a homomorphism π from q_2 to p that maps the topmost d-node of q_1 to the middle d-node of p. The equivalence then follows from Lemma 3.2. □

Minimality of $P(p, q_1, r)$ and $P(p, q_2, r)$ is more technical to prove and requires material which we develop in Section 4. More precisely, minimality of $P(p, q_1, r)$ and $P(p, q_2, r)$ follows from Lemma 4.2. We briefly explain why. Since p and r are $*$-narrow, their minimality is immediate because they are nonredundant (Lemma 2.3). It remains to prove that q_1 and q_2 are smallest tree patterns such that $p \subseteq_S q_i \subseteq_S r$ holds. The intuitive[3] reason is that, starting with tree pattern r, we can only get a smaller pattern q that still satisfies $q \subseteq_S r$ by merging f-nodes that are siblings. Furthermore, we can only merge one pair of f-nodes, because otherwise the pattern does not satisfy $p \subseteq_S q$ any more.

4. THE COMPLEXITY OF MINIMIZATION

In this section we study the complexity of the following problems.

TREE PATTERN MINIMIZATION	
Given:	A tree pattern p and $k \in \mathbb{N}$
Question:	Does there exist a tree pattern q such that size$(q) \leq k$ and $q \equiv_S p$?

MINIMALITY	
Given:	A tree pattern p
Question:	Is p minimal?

It was proved in [16, Theorem 5.9] that TREE PATTERN MINIMIZATION is coNP-hard. MINIMALITY is known to be NP-hard [23, Theorem 6.3]. The central theorem of this section is:

[3]The more formal proof is completely analogous to the proof of Claim 4.7, which we prove in Appendix C.2.

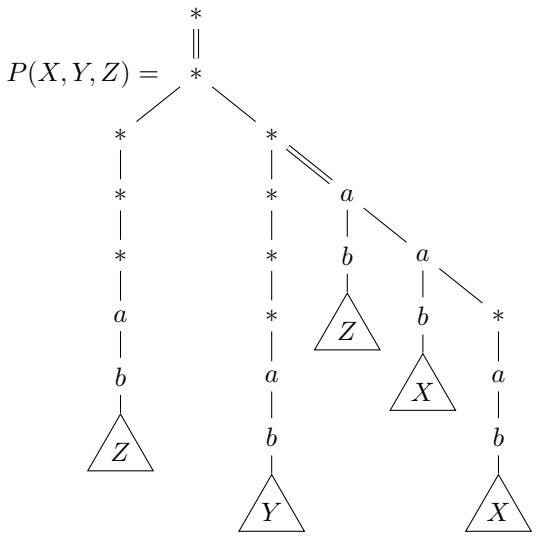


Figure 8: Gadget for constructing tree patterns (Lemma 4.3).

THEOREM 4.1. TREE PATTERN MINIMIZATION *is Σ_2^P-complete.*

Since testing non-redundancy of a tree pattern is NP-complete [23, Theorem 6.3], the above theorem proves that, under standard complexity theoretic assumptions, TREE PATTERN MINIMIZATION is more difficult than testing non-redundancy.

We will now prove Theorem 4.1. The Σ_2^P upper bound is straightforward: given an instance of TREE PATTERN MINIMIZATION consisting of tree pattern p and $k \in \mathbb{N}$, one can guess a tree pattern q of size at most k and test if it is equivalent to p. Since we make polynomially many non-deterministic guesses and since testing equivalence is coNP-complete, this is a Σ_2^P algorithm.

To show hardness, we will introduce an intermediate problem called *relative minimization*. We will give two reductions: the first is from relative minimization to tree pattern minimization and the second one shows that relative minimization is Σ_2^P-hard.

RELATIVE TREE PATTERN MINIMIZATION	
Given:	Minimal tree patterns p and r such that $p \subseteq_S r$ and $k \in \mathbb{N}$
Question:	Is there a pattern q such that size$(q) \leq k$ and $p \subseteq_S q \subseteq_S r$?

We will use the gadget $P(X, Y, Z)$ from Figure 8. Recall that, for tree patterns p, q, and r, we denote by $P(p, q, r)$ the tree pattern obtained from P by instantiating the subtrees marked X, Y, and Z by p, q, and r, respectively.

The following lemma is the technically most difficult result in the paper. It is crucial for connecting TREE PATTERN MINIMIZATION with RELATIVE TREE PATTERN MINIMIZATION and essentially proves two things. First, if one inserts minimal patterns p, q, r in the positions X, Y and Z, then one may still be able to obtain a

smaller pattern by changing q. Second, all minimal patterns that are equivalent to $P(p,q,r)$ and satisfy some side-conditions are, in a sense, similar to $P(p,q,r)$. It is this second part that has a very technical proof.

LEMMA 4.2. *Let p, q, and r be tree patterns that have at least one node, do not use labels in $\{a,b\}$, and such that $p \subseteq_S q \subseteq_S r$. Then $P(p,q,r)$ is minimal if and only if*

(1) p and r are minimal; and

(2) there is no tree pattern q' smaller than q such that $p \subseteq_S q' \subseteq_S r$.

The proof of the lemma considers an arbitrary pattern that is minimal and equivalent to $P(p,q,r)$ and proves step by step that it must be similar to $P(p,q,r)$. In a second step, we infer that the pattern at Y must be a smallest possible pattern q such that $p \subseteq_S q \subseteq_S r$. A particular challenge in the proof is the lack of methods that work for the general class of tree patterns.

LEMMA 4.3. RELATIVE TREE PATTERN MINIMIZATION *is reducible to* TREE PATTERN MINIMIZATION *in logarithmic space.*

PROOF. Consider an arbitrary instance of RELATIVE TREE PATTERN MINIMIZATION consisting of minimal tree patterns p and r such that $p \subseteq_S r$, and a number $k \in \mathbb{N}$. We can assume w.l.o.g. that p and r have at least one node. Since we can rename labels, we can also assume that p and r do not use the labels a or b. We will construct an instance p_m and $k' \in \mathbb{N}$ of TREE PATTERN MINIMIZATION so that p_m has an equivalent pattern of size at most k' if and only if there is a tree pattern q with $\text{size}(q) \leq k$ and $p \subseteq_S q \subseteq_S r$.

Tree pattern p_m is $P(p,p,r)$, where the gadget P is illustrated in Figure 8. We define k' as $k+2|p|+2|r|+20$.

We now prove that the reduction is correct. We need to prove two implications. For the first, assume that p, r, and k have a solution q w.r.t. RELATIVE TREE PATTERN MINIMIZATION. In this case we know from Lemma 3.2 that $p_m = P(p,p,r) \equiv_S P(p,q,r)$. Furthermore, the size of $P(p,q,r)$ is $\text{size}(q)+2|p|+2|r|+20 \leq k'$.

We prove the other implication. We assume that $p_m = P(p,p,r)$ has an equivalent pattern of size at most k' and we want to prove that there exists a pattern q of size at most k such that $p \subseteq_S q \subseteq_S r$. Let q a smallest pattern such that $p \subseteq_S q \subseteq_S r$.

By Lemma 3.2, we have that $P(p,q,r) \equiv_S p_m$. Tree patterns p and r are minimal and q is minimal among $p \subseteq_S q \subseteq_S r$ so by Lemma 4.2 pattern $P(p,q,r)$ is minimal as well. Therefore its size is at most $k' = k + 2|p| + 2|r| + 20$. This implies that $|q| \leq k$ and concludes the proof. □

To prove Theorem 4.1 it therefore only remains to prove that RELATIVE TREE PATTERN MINIMIZATION is Σ_2^P-complete, which we do next. We will reduce from the following problem, which is a mild variation of the canonical satisfiability problem of quantified $\exists\forall$-formulas ($\exists\forall$-QBF).

∃-VALIDITY	
Given:	A set of pairs of conjunctive clauses $\{(c_1^1, c_1^2), \ldots, (c_m^1, c_m^2)\}$ over variables $x_1, \ldots, x_n$
Question:	Is there a $(i_1, \ldots, i_m) \in \{1,2\}^m$ such that $c_1^{i_1} \vee \cdots \vee c_m^{i_m}$ is true for every valuation of $x_1, \ldots, x_n$?

Due to the similarity between ∃-VALIDITY and $\exists\forall$-QBF, it is not surprising that ∃-VALIDITY is Σ_2^P-complete.

LEMMA 4.4. ∃-VALIDITY *is Σ_2^P-complete.*

PROOF. Membership in Σ_2^P is obvious. For the other direction let

$$\Psi = \exists x_1, \ldots, x_n \forall y_1, \ldots, y_\ell \; \Phi(x_1, \ldots, x_n, y_1, \ldots, y_\ell)$$

be a QBF formula such that $\Phi = c_1 \vee \cdots \vee c_m$ is quantifier-free and in disjunctive normal form. We compute the ∃-VALIDITY instance

$$\{(c_i, c_i) \mid i \in [1,m]\} \cup \{(x_i, \neg x_i) \mid i \in [1,n]\} .$$

For the correctness of the reduction, we first observe, that the formula Ψ is equivalent to

$$\Psi' = \exists x_1, \ldots, x_n \forall y_1, \ldots, y_{n+\ell} \; \Phi(y_1, \ldots, y_{n+\ell}) \vee y_1 \neq x_1 \vee \cdots \vee y_n \neq x_n .$$

Now it is easy to see the correctness, as the pairs $(c_1, c_1), \ldots, (c_m, c_m)$ enforce that each original clause has to be satisfied (there is no choice) and the pairs $(x_1, \neg x_1), \ldots, (x_n, \neg x_n)$ allow an existential choice for the values of the x-variables as demonstrated in Ψ', i.e., if some x-variable x_i should be true, we choose $\neg x_i$ from the pair $(x_i, \neg x_i)$ and vice versa. □

We now use ∃-VALIDITY to prove that RELATIVE TREE PATTERN MINIMIZATION is Σ_2^P-complete, which is our final step in proving Theorem 4.1.

