EXISTENTIAL DEFINABILITY OVER THE SUBWORD ORDERING

PASCAL BAUMANN ⬤, MOSES GANARDI ⬤, RAMANATHAN S. THINNIYAM ⬤,
AND GEORG ZETZSCHE ⬤

Max Planck Institute for Software Systems (MPI-SWS), Kaiserslautern, Germany
e-mail address: {pbaumann, ganardi, thinniyam, georg}@mpi-sws.org

ABSTRACT. We study first-order logic (FO) over the structure consisting of finite words over some alphabet A, together with the (non-contiguous) subword ordering. In terms of decidability of quantifier alternation fragments, this logic is well-understood: If every word is available as a constant, then even the Σ_1 (i.e., existential) fragment is undecidable, already for binary alphabets A.

However, up to now, little is known about the expressiveness of the quantifier alternation fragments: For example, the undecidability proof for the existential fragment relies on Diophantine equations and only shows that recursively enumerable languages over a singleton alphabet (and some auxiliary predicates) are definable.

We show that if $|A| \geq 3$, then a relation is definable in the existential fragment over A with constants if and only if it is recursively enumerable. This implies characterizations for all fragments Σ_i: If $|A| \geq 3$, then a relation is definable in Σ_i if and only if it belongs to the i-th level of the arithmetical hierarchy. In addition, our result yields an analogous complete description of the Σ_i-fragments for $i \geq 2$ of the *pure logic*, where the words of A^* are not available as constants.

1. INTRODUCTION

The subword ordering. A word u is a *subword* of another word v if u can be obtained from v by deleting letters at an arbitrary set of positions. The subword ordering has been studied intensively over the last few decades. On the one hand, it appears in many classical results of theoretical computer science. For example, subwords have been a central topic in string algorithms [BY91, ERW08, Mai78]. Moreover, their combinatorial properties are the basis for verifying lossy channel systems [AJ96]. Particularly in recent years, subwords have received a considerable amount of attention. Notable examples include lower bounds in fine-grained complexity [BK15, BK18], algorithms to compute the set of all subwords of formal languages [ACH+16, AMMS17, BCCP20, CPSW16, HMW10, HKO16, Zet15a, Zet15b, Zet16, Zet18, GLHK+20, AZ23], and applications thereof to infinite-state verification [ABQ11, TMW15, BMTZ22, MTZ22, BMTZ20, BGM+23b, BGM+23a]. Subwords are also the basis of Simon's congruence [SS97], which has recently been studied from algorithmic [FK18, GKK+21, FKK+23] and combinatorial [BFH+20, DFK+21, KKS15, KS19, SV23] viewpoints.

First-order logic over subwords. The importance of subwords has motivated the study of first-order logics (FO) over the subword ordering. This has been considered in two variants: In the *pure logic*, one has FO over the structure $(A^*, \preccurlyeq)$, where A is an alphabet and $\preccurlyeq$ is the subword ordering. In the version *with constants*, we have the structure $(A^*, \preccurlyeq, (w)_{w \in A^*})$, which has a constant for each word from A^*. Traditionally for FO, the primary questions are *decidability* and *definability*, particularly regarding quantifier alternation fragments Σ_i. Here, decidability refers to the *truth problem*: Given a formula φ in a particular fragment over $(A^*, \preccurlyeq)$ or $(A^*, \preccurlyeq, (w)_{w \in A^*})$, respectively, does φ hold? By definability, we mean understanding which relations can be defined by formulas in a particular fragment. The Σ_i-fragment consists of formulas in prenex form that begin with existential quantifiers and then alternate $i - 1$ times between blocks of universal and existential quantifiers. For example, the formula

$$\exists x \colon (a \not\preccurlyeq x \ \lor \ b \not\preccurlyeq x) \ \land \ x \not\preccurlyeq u \ \land \ x \preccurlyeq v$$

belongs to the Σ_1-fragment, also called the *existential fragment* over $(A^*, \preccurlyeq, (w)_{w \in A^*})$ with $A = \{a, b\}$. The formula has free variables u, v and refers to the constants a and b. It holds if and only if v has more b's or more a's than u.

For FO over subwords, decidability is well-understood. In the pure logic, the Σ_2-fragment is undecidable, already over two letters [HSZ17, Corollary III.6], whereas the Σ_1-fragment (i.e., existential formulas) is decidable [Kus06, Theorem 2.2] and NP-complete [KS15, Theorem 2.1]. This fueled hope that the Σ_1-fragment might even be decidable with constants, but this turned out to be undecidable, already over two letters [HSZ17, Theorem III.3]. Decidability (and complexity) have also been studied for the two-variable fragment [KS15, KS19, KS20], and extended with counting quantifiers and regular predicates [KS20, KZ19].

Nevertheless, little is known about definability. Kudinov, Selivanov, and Yartseva have shown that using arbitrary first-order formulas over $(A^*, \preccurlyeq)$, one can define exactly the relations from the arithmetical hierarchy[1] that are invariant under automorphisms of $(A^*, \preccurlyeq)$ [KSY10, Theorem 5], if $|A| \geq 2$. However, this does not explain definability of the Σ_i-fragments. For example, in order to define all recursively enumerable languages, as far as we can see, their proof requires several quantifier alternations. An undecidability proof by Karandikar and Schnoebelen [KS15, Theorem 4.6] for the Σ_2-fragment can easily be adapted to show that for each alphabet A, there exists a larger alphabet B such that every recursively enumerable language $L \subseteq A^*$ is definable in the Σ_2-fragment over $(B^*, \preccurlyeq, (w)_{w \in B^*})$. However, a full description of the expressiveness of the Σ_2-fragment is missing.

Existential formulas. The expressiveness of existential formulas is even further from being understood. The undecidability proof in [HSZ17] reduces from solvability of Diophantine equations, i.e., polynomial equations over integers, which is a well-known undecidable problem [Mat93]. To this end, it is shown in [HSZ17] that the relations $\mathsf{ADD} = \{(a^m, a^n, a^{m+n}) \mid m, n \in \mathbb{N}\}$ and $\mathsf{MULT} = \{(a^m, a^n, a^{m \cdot n}) \mid m, n \in \mathbb{N}\}$ are definable existentially using the subword ordering, if one has at least two letters. Since Diophantine equations can be used to define all recursively enumerable relations over natural numbers, this implies that all recursively enumerable relations involving a single letter are definable existentially. However, this says little about which languages (let alone relations) over more than one letter are definable. For example, it is not clear whether the language of all $w \in \{a, b\}^*$ that do *not* contain *aba*

---

[1] Also known as the Kleene–Mostowski hierarchy

as an infix, or the reversal relation $\mathsf{REV}_A = \{(u,v) \mid u,v \in A^*,\ v \text{ is the reversal of } u\}$, are definable—it seems particularly difficult to define them over the subword ordering using the methods from [HSZ17].

Contribution. We show that for any alphabet A with $|A| \geq 3$, every recursively enumerable relation $R \subseteq (A^*)^k$, $k \in \mathbb{N}$, is existentially definable in $(A^*, \preccurlyeq, (w)_{w \in A^*})$. In fact, similarly to an observation made in [HSZ17], we even show that there is a single sufficiently complex word $W \in A^*$ such that the structure $(A^*, \preccurlyeq, W)$ with just this