LEMMA 4.5. RELATIVE TREE PATTERN MINIMIZATION *is Σ_2^P-complete.*

PROOF. The upper bound follows from the straightforward algorithm: guess q and check whether $p \subseteq_S q \subseteq_S r$. Clearly, guessing q can be done by a polynomial number of guesses and the containment tests can be done in coNP by Theorem 2.1.

For the lower bound, we reduce from ∃-VALIDITY. We build on Miklau and Suciu's proof that containment of tree patterns is coNP-hard ([27, Proofs of Lemma 3 and Theorem 4]) and extend their idea. Let $I = \{(c_1^1, c_1^2), \ldots, (c_m^1, c_m^2)\}$ be an instance of ∃-VALIDITY. We compute the patterns p and r as given in Figure 9. We let $k = |r| - m$. Notice that the pattern p only depends on the *number* of clauses and variables of I (it uses m in the picture of p and n in the subpatterns C and D) and not on the clauses itself. Furthermore, p does not contain any wildcards and only contains descendant edges in its subquery D. Pattern r does contain wildcards in the subpatterns c_i^j, but these wildcards have only one child. Therefore, p and r are $*$-narrow patterns.

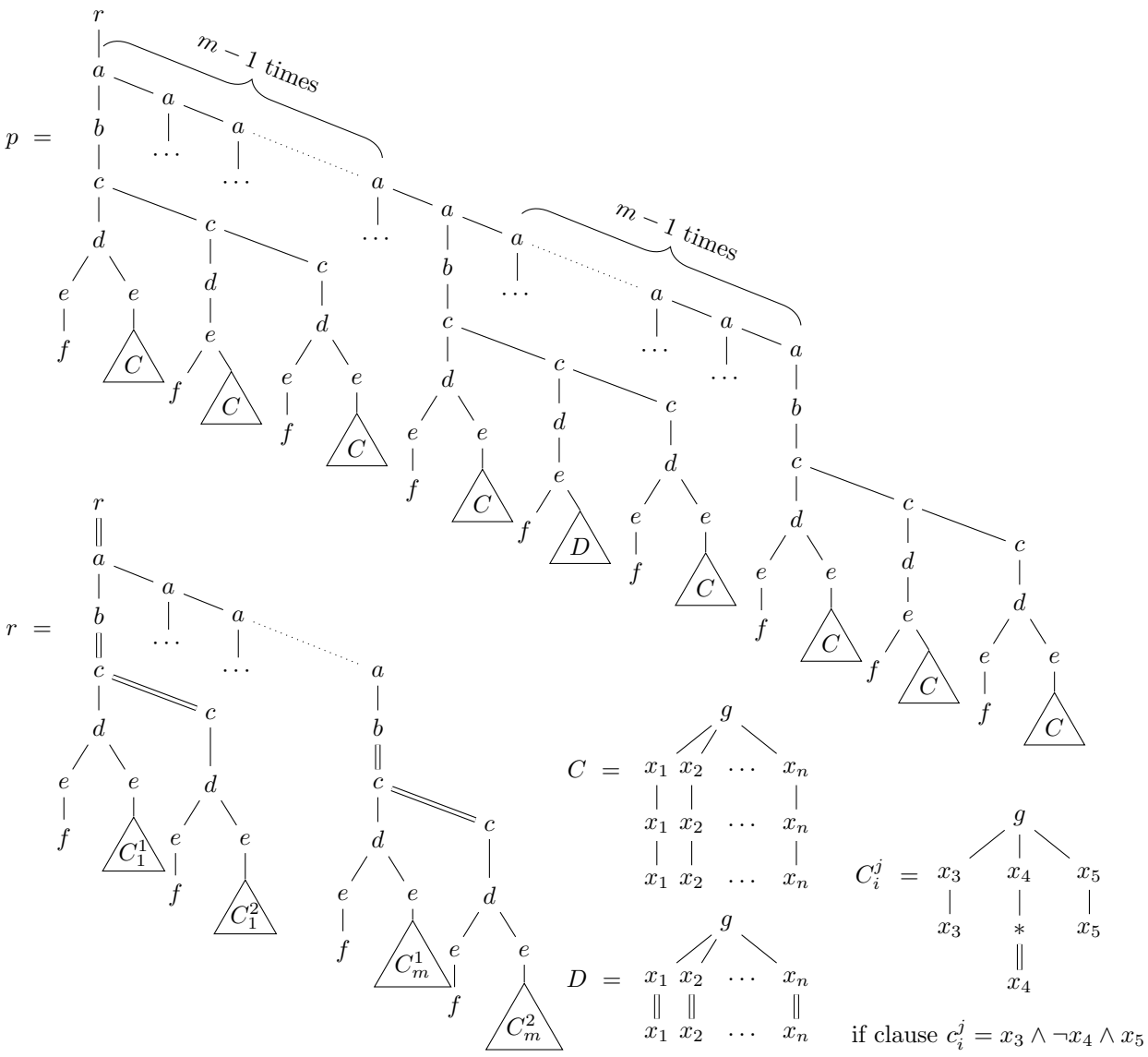


Figure 9: Patterns used in the proof of Theorem 4.1

The idea of p and r is that each subpattern of r that is rooted at a b-labeled node represents a pair of clauses and the subpattern C_i^j represents the clause c_i^j for each $i \in \{1, \ldots, m\}$ and $j \in \{1, 2\}$. The subpattern C_i^j has as root a g-labeled node. For each positive literal x_i of c_i^j, the g-labeled node has an x_i-labeled child that itself has an x_i-labeled child, connected by a child edge. For each negative literal $\neg x_i$ of c_i^j, the g-labeled node has an x_i-labeled child, that has a wildcard node as child which has an x_i-labeled child connected by a descendant edge.

Since the only descendant edges of p occur in the subpattern D, the canonical trees of p only differ from p in the subtree corresponding to D.

We will show that I is a true instance of $\exists$-VALIDITY if and only if p, r, and k are a true instance of RELATIVE TREE PATTERN MINIMIZATION. To this end, we first present two claims (both are proved in Appendix C) and prove correctness of the reduction based on these claims.

The first claim states that p, r, and k are an instance of RELATIVE TREE PATTERN MINIMIZATION.

CLAIM 4.6. *The tree patterns p and r are minimal.*

The second claim limits the form of solutions q to the instance. We say that a tree pattern q is in *normal form* if it can be obtained from pattern r by the following algorithm: Select in each subpattern rooted at a b-node at most one subpattern rooted at a d-node. In each selected subpattern, merge the two nodes labeled e.

CLAIM 4.7. *Every minimal pattern q that satisfies $p \subseteq_S q \subseteq_S r$ is in normal form.*

A pattern q in normal form that has size k and satis-

fies $p \subseteq_S q \subseteq_S r$, i.e., where, in every subpattern rooted at a b-labeled node, two e-labeled nodes are merged, is called a *solution* to I.

Let q be a tree pattern of size k in normal form. We denote by v_i^j for $i \in \{1,\dots,m\}$ and $j \in \{1,2\}$ the d-labeled node of q that is ancestor of the C_i^j subpattern. We define a function $\mathrm{f}^q : \{1,\dots,m\} \to \{1,2\}$ as follows:

$$\mathrm{f}^q(i) = \begin{cases} 1 & \text{if } v_i^1 \text{ has exactly one child} \\ 2 & \text{if } v_i^2 \text{ has exactly one child} \end{cases}$$

Notice that f^q is well-defined for every pattern q in normal form.

Let t be a canonical tree of pattern p and let $d(x_i)$ denote the distance of the two x_i-labeled nodes in the subtree of t corresponding to the D-subpattern of p. With t we associate a valuation σ^t of the variables $x_1,\dots,x_n$ as follows:

$$\sigma^t(x_i) = \begin{cases} \text{true} & \text{if } d(x_i) = 1 \\ \text{false} & \text{if } d(x_i) > 1 \end{cases}$$

We can show the following points, which prove the equivalence between tree pattern minimization and ∃-VALIDITY.

(a) If q is a solution to I, then σ^t satisfies

$$c_1^{\mathrm{f}^q(1)} \vee \dots \vee c_m^{\mathrm{f}^q(m)}$$

for every canonical tree t of r. This shows universality of the formula because there exists a canonical tree t of p such that $\sigma^t = \rho$ for each possible valuation ρ of variables.

For the other direction, let $\iota : \{0,\dots,m\} \to \{1,2\}$ be a choice of clauses. We can show the following.

(b) If the formula $c_1^{\iota(1)} \vee \dots \vee c_m^{\iota(m)}$ is universally true, then the normal form pattern q of size k such that

$$\mathrm{f}^q(j) = \iota(j) \text{ for all } j \in \{1,\dots,m\}$$

satisfies $r \subseteq_S q \subseteq_S p$.

The statements (a) and (b) together with Claims 4.6 and 4.7 show that the reduction to tree pattern minimization is correct. Notice that the patterns p and q and the number k can be computed in LOGSPACE.

It therefore remains to prove statements (a) and (b) and Claims 4.6 and 4.7. We do this in Appendix C. □

The techniques we used for proving that TREE PATTERN MINIMIZATION is Σ_2^P-complete can also be used to prove that MINIMALITY is Π_2^P-complete. The proof for minimality uses the same ideas as the proof for minimization, but it is not immediate, as hardness for exists k, does not imply hardness for $k = n - 1$. The basic observation idea is that we can compute a tree pattern q such that $p \subseteq_S q \subseteq_S r$ and $\mathrm{size}(q) = k+1 = |r|-m+1$. Therefore, $P(p,q,r)$ is minimal if and only if the underlying ∃-validity instance is a false instance.

THEOREM 4.8. MINIMALITY *is* Π_2^P*-complete.*

PROOF SKETCH. Membership in Π_2^P is trivial: the algorithm just has to check whether each smaller pattern is non-equivalent. The latter can be done in NP since equivalence of tree patterns is in coNP.

For Π_2^P-hardness, we use essentially the same (combined) reduction as in the proofs of Lemma 4.5 and 4.3. Let I be an instance of ∃-VALIDITY and let p and r be the patterns computed in the reduction from ∃-VALIDITY to RELATIVE TREE PATTERN MINIMIZATION in the proof of Lemma 4.5.

We compute a pattern q from p by merging the e-nodes above the subpatterns C_1^1 to C_{m-1}^1 with their siblings.

It is easy to see that $p \subseteq_S q \subseteq_S r$. The second inclusion holds again, because we restrict the trees by merging nodes. The first inclusion holds, because there exists a homomorphism from q to p which embeds the lowest b-subpattern of q on the b-subpattern containing D. The two c-labeled nodes from the q-pattern can be embedded on the upper and lower c-labeled nodes inside the b-subpattern of p.

Finally, we ask whether the pattern $P(p,q,r)$ is minimal. If the answer is yes, then I has no solution because we already know that a solution to I implies the existence of a pattern q' with $\mathrm{size}(q') = \mathrm{size}(r) - m < \mathrm{size}(q) = \mathrm{size}(r) - m + 1$ and $p \subseteq_S q' \subseteq_S r$. By Lemma 3.2, we know that $P(p,q',r) \equiv_S P(p,q,r)$. Furthermore, $P(p,q',r)$ is smaller than $P(p,q,r)$.

On the other hand, if there exists a pattern $P_{\min}$ with $P_{\min} \equiv_S P(p,q,r)$ and $\mathrm{size}(P_{\min}) < \mathrm{size}(P(p,q,r))$, then, by Lemma 4.2, we know that there exists a pattern q' with $\mathrm{size}(q') < \mathrm{size}(q)$ and $p \subseteq_S q' \subseteq r$. As $\mathrm{size}(q') < \mathrm{size}(q) = \mathrm{size}(r) - m + 1$ (and therefore $\mathrm{size}(q) \leq \mathrm{size}(r) - m$), we know from the reduction in Lemma 4.5, that I has a solution. □

5. DISCUSSION AND OUTLOOK

Boolean versus k-ary Queries.

We proved that minimization for Boolean tree patterns is Σ_2^P-complete. This result can also be extended to k-ary tree patterns (as considered in [27, 23]). However, the technique to transfer this result is not the usual one from [23, Section 5] because, as the authors say, it is not clear if the reduction presented there preserves minimality. We can transfer the complexity directly, however. For k-ary queries, the Σ_2^P upper bound follows by applying the naive algorithm and the lower bound follows from attaching all k output nodes to the root of our gadgets.

Conjunctive Queries over Trees.

We believe that our reduction can be used to show that minimization of conjunctive queries over trees (using child and descendant relations) is Σ_3^P-complete. In particular, we believe that Lemma 4.2 also holds true when p, q, and r are conjunctive queries over trees sat-

isfying certain sanity conditions[4]. In fact, in our proof we do not require the languages depicted by p, q and r to be defined by tree patterns.

We believe that combining the ideas from the proof of Lemma 4.5 combined with ideas from [7] allows to show Σ_3^P-hardness of relative conjunctive query minimization (defined analogously to relative tree pattern minimization). It also seems that Lemma 4.3 can be adapted to show Σ_3^P-hardness of conjunctive query minimization.

Lessons for Minimization.

This work gives new insights on how minimal tree patterns may need to be obtained and is the first to give a tight complexity bound for doing so. Even though our main result is a hardness result, we believe that the new insights can be used to develop better tree pattern optimization algorithms. For example, we know that minimization of $*$-narrow tree patterns can always be done by (iteratively) removing leaves [16]. It was long believed that all tree patterns can be minimized in such a way (in [15] it was claimed to be true) but, from this paper we now know that this is not the case and, in particular, it may also be necessary to merge nodes.

This is a fact that we can use for developing better heuristics for greedy tree pattern minimization. That is, we can approximate the minimal pattern by iteratively removing a leaf or merging two nodes and testing if the resulting tree pattern is still equivalent; until no such operation can be done anymore. If a polynomial-time algorithm for the equivalence test can be used, we have a polynomial-time algorithm for approximating minimal trees patterns.

We note that the presented approach is still a heuristic and does not always produce a minimal pattern. Indeed, not every minimal tree pattern can be obtained from a given pattern by removing leaves and merging nodes. For example, it may even be necessary to split nodes. This is easy to see if we take an instance of $P(p, q, r)$, where p and r are the patterns from Figure 9 and q results from r by merging the "wrong" nodes. It is an interesting question which (non-trivial) set of operations would be sufficient to guarantee that a minimal pattern can always be obtained by applying a sequence of such operations to the input pattern.

Acknowledgments

We are very grateful to Benny Kimelfeld for insightful discussions and for bringing the tree pattern query minimization problem to our attention.

6. REFERENCES

[1] S. Abiteboul, L. Segoufin, and V. Vianu. Static analysis of active XML systems. *ACM Trans. Database Syst.*, 34(4), 2009.

[2] N. Alechina, S. Demri, and M. de Rijke. A modal perspective on path constraints. *J. Log. Comput.*, 13(6):939–956, 2003.

[4]In particular p, q, and r should not allow to embed any nodes above the b-nodes of $P(p, q, r)$.

[3] S. Amer-Yahia, S. Cho, L. V. S. Lakshmanan, and D. Srivastava. Tree pattern query minimization. *VLDB J.*, 11(4):315–331, 2002.

[4] M. Arenas, P. Barceló, L. Libkin, and F. Murlak. *Foundations of Data Exchange*. Cambridge University Press, 2014.

[5] P. Barceló, L. Libkin, A. Poggi, and C. Sirangelo. XML with incomplete information. *J. ACM*, 58(1):4, 2010.

[6] M. Benedikt, W. Fan, and F. Geerts. XPath satisfiability in the presence of DTDs. *J. ACM*, 55(2), 2008.

[7] H. Björklund, W. Martens, and T. Schwentick. Conjunctive query containment over trees. *J. Comput. Syst. Sci.*, 77(3):450–472, 2011.

[8] H. Björklund, W. Martens, and T. Schwentick. Validity of tree pattern queries with respect to schema information. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 171–182, 2013.

[9] S. Cassidy. Generalizing XPath for directed graphs. In *Extreme Markup Languages Conference*, 2003.

[10] E. P. F. Chan. Containment and minimization of positive conjunctive queries in OODB's. In *Symposium on Principles of Database Systems (PODS)*, pages 202–211, 1992.

[11] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Symposium on Theory of Computing (STOC)*, pages 77–90, 1977.

[12] C. Chekuri and A. Rajaraman. Conjunctive query containment revisited. *Theor. Comput. Sci.*, 239(2):211–229, 2000.

[13] D. Chen and C. Y. Chan. Minimization of tree pattern queries with constraints. In *International Conference on Management of Data (SIGMOD)*, pages 609–622, 2008.

[14] W. Czerwiński, W. Martens, P. Parys, and M. Przybyłko. The (almost) complete guide to tree pattern containment. In *Symposium on Principles of Database Systems (PODS)*, pages 117–130, 2015.

[15] S. Flesca, F. Furfaro, and E. Masciari. On the minimization of XPath queries. In *VLDB*, pages 153–164, 2003.

[16] S. Flesca, F. Furfaro, and E. Masciari. On the minimization of XPath queries. *J. ACM*, 55(1), 2008.

[17] G. H. L. Fletcher, M. Gyssens, D. Leinders, J. V. den Bussche, D. V. Gucht, S. Vansummeren, and Y. Wu. Relative expressive power of navigational querying on graphs. In *International Conference on Database Theory (ICDT)*, pages 197–207, 2011.

[18] A. Gheerbrant, L. Libkin, and C. Sirangelo. Reasoning about pattern-based XML queries. In *International Conference on Web Reasoning and Rule Systems (RR)*, pages 4–18, 2013.

[19] G. Gottlob, C. Koch, and R. Pichler. Efficient

algorithms for processing XPath queries. *ACM Trans. Database Syst.*, 30(2):444–491, 2005.

[20] G. Gottlob, C. Koch, and K. U. Schulz. Conjunctive queries over trees. *J. ACM*, 53(2):238–272, 2006.

[21] G. Gottlob, N. Leone, and F. Scarcello. The complexity of acyclic conjunctive queries. *J. ACM*, 48(3):431–498, 2001.

[22] Gremlin Language. https://github.com/tinkerpop/gremlin/wiki, 2013.

[23] B. Kimelfeld and Y. Sagiv. Revisiting redundancy and minimization in an XPath fragment. In *International Conference on Extending Database Technology (EDBT)*, pages 61–72, 2008.

[24] E. V. Kostylev, J. L. Reutter, and D. Vrgoc. Containment of data graph queries. In *International Conference on Database Theory (ICDT)*, pages 131–142, 2014.

[25] L. Libkin, W. Martens, and D. Vrgoc. Querying graph databases with XPath. In *International Conference on Database Theory (ICDT)*, pages 129–140, 2013.

[26] M. Marx. XPath and modal logics of finite DAG's. In *International Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, pages 150–164, 2003.

[27] G. Miklau and D. Suciu. Containment and equivalence for a fragment of XPath. *J. ACM*, 51(1):2–45, 2004.

[28] T. Milo and D. Suciu. Index structures for path expressions. In *International Conference on Database Theory (ICDT)*, pages 277–295, 1999.

[29] F. Neven and T. Schwentick. On the complexity of XPath containment in the presence of disjunction, DTDs, and variables. *Logical Methods in Computer Science*, 2(3), 2006.

[30] P. Ramanan. Efficient algorithms for minimizing tree pattern queries. In *International Conference on Management of Data (SIGMOD)*, pages 299–309, 2002.

[31] J. Robie, D. Chamberlin, M. Dyck, and J. Snelson. XML Path Language 3.0. Technical report, World Wide Web Consortium, April 2014. Recommendation, http://www.w3.org/TR/2014/REC-xpath-30-20140408/.

[32] S. Staworko and P. Wieczorek. Characterizing XML twig queries with examples. In *International Conference on Database Theory (ICDT)*, pages 144–160, 2015.

[33] P. T. Wood. Minimising simple XPath expressions. In *WebDB*, pages 13–18, 2001.

[34] W. Xu and Z. M. Özsoyoglu. Rewriting XPath queries using materialized views. In *International Conference on Very Large Data Bases (VLDB)*, pages 121–132, 2005.

[35] M. Yannakakis. Algorithms for acyclic database schemes. In *International Conference on Very Large Data Bases*, pages 82–94, 1981.

APPENDIX

The Appendix is structured as follows: Appendix A introduces some notation that will be used throughout the Appendices. Appendix B contains some technical lemmas that are not specific to our gadget in Figure 8, while Appendix C is dedicated to show Lemmas 4.2 and 4.5 and proves properties specific to the structure of the tree pattern in Figure 8.

A. BASIC NOTIONS

Subtrees and Subpatterns.

Recall that, for a tree pattern p and node u we use p^u to denote the subtree of p rooted at u. For a child v of the root of p, we denote by p_v the pattern consisting of the root of p, connected to the subpattern p^v in the same way as they are connected in p. We illustrate the notation in Figure 10. In Figure 10(a) we assume that u has precisely two children. Then, the root of p^u also has exactly two children (Figure 10(b)) and the types of edges are inherited from p. Figures 10(c) and Figures 10(d) illustrate the notation with subindices but already start from the pattern p^u.

Canonical Embeddings.

Given a canonical tree t of a tree pattern p, we sometimes consider injective embeddings π_t of p in t, such that for every node u of p, we have that $t^{\pi_t(u)}$ is a canonical tree of p^u. Notice that the "canonical" embedding of p in t (i.e., the one which we use in the definition of canonical trees t) is always such an injective embedding but, given a tree pattern p and canonical tree t, there may be more than one such injective embedding.

For such injective embeddings π_t we sometimes also consider the inverse π_t^{-1} which is a partial function that is defined for all nodes, except nodes inserted into t due to descendant edges. In particular, π_t^{-1} is defined for all nodes with a label different from z.

B. PREPARING THE COMPLEXITY PROOF

The following lemma is very similar to Proposition 3 in [27] but is about weak inclusion instead of strong inclusion.

LEMMA B.1. *Let p and q be tree patterns. Then $p \subseteq_W q$ if and only if $Can(p) \subseteq L_W(q)$.*

PROOF. The "only if" direction follows immediately from the fact that $\mathrm{Can}(p) \subseteq L_W(p)$, so in order to have $L_W(p) \subseteq L_W(q)$ we need to have $\mathrm{Can}(p) \subseteq L_W(q)$.

In order to show the "if" direction assume towards a contradiction that $\mathrm{Can}(p) \subseteq L_W(q)$ but $L_W(p) \not\subseteq L_W(q)$. Therefore, there exists a tree $t \in L_W(p)$ such that $t \notin L_W(q)$. We will apply some modifications to the tree t obtaining a tree in $\mathrm{Can}(p)$, but not belonging to $L_W(q)$, which will contradict our assumption. Let V_p be the set of nodes of p, V be the set of nodes of t and let $\pi : V_p \to V$ be a weak embedding of p in t. Then let t' be a subtree of t obtained by cutting off all the nodes which do not have a descendant in $\pi(V_p)$. Observe that p also embeds in t', by the same embedding π. We also still have $t' \notin L_W(q)$. Let t'' be the tree t' relabeled appropriately. Concretely, every node of t' which is not equal to $\pi(v)$ for some non-wildcard node v is relabeled to z. Note that still $t'' \in L_W(p)$ and $t'' \notin L_W(q)$. Moreover t'' is now a canonical tree of p. Thus indeed $\mathrm{Can}(p) \not\subseteq L_W(q)$, which is a contradiction and finishes the proof. □

LEMMA B.2 (LEMMA 4.4 IN [16]). *Let p and q be tree patterns. Then $p \subseteq_S q$ if and only if, for each child j of q's root there exists a child i of p's root such that $p_i \subseteq_S q_j$.*

Lemma B.2 can be used to prove the following result which is useful to infer similarities between equivalent patterns.

LEMMA B.3. *Let $p \equiv_S q$ and both p and q are strongly nonredundant. Then p and q have the same number of children n. Moreover, there exists a bijection $\varphi : \{1,\dots,n\} \to \{1,\dots,n\}$ such that for all $i \in \{1,\dots,n\}$ it holds $p_i \equiv_S q_{\varphi(i)}$.*

PROOF. Since $p \subseteq_S q$. Say that p's root has n children $\{1,\dots,n\}$ and q's root has m children $\{1,\dots,m\}$. By Lemma B.2, for every q_j (where $j \in \{1,\dots,m\}$), there exists a p_i (where $i \in \{1,\dots,n\}$) such that $p_i \subseteq_S q_j$. Define f to be the function that maps each such j to the corresponding i.

Similarly, for every $i \in \{1,\dots,n\}$, there exists a $g(i) \in \{1,\dots,m\}$ such that $q_{g(i)} \subseteq_S p_i$.

We will show that $f = g^{-1}$, so $f \circ g$ is the identity. Assume otherwise; say that $f(g(i)) = i' \neq i$. Then

$$p_i \supseteq_S q_{g(i)} \supseteq_S p_{f(g(i))} = p_{i'}.$$

However, this means that p is strongly redundant. Indeed: by removing the subtree p^i we would obtain a strongly equivalent pattern. This is a contradiction.

So, $f = g^{-1}$, which means in particular that the set of branches of p and q have the same cardinality and there is a bijection φ (defined as $\varphi(i) = f(i)$) such that

$$\forall j \in \{1,\dots,n\} : p_j \subseteq_S q_{\varphi(j)} \text{ and } p_j \supseteq_S q_{\varphi(j)}$$

which simply means that

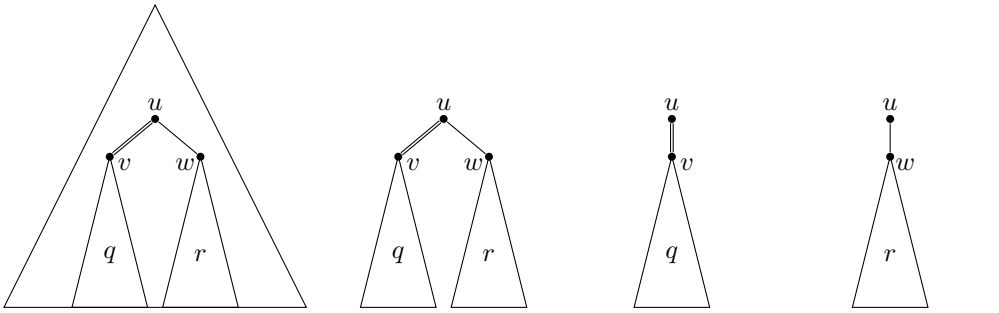
$$\forall j \in \{1,\dots,n\} : p_j \equiv_S q_{\varphi(j)}.$$

This concludes the proof. □

The following is a corollary of Lemma 4.11 in [23].

LEMMA B.4. *Let $p \equiv_W q$ and both p and q are weakly nonredundant. Then p and q have the same number of children n. Moreover there exists a bijection $\varphi : \{1,\dots,n\} \to \{1,\dots,n\}$ such that for all $i \in \{1,\dots,n\}$ it holds $p_i \equiv_W q_{\varphi(i)}$.*

PROOF SKETCH. Lemma 4.11 in [23] is phrased differently from the present lemma and, in particular, its second condition is about *relative containment*. However, as the paper explains after Definition 3.11, in this case, weak containment in both directions (and therefore weak equivalence) is implied by the two directions of relative containment. □



(a) A pattern p. (b) Subpattern p^u. (c) Subpattern p^u_v. (d) Subpattern p^u_w.

Figure 10: Our notation for subtrees and subpatterns. All labels or wildcards are inherited from p.

LEMMA B.5 (IMPLICIT IN THEOREM 4.3 IN [23]). *Let p be a tree pattern such that the root of p is not labeled $*$. Then, for every tree pattern q:*

(a) $p \subseteq_W q$ implies $p \subseteq_S q$; and

(b) $p \equiv_W q$ implies $p \equiv_S q$.

Lemma B.5(b) is presented as case 1 of Theorem 4.3 in [23]. The presented proof shows both inclusions independently and therefore also shows Lemma B.5(a), even if not stated explicitly.

LEMMA B.6. *Let $k \in \mathbb{N}$ and p and q be two tree patterns such that $p \subseteq_W q$. Then for every k-descendant v of root(q), there exists a k-descendant u of root(p), such that $p^u \subseteq_W q^v$.*

PROOF. Assume towards a contradiction that there exists a k-descendant v of q such that there is no k-descendant u of p with $p^u \subseteq_W q^v$.

Let $t \in \mathrm{Can}(p)$ be a canonical tree of p such that

(a) every descendant edge above depth k is embedded in a path of length one (i.e., in a child edge); and

(b) for every k-descendant w of root(t), it holds that q^v cannot be weakly embedded in t^w.

By assumption and by Lemma B.1, such a canonical tree exists. By (b), we know that q^v also cannot be strongly embedded on any node of depth at least k in t. Therefore, q cannot be weakly embedded in t, which is a contradiction to $p \subseteq_W q$. □

LEMMA B.7. *Let p be a tree pattern such that its root has only one child u. Furthermore, let the edge from the root of p to u be a descendant edge. Then if p is strongly nonredundant then p^u is weakly nonredundant.*

PROOF. Towards a contradiction, assume that p^u is weakly redundant, thus there exists a leaf n such that $p^u \equiv_W p^u \setminus n$. We will show that $p \equiv_S p \setminus n$. It is clear that $p \subseteq_S p \setminus n$, so it remains to show that $p \setminus n \subseteq_S p$. Consider a tree $t \in \mathrm{Can}(p \setminus n)$ and fix an injective embedding π_t of $p \setminus n$ to t. Let u_t be the image of u in t through the embedding π_t, thus $t^{u_t} \in \mathrm{Can}(p^u \setminus n)$. Then we have that p^u weakly embeds into t^{u_t} because $p^u \equiv_W p^u \setminus n$. Therefore, p strongly embeds into t. This shows that in any canonical tree of $p \setminus n$ the pattern p strongly embeds, which means that $p \setminus n \subseteq_S p$. This would imply that p is strongly redundant, which is a contradiction. □

LEMMA B.8. *Let $p \equiv_S q$ be two tree patterns that are both strongly nonredundant. Let furthermore the roots of p and q have exactly one child (u and v, respectively). Moreover let the edge from the root of p be a descendant edge. Then $p^u \equiv_W q^v$ and both p^u and q^v are weakly nonredundant.*

PROOF. We first show $p^u \subseteq_W q^v$. Let $t' \in \mathrm{Can}(p^u)$. Let t be a tree obtained from t' by adding a new root above root(t') and labeling it by the label of root(p). Clearly we have that $t \in \mathrm{Can}(p)$. By the strong equivalence between p and q, there exists a strong embedding of q into t. In this embedding, the image of v is a node below root(t), that is, a node inside t'. This means that q^v weakly embeds into t', which shows that $p^u \subseteq_W q^v$.

The proof for the other direction is completely symmetric. We note that we did not use the fact that $(\mathrm{root}(p), u)$ is a descendant edge until now.

It remains to show the weak nonredundancy of p^u and q^v.

Since we know that $(\mathrm{root}(p), u)$ is a descendant edge, Lemma B.7 immediately gives us that p^u is weakly nonredundant. The same holds for q^v if $(\mathrm{root}(q), v)$ is a descendant edge. Therefore, we assume in the following that $(\mathrm{root}(q), v)$ is a child edge.

Let T be the set of nodes of q which are reachable from root(q) by only following child edges. We first show that every node in $T \setminus \{\mathrm{root}(q)\}$ is labeled by $*$. Take an arbitrary $c \in T \setminus \{\mathrm{root}(q)\}$. Consider a tree $t \in \mathrm{Can}(p)$ where, in the canonical embedding, the edge $(\mathrm{root}(p), u)$ is mapped into a path longer than depth of c in q. By the strong equivalence of p and q we know that there exists a strong embedding of p in t. Since c is connected to root(q) by only child edges, such a strong embedding maps c to a node labeled by z, which means that c has to be labeled by $*$.

We can conclude that the pattern q looks as follows: Below the root, there is a non-empty region T of nodes which are connected to the root by paths of child edges. All nodes in this region are labeled by $*$.

Below the region T, there may still be other nodes, but every edge that leaves T is a descendant edge. We use this to information to show that q^v is indeed weakly nonredundant. Let w be a leaf of q and assume towards a contradiction that $q^v \equiv_W q^v \setminus w$. We show that then $q \equiv_S q \setminus w$.

Clearly $q \subseteq_S q \setminus w$, so it remains to show that $q \supseteq_S q \setminus w$. Let $t \in \text{Can}(q \setminus w)$. Let c be the child of root(t). Then clearly $t^c \in \text{Can}(q^v \setminus w)$ because $(\text{root}(q), v)$ is a child edge. As $q^v \equiv_W q^v \setminus w$, we have that q^v weakly embeds in t^c. Take an arbitrary such embedding π. We now construct a strong embedding π' of q in t. For a node $d \notin T$ we define $\pi'(d) = \pi(d)$. For a node $d \in T$ we define $\pi'(d)$ to be the ancestor of $\pi(d)$, which is at the depth equal to the depth of d in q. If d is the root of q it is embedded therefore into the root of t, so the label is the same because $t \in \text{Can}(q \setminus w)$. If $d \in T$ is not the root, we know that d is labeled by $*$, so the label of $\pi'(d)$ does not matter. One can easily verify that all the other conditions are fulfilled and π' is indeed a strong embedding of q in t. This contradicts the strong nonredundancy of q. Thus indeed q^v is weakly nonredundant, which finishes the proof. □

C. PROOFS FOR SECTION 4

We start with a simple observation. Let $t \in \text{Can}(\alpha)$. By definition of α and canonical trees, t has a unique a-labeled node that has an a-labeled child.

OBSERVATION C.1. *Let $t \in Can(\alpha)$, let π be a weak embedding of α in t and let n_{aa} be the unique a-labeled node with an a-labeled child in t. Then $\pi(u_{RR}) = n_{aa}$.*

Before we prove Lemma 4.2, we do a first step by showing that the pattern $P(p, q, r)$ is strongly nonredundant, if the conditions (1) and (2) of Lemma 4.2 are satisfied.

DEFINITION C.2. We say that a pattern α is *barely included* in a pattern β if $\alpha \subseteq_S \beta$ but, for every leaf n of α, it holds that $\alpha \setminus n \not\subseteq_S \beta$. We denote it by $\alpha \subseteq_S^B \beta$.

LEMMA C.3. *Let p, q, and r be strongly nonredundant patterns such that $q \subseteq_S^B r$. Then $P(p, q, r)$ is strongly nonredundant.*

PROOF. Let $\alpha = P(p, q, r)$. We show that, for every leaf u of α, we have $\alpha \setminus u \not\equiv_S \alpha$. The proof will go by a case distinction depending on where u is located. Let $T_{i,j}(t_1, t_2, t_3, t_4, t_5)$ for $t_1, t_3 \in \text{Can}(r)$, $t_2 \in \text{Can}(q)$ and $t_4, t_5 \in \text{Can}(p)$ be the canonical tree of α in which the i-th pattern p, q or r from the left (ordered as in Figure 8) was mapped into t_i, the descendant edge from the root is mapped to a path of length i, and the descendant edge (u_R, u_{RR}) is mapped to a path of length j. Let $t_p \in \text{Can}(p)$, $t_q \in \text{Can}(q)$, and $t_r \in \text{Can}(r)$ be arbitrary but fixed canonical trees.

Consider first the case where u is a descendant of the node u_L, i.e., it is a leaf in the left-most subpattern r. We know that $r \not\equiv_S r \setminus u$, so by Lemma 2.4 there exists a tree $t \in \text{Can}(r \setminus u)$ such that $t \notin L_S(r)$. We claim that the tree $T = T_{1,1}(t, t_q, t_r, t_p, t_p)$ does not belong to $L_S(\alpha)$ (see Figure 11(a)). Towards a contradiction, assume otherwise. Therefore, there must exist a strong embedding of α in T. By Observation C.1, the node u_{RR} has to be mapped to n_{aa}. Since u_{RR} is mapped into n_{aa}, which is on depth 3, the node u_ε has to be mapped into a node at depth 1 and thus the node u_{LD} has to be mapped into a node at depth 5. Therefore u_{LD} also has to be mapped to n_{LD} in T, because it is the only a-labeled node in T at depth 5. Hence the pattern r has to be mapped into t, which, as we know, is impossible. So indeed $T \notin L_S(\alpha)$.

Assume now that u is a descendant of the node u_{RL}, so it is a leaf in the subpattern q. This is the most complicated case. Since q is barely included in r we know by Lemma 2.4 that there exists a tree $t \in \text{Can}(q \setminus u)$ such that $t \notin L_S(r)$. As $L_S(q) \subseteq L_S(r)$ then clearly also $t \notin L_S(q)$. We claim that the tree $T = T_{1,2}(t_r, t, t_r, t_p, t_p)$ (see Figure 11(b)) does not belong to $L_S(\alpha)$. Again, towards a contradiction, assume otherwise and consider an arbitrary strong embedding of α in T. We note that in T, there is exactly one a-node on depth 6: the node n_{RLD}. Therefore, if u_ε embeds in n_ε in T then u_{RLD} has to embed in some a-labeled node on depth 6, that is, in n_{RLD}. This means that q would have to strongly embed into t, which is impossible. Thus, u_ε cannot embed into n_ε. Notice that, again, by Observation C.1, u_{RR} has to embed into n_{aa}. Here, n_{aa} is a node on depth 4. Therefore the only remaining option is that u_ε embeds into n_R, a node on depth 2. Notice now that u_{LD} has to embed into some a-labeled node 4 levels lower. There is however only one such node in T, which is n_{RLD}. This means that r has to embed into t, which, as we already said, is impossible. Thus we also have that $T \notin L_S(\alpha)$ in this case.

For the remainder of the proof, notice first that it is easy to show that since p and r are strongly nonredundant then so is $\alpha^{u_{RR}}$. Consider now the case when u is a descendant of node u_{RR}. As $\alpha^{u_{RR}}$ is strongly nonredundant, there exists a tree $t \in \text{Can}(\alpha^{u_{RR}} \setminus u)$ such that $t \notin L_S(\alpha^{u_{RR}})$. To construct this t we use the trees t_3, t_4 and t_5. We claim now that $T = T_{1,1}(t_r, t_q, t_3, t_4, t_5)$ (see Figure 11(c)) does not belong to $L_S(\alpha)$. Indeed, towards a contradiction assume that there exists a strong embedding of α in T. Again by Observation C.1, u_{RR} has to embed into n_{aa}. This however means that $\alpha^{u_{RR}}$ has to strongly embed into t, which is impossible, as we said before. Thus also in this case $\alpha \not\equiv_S \alpha \setminus u$, which finishes the proof that α is strongly nonredundant. □

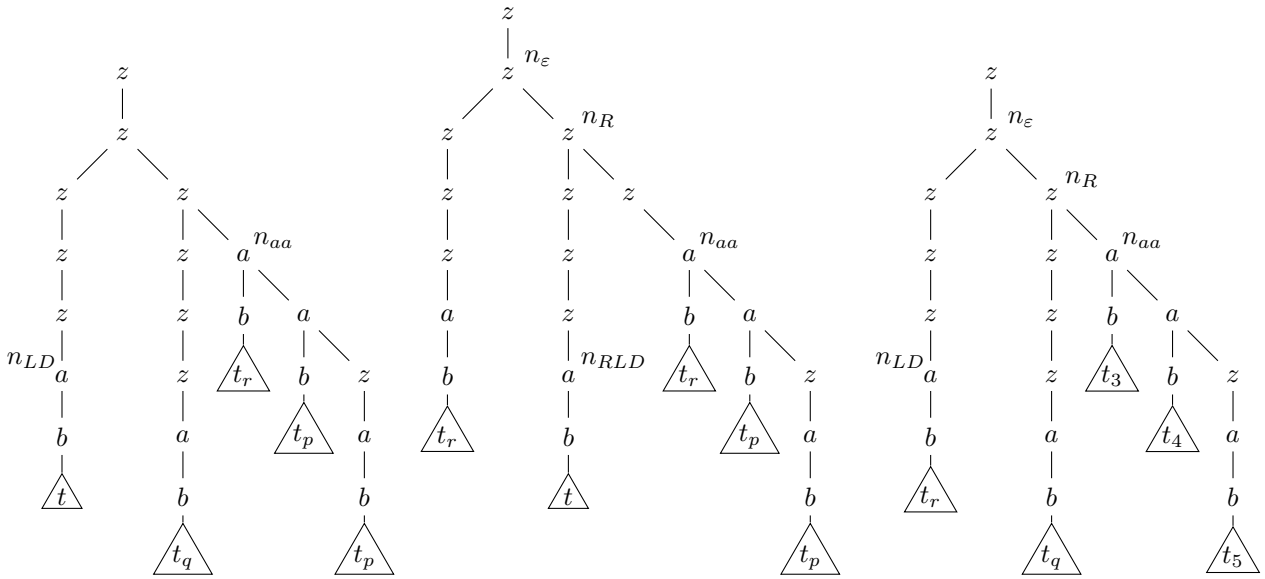
C.1 Proof of Lemma 4.2

LEMMA 4.2: *Let p, q, and r be tree patterns that have at least one node, do not use labels in $\{a, b\}$, and such that $p \subseteq_S q \subseteq_S r$. Then $P(p, q, r)$ is minimal if and only if*

(1) p and r are minimal; and

(2) there is no tree pattern q' smaller than q such that $p \subseteq_S q' \subseteq_S r$.

PROOF. The "only if"-direction is a corollary from Lemma 3.2. Indeed, if (1) or (2) are not satisfied, one



(a) $T = T_{1,1}(t, t_q, t_r, t_p, t_p)$ (b) $T = T_{1,2}(t_r, t, t_r, t_p, t_p)$ (c) $T = T_{1,1}(t_r, t_q, t_3, t_4, t_5)$

Figure 11: Canonical trees for the proof of Lemma C.3.

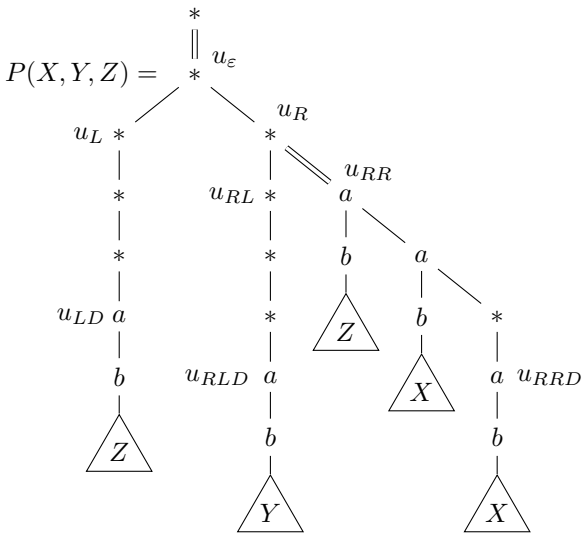


Figure 8: Gadget for constructing tree patterns (Lemma 4.3). (repeated from page 7)

can exchange p, q or r with a smaller pattern leading to a smaller, yet equivalent pattern.

We continue showing the "if"-direction. So we know that (1) and (2) are true. Towards a contradiction, assume that α is not minimal and thus there is some smaller tree pattern β such that $\alpha \equiv_S \beta$. We assume w.l.o.g. that β is minimal and therefore also strongly nonredundant. Along the proof, we will gradually restrict the possible form of β and we will finally reach a contradiction.

We first want to apply Lemma C.3 to α. To this end we first note that q is barely included in r. Indeed, if q would not be barely included in r, then we could remove a leaf n of q and obtain a pattern q' such that $p \subseteq_S q' \subseteq_S r$ and q' is smaller than q, which contradicts (2). Therefore, q is barely included in r and, by Lemma C.3 we obtain that α is strongly nonredundant.

Since $\alpha \equiv_S \beta$ and both are strongly nonredundant, we can apply Lemma B.3 and obtain that the root of pattern β has exactly one child, because α's root also has one child. Furthermore, since α's root is labeled $*$ and $\alpha \equiv_S \beta$, the root of β is also labeled $*$. From now on, we use u and v (possibly with some index) to denote nodes from α and β, respectively. We use the node names from Figure 8 for the nodes of α. Where possible, we use similar names (with u replaced by v) for corresponding nodes from β. We use v_ε to denote the unique child of root(β).

By Lemma B.8 we know that

$$\beta^{v_\varepsilon} \equiv_W \alpha^{u_\varepsilon}$$

and that

$$\alpha^{u_\varepsilon} \text{ and } \beta^{v_\varepsilon} \text{ are weakly nonredundant.}$$

Therefore we can apply Lemma B.4 to α^{u_ε} and β^{v_ε} and

obtain that the root of β^{v_ε} has exactly two children, v_L and v_R; and

$$(\diamond) \qquad \alpha^{u_\varepsilon}_{u_L} \equiv_W \beta^{v_\varepsilon}_{v_L} \text{ and } \alpha^{u_\varepsilon}_{u_R} \equiv_W \beta^{v_\varepsilon}_{v_R}.$$

Indeed, if the last condition is not the case, we can ensure it by swapping the names v_L and v_R.

To conclude the proof, we will show that

(A) $\alpha^{u_\varepsilon}_{u_L} \equiv_S \beta^{v_\varepsilon}_{v_L}$ and $\alpha^{u_L} \equiv_S \beta^{v_L}$;

(B) v_R has a child v_{RR} such that $\alpha^{u_{RR}} \equiv_S \beta^{v_{RR}}$;

(C) (v_ε, v_R) is a child edge;

(D) (v_R, v_{RR}) is a descendant edge; and

(E) v_R has a child v_{RL} which is the root of a subtree of the form $*-*-*-a-b-q'$, where $p \subseteq_S q' \subseteq_S r$.

We first argue that the statements (A) to (E) imply minimality of α. If we know that (A)–(E) hold, then we know that β has the following form: The root has one child v_ε, which has two children v_L and v_R. By (A), $\alpha^{u_L} \equiv_S \beta^{v_L}$. Therefore by the fact that r is minimal we have that $\text{size}(\alpha^{v_L}) \leq \text{size}(\beta^{u_L})$. By (B) and (E), we have that the node v_R has at least two children: v_{RL} and v_{RR}. By (B), we have that $\alpha^{u_{RR}} \equiv_S \beta^{v_{RR}}$. Since p and r are minimal, we have that $\text{size}(\alpha^{v_{RR}}) \leq \text{size}(\beta^{u_{RR}})$. Finally by (E), we have that there is a 5-descendant v_{RLDD} of v_{RL} such that $p \subseteq_S \beta^{v_{RLDD}} \subseteq_S r$. By the fact that q is a smallest pattern to fulfil $p \subseteq_S q \subseteq_S r$, we have that $\text{size}(\alpha^{v_{RL}}) \leq \text{size}(\beta^{u_{RL}})$. Summarizing we have that $\text{size}(\alpha) \leq \text{size}(\beta)$, which would contradict that β is smaller than α and shows that α is indeed minimal.

It remains to show (A) to (E). We start with (A). By $(\diamond)$ we know that $\beta^{v_\varepsilon}_{v_L} \subseteq_W \alpha^{u_\varepsilon}_{u_L}$. Therefore, by Lemma B.6, there has to be a node v_{LD} at depth 4 in $\beta^{v_\varepsilon}_{v_L}$ such that $\beta^{v_{LD}} \subseteq_W \alpha^{u_{LD}}$. We also know by $(\diamond)$ that $\alpha^{u_\varepsilon}_{u_L} \subseteq_W \beta^{v_\varepsilon}_{v_L}$ and therefore, by Lemma B.6, there has to be a node at depth 4 in $u \in \alpha^{u_\varepsilon}_{u_L}$ such that $\alpha^u \subseteq_W \beta^{v_{LD}} \subseteq_W \alpha^{u_{LD}}$. Since $\alpha^{u_\varepsilon}_{u_L}$ only has one node at depth 4, we have that $u = u_{LD}$. We therefore established that $\alpha^{u_{LD}} \equiv_W \beta^{v_{LD}}$ and, by Lemma B.5,

$$(\star) \qquad \alpha^{u_{LD}} \equiv_S \beta^{v_{LD}}.$$

Furthermore, all nodes above v_{LD} need to be wildcard nodes, because otherwise $\beta^{v_\varepsilon}_{v_L}$ would not embed into any canonical tree of $\alpha^{u_\varepsilon}_{u_L}$. To finish the proof that $\alpha^{u_\varepsilon}_{u_L} \equiv_S \beta^{v_\varepsilon}_{v_L}$, we still need to show that all edges above v_{LD} are child edges. Assume otherwise. Then — contrary to our assumption — β would be strongly redundant, because $\beta^{v_\varepsilon}_{v_R} \subseteq_W \alpha^{u_\varepsilon}_{u_R} \subseteq_W \alpha^{u_\varepsilon}_{u_L} \equiv_W \beta^{v_\varepsilon}_{v_L}$. Indeed, we have $\beta^{v_\varepsilon}_{v_R} \subseteq_W \alpha^{u_\varepsilon}_{u_R}$ by $(\diamond)$, $\alpha^{u_\varepsilon}_{u_R} \subseteq_W \alpha^{u_\varepsilon}_{u_L}$ (by definition of α and because $p \subseteq_S r$), and $\beta^{v_\varepsilon}_{v_L} \equiv_W \alpha^{u_\varepsilon}_{u_L}$ again by $(\diamond)$. Therefore, if one of the edges of above v_{LD} would be a descendant edge, β would be strongly equivalent to $\beta \setminus v_{LD}$. This concludes the proof of (A).

We continue with (B), i.e., we show that v_R has a child v_{RR} such that $\alpha^{u_{RR}} \equiv_S \beta^{v_{RR}}$. By $(\diamond)$ we know that $\alpha^{u_\varepsilon}_{u_R} \equiv_W \beta^{v_\varepsilon}_{v_R}$. By Lemma B.6, there exists a 2-descendant v_{RR} of v_ε in $\beta^{v_\varepsilon}_{v_R}$ such that $\beta^{v_{RR}} \subseteq_W \alpha^{u_{RR}}$. As v_ε has only one child in $\beta^{v_\varepsilon}_{v_R}$, this means that v_{RR} has to be a child of v_R.

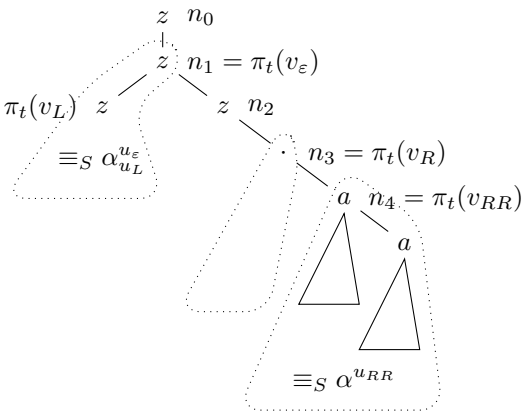


Figure 13: Tree t and embedding π_t, as used in the proof of (C)

Using Lemma B.6, once more, we also get that there is a 2-descendant u of u_ε in $\alpha^{u_\varepsilon}_{u_R}$, such that $\alpha^u \subseteq_W \beta^{v_{RR}} \subseteq_W \alpha^{u_{RR}}$. The only possibility is that $u = u_{RR}$. As $\alpha^{u_{RR}} \equiv_W \beta^{v_{RR}}$ and the root of v_{RR} is not labeled by a wildcard, we obtain by Lemma B.5 that $\alpha^{u_{RR}} \equiv_S \beta^{v_{RR}}$, which proves (B).

We continue to show (C), i.e., that (v_ε, v_R) is a child edge. We first show the following.

(†) In β there cannot be an a-labeled node which is 3-descendant of v_R and such that the path from v_R contains only child edges.

Indeed, in such a case, β^{v_ε} would not weakly embed into a canonical tree t_α of α^{u_ε} in which all the descendant edges are instantiated as paths of length 1. (Therefore, t_α can be seen as the subtree of the tree in Figure 11(c) rooted at n_ε.) We note that v_{RR} would need to embed into n_{aa} due to Observation C.1 and because $\beta^{v_{RR}} \equiv_S \alpha^{v_{RR}}$ due to (B). Therefore, v_R would need to embed into n_R (see Figure 11(c)), which means that we would have an a-labeled node 3 levels below v_R which cannot embed anywhere in t. Therefore (†) is true.

Towards a contradiction, assume that (v_ε, v_R) is a descendant edge. Then consider a tree $t \in \text{Can}(\beta)$ and an injective canonical embedding π_t of β in t such that

- $(\text{root}(\beta), v_\varepsilon)$ is mapped onto a child edge,
- (v_ε, v_R) is mapped into the path of length two,
- (v_R, v_{RR}) is mapped onto a child edge, and
- all other descendant edges are mapped onto paths of length 4.

We sketched t and π_t in Figure 13.

We will show that α does not strongly embed in t, which will contradict the fact that $\alpha \equiv_S \beta$. Notice that u_{RR} of α needs to be embedded on $\pi_t(v_{RR})$ because, since p, q and r do not use the labels a or b, it is the only place in t where an a-labeled node has an a-labeled child. In particular, u_{RR} is mapped to a node of depth 4 in t. We name this node n_4 and its ancestors on depth 0, 1, 2, and 3 we name n_0 (the root), $n_1 = \pi_t(v_\varepsilon)$, n_2,

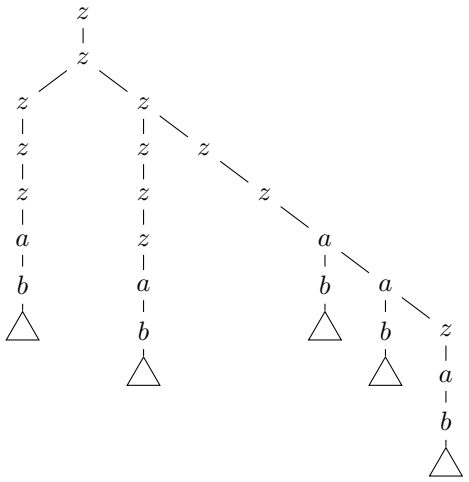


Figure 14: Tree t as used in the proof of (D).

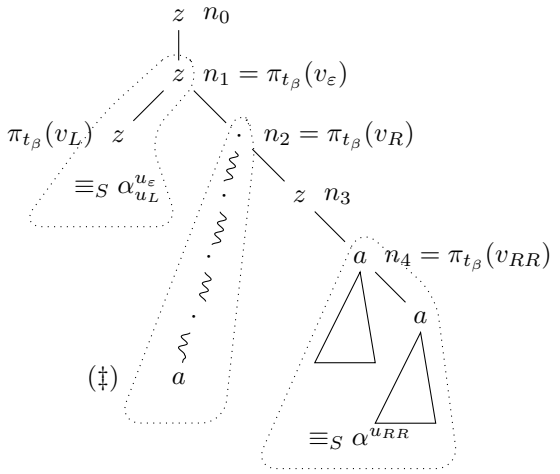


Figure 15: Sketch of a tree t_β, as used in the proof of (E)

and $n_3 = \pi_t(v_R)$. We now have two cases. The nodes u_ε and u_R are either mapped to n_1 and n_2, respectively, or to n_2 and n_3, respectively. We show that both cases are impossible.

In the case where u_ε is mapped to n_1, the node u_{RLD} has to be mapped to some a-labeled node on depth 6. In the case when u_ε is mapped to n_2, the node u_{LD} has to be mapped to some a-labeled node on depth 6. We will show however that there is no a-labeled node on depth 6 in t, ruling out both cases. In the subtree $t^{\pi_t(v_L)}$ of t corresponding to β^{v_L}, there is clearly no such node due to (A).

We now consider the other subtree, i.e., the one corresponding to β^{v_R}. Node $\pi_t(v_R)$ is on depth 3 in t. Consider an arbitrary a-labeled descendant n_a of $\pi_t(v_R)$. If n_a is in $t^{\pi_t(v_{RR})}$ we know, due to the strong equivalence in (B) that it is on depth 4, 5, or 7 since $\pi_t(v_{RR})$ is on depth 4. If n_a is not in $t^{\pi_t(v_{RR})}$ then $\pi_t^{-1}(n_a)$, which is an a-labeled node[5] in β^{v_R}, either is connected to v_R only through child edges or is *not* connected to v_R only through child edges. In the first case, n_a does not have depth 6 by (†). In the second case, the depth of n_a is at least $7 = 3 + 4$ because we mapped all descendant edges to paths of length 4. Thus there is indeed no a-labeled node on depth 6 in t. Therefore the edge (v_ε, v_R) has to be a child edge, which shows (C).

Now we show (D), that is, (v_R, v_{RR}) is a descendant edge. Assume towards a contradiction that it is a child edge. Consider a canonical tree t of α in which (u_R, u_{RR}) is mapped to a path of length 3 and all the other descendant edges are mapped onto child edges. We sketched t in Figure 14. Let π be an embedding of β into t and d be the depth of $\pi(v_\varepsilon)$. Then the node v_{LD} (which exists by $(\star)$) is mapped into a node of depth $d + 4$. We note that we can exclude the possibility of descendant edges on the path v_ε to the 3-descendant of v_L, because $\alpha_{u_L}^{u_\varepsilon} \equiv_S \beta_{v_L}^{v_\varepsilon}$ by (A).

[5]Notice that $\pi_t^{-1}(n_a)$ is well-defined because π_t is injective and because n_a has label a.

On the other hand, the a-labeled descendants of v_{RR} are mapped into nodes of depths $d+2$, $d+3$ and $d+5$. We note that $\alpha_{u_L}^{u_\varepsilon} \equiv_S \beta_{v_L}^{v_\varepsilon}$ allows us to exclude the possibility of descendant edges between v_{RR} and the aforementioned a-labeled nodes. Also, we assumed (v_R, v_{RR}) to be a child edge and (v_ε, v_R) is a child edge by (C).

Together, this means that there have to be 4 consecutive depths in t which have a-labeled nodes. However, this is not the case in the tree t which has its a-labeled nodes are on depths 5, 6 and 8. Thus indeed the assumption that (v_R, v_{RR}) is a child edge has to be wrong, which finishes the proof of (D).

We now continue with (E), i.e., we show that in β, the node v_R has a child v_{RL} which is the root of a subtree of the form $* - * - * - a - b - q'$, where $p \subseteq_S q' \subseteq_S r$.

Let $T_\beta \subseteq \mathrm{Can}(\beta)$ be the set of canonical trees of β for which that the edge between root(β) and v_ε is mapped into a child edge and the edge (v_R, v_{RR}) is mapped into a path of length 2.

Assume that

(‡) v_R has no 4-descendant v_{RLD} which is root of a subtree of the form $a - b - q'$, where $q' \subseteq_S r$.

Let t_β be a tree from T_β and π_{t_β} be a canonical, injective mapping of β into t_β such that $\alpha^{u_{LD}}$ cannot be weakly embedded in any node of $t_\beta \backslash \pi_{t_\beta}(v_{RR})$ at depth 6. By (‡) and Lemma B.1, such a canonical tree and mapping exist. We put a sketch of t_β and π_{t_β} in Figure 15. (We also depict the 4-descendant that we assume *not* to exist by (‡).)

As $\beta \subseteq_S \alpha$, there is a strong embedding of α into t_β. The node u_{RR} has to embed into $\pi_{t_\beta}(v_{RR})$ (the only a-labeled node with an a-labeled child), which is at depth 4. We call this node n_4 and its ancestors of depth i, we call n_i, similarly as in the proof of (C). Nodes u_ε and u_R have to embed either in n_1 and n_2, respectively (Case 1), or into n_2 and n_3, respectively (Case 2).

Let us first consider Case 1. Then u_R is embedded to $\pi_{t_\beta}(v_R)$. The node u_{RLD} has to embed in an a-labeled node 4 levels below $\pi_{t_\beta}(v_R)$. Consider now Case 2,

where u_ε embeds into $\pi_{t_\beta}(v_R)$. Then the node u_{LD} has to embed in an a-labeled node 4 levels below $\pi_{t_\beta}(v_R)$. So, irrespective of Case 1 or 2, we know that there must be an a-labeled node 4 levels below $\pi_{t_\beta}(v_R)$.

Clearly in $t_\beta^{\pi_{t_\beta}(v_{RR})}$, the image of the subtree $\beta^{v_{RR}}$, there is no such node, because $\pi_{t_\beta}(v_{RR})$ has no 2-descendant a-labeled node (since $\beta^{v_{RR}} \equiv_S \alpha^{u_{RR}}$ by (B)). Therefore, there has to be some other child n of $\pi_{t_\beta}(v_R)$, in which either the subtree $\alpha^{u_{RL}}$ can embed (which is Case 1) or the subtree α^{u_L} can embed (which is Case 2).

However, since $q \subseteq_S r$, we have that $\alpha_{u_{RL}}^{u_R} \subseteq_S \alpha_{u_L}^{u_\varepsilon}$. This means that, in both cases it must be possible to embed $\alpha_{u_L}^{u_\varepsilon}$ to an a-labeled node on level 4 of t_β, outside the subtree of the node v_{RR}. By (‡), such a node does not exist and therefore, α cannot be embedded into t_β, which is a contradiction to $\beta \subseteq_S \alpha$. We can conclude that (‡) is false. This means that there is indeed a 4-descendant v_{RLD} of v_R outside the subtree rooted at v_{RR} and $\beta^{v_{RLD}} \subseteq_W \alpha^{u_{LD}}$. By Lemma B.5, we furthermore know that $\beta^{v_{RLD}} \subseteq_S \alpha^{u_{LD}}$, because u_{LD} is no wildcard node.

Now we aim at showing $\alpha^{u_{RRD}} \subseteq_S \beta^{v_{RLD}}$. We know by Lemma B.6 that there has to be a node u on depth 6 in α such that $\alpha^u \subseteq_W \beta^{v_{RLD}}$. As there are only two a-labeled nodes on depth 6 in α, we either have that $u = u_{RLD}$ or $u = u_{RRD}$. As furthermore $\alpha^{u_{RRD}} \subseteq_S \alpha^{u_{RLD}}$, we can conclude that $\alpha^{u_{RRD}} \subseteq_W \beta^{v_{RLD}}$. By Lemma B.5, since u_{RRD} is not a wildcard node, we have that $\alpha^{u_{RRD}} \subseteq_S \beta^{v_{RLD}}$.

Summarizing, we have that $\alpha^{u_{RRD}} \subseteq_S \beta^{v_{RLD}} \subseteq_S \alpha^{u_{LD}}$, which implies that v_{RLD} is the root of a subtree of the form $a - b - q'$ with $p \subseteq_S q' \subseteq_S r$. Furthermore, v_{RLD} is a 4-descendant of v_R and therefore not in the subtree $\beta^{v_{RR}}$. This implies that v_R has a child v_{RL} different from v_{RR}, such that v_{RLD} is a 3-descendant of v_{RL}. Putting everything together, v_{RL} is the root of a subtree of the form $*-*-*-a-b-q'$ with $p \subseteq_S q' \subseteq_S r$. We note that the edges between v_R and v_{RLD} cannot be descendant edges, as otherwise $\beta_{v_{RR}}^{v_R} \subseteq_W \beta_{v_{RL}}^{v_R}$ and therefore the branch would be redundant. This finishes the proof of (E) and the lemma. □

C.2 Proof of Lemma 4.5

We first prove the two claims in the proof of Lemma 4.5. We then continue with the proof of the lemma. The beginning of the proof is already in the body of the paper and is omitted here. We denote this by $\boxed{\cdots}$.

Claim 4.6: *The tree patterns p and r are minimal.*

Proof. As p and r both are $*$-narrow, by Lemma 2.3, it is enough to show that both patterns are non-redundant. It is easy (but tedious) to verify that removing any node from any canonical tree of p results in a tree that is not in $L(p)$. The same holds for canonical trees of q and $L(q)$. We conclude by Lemma 2.4. This concludes the proof of Claim 4.6. □

Claim 4.7: *Every minimal pattern q that satisfies $p \subseteq_S q \subseteq_S r$ is in normal form.*

Proof. Let q be a smallest tree pattern such that $p \subseteq_S q \subseteq_S r$. Let t_q be the smallest canonical tree of q and π be an embedding of r into t_q.

We first show that π is surjective. Assume otherwise, then we can construct a tree pattern q' such that $p \subseteq_S q' \subseteq_S r$ and $\text{size}(q') < \text{size}(q)$ by removing a node of t_q not used by π, relabeling every z-node as wildcard node, and changing any edge $(\pi(u), \pi(v))$ of t_q such that (u, v) is a descendant edge in r into a descendant edge. We note that there are no nodes u, v and w in r such that all of the following hold:

- v is a descendant of u;
- (u, w) is a descendant edge;
- v and w have the same label or one of the nodes is a wildcard; and
- v and w are in different subtrees, i.e., v is not a descendant of w or vice versa.

If for two nodes v and w of r we have that $\pi(v) = \pi(w)$, then both nodes are e-nodes and siblings. All other possibilities are easily excluded by looking at the relative level of nodes in the pattern and ancestor-descendant relationships. Furthermore, in any b-subpattern, there is at most one pair of e-siblings, such that $\pi(v) = \pi(w)$, as otherwise $p \not\subseteq_S q$. □

Lemma 4.5: Relative Tree Pattern Minimization *is Σ_2^P-complete.*

Proof. $\boxed{\cdots}$ We start with (a). Let t be a canonical tree of p, let π be an embedding of p in t, and ρ be an embedding of q in t. Such an embedding exists due to Lemma 2.4 and since $p \subseteq_S q$.

We denote the subtree of t corresponding to the subpattern D of p by t_D.

As there is a path of m a-labeled nodes in q (and therefore in t) and a path of $2m-1$ a-labeled nodes in p, one a-labeled node u_j of q has to be embedded on the middle a-labeled node of t.

We show that the clause $c_j^{\mathrm{f}^q(j)}$ is satisfied by $\sigma(t)$, which directly shows (a). To achieve this goal it is sufficient to look how the subpattern rooted at u_j is embedded on the subtree rooted at v_{target}.

As there is only one e-node in the a_m-subtree which has both an f-labeled and a g-labeled child, we know that ρ needs to embed $C_j^{\mathrm{f}^q(j)}$ on t_D. We show that every literal of $c_j^{\mathrm{f}^q(j)}$ is satisfied by $\sigma(t)$.

Let x_i be a positive literal of $c_j^{\mathrm{f}^q(j)}$. Then there are two x_i-labeled nodes in $C_j^{\mathrm{f}^q(j)}$, connected by a child edge. These nodes have to be embedded on the x_i-labeled nodes of t_D. Therefore $d(x_i) = 1$ and $\sigma(t)(x_i) = \text{true}$.

Now let $\neg x_i$ be a negative literal of $c_j^{\mathrm{f}^q(j)}$. Then there are two x_i-labeled nodes in $C_j^{\mathrm{f}^q(j)}$, connected by a path of length at least two, i.e., a path consisting a one child-edge and one descendant-edge. Again, these nodes have to be embedded on the x_i-labeled nodes of t_D. Therefore $d(x_i) > 1$ and $\sigma(t)(x_i) = \text{false}$. This concludes the proof of (a).

Concerning (b), let q be the pattern according to statement (b). We first observe that, as q is in nor-

mal form, $q \subseteq_S r$ holds. In this case q results from r by merging some sibling e-labeled nodes, which restricts the trees in the language, as the f-labeled nodes and g-labeled nodes (from the patterns C_i^j) need to have the same parent.

It remains to show that $p \subseteq_S q$ holds, which we do by giving an embedding π of q for every canonical tree t of p. Let therefore t be an arbitrary canonical tree of p. Let i be such that the clause $c_i^{\iota(i)}$ is satisfied by $\sigma(t)$. As $c_1^{\iota(1)} \vee \cdots \vee c_m^{\iota(m)}$ is universally true, such a clause has to exist.

We define π such that

- the root of q is embedded at the root of t;
- the i-th a-labeled node of q is embedded at a_m;[6]
- for any subpattern of q rooted at an a-labeled node, the c-labeled nodes are embedded such that the c-labeled node with only one child is embedded at the c-labeled node that has only one child of the corresponding subtree of t.

From this point on, the embedding of the d-, e-, and f-labeled nodes is obvious, as there is no choice. Furthermore, any C_i^j-subpattern can be embedded in one of the copies of C. It remains to show that $C_i^{\iota(i)}$ can be embedded in the part of t_D of t corresponding to D.

If there exists x_i-labeled nodes below g in $C_i^{\iota(i)}$ with a distance of one, then $x_i \in c_i^{\iota(i)}$. Therefore, $\sigma(t)(x_i)$ has to be true (as $c_i^{\iota(i)}$ is satisfied) and we can conclude that $d(x_i) = 1$ by definition of $\sigma(t)$. As $d(x_i) = 1$, we can embed the two x_i-labeled nodes in t_D.

If on the other hand, there exists x_i nodes below g in $C_i^{\iota(i)}$ with a distance of at least two, then $\neg x_i \in c_i^{\iota(i)}$. Therefore, $\sigma(t)(x_i)$ has to be false and we can conclude that $d(x_i) > 1$ by definition of $\sigma(t)$. As $d(x_i) > 1$, we again can embed the two x_i-labeled nodes in t_D (including the wildcard node in between both nodes).

From the construction, it is clear that π is a valid embedding. This concludes the proof of (b).

This concludes the proof of Lemma 4.5. $\square$

[6] In other words: the first a-labeled node of q is embedded at the $m - i + 1$-th a-labeled node of t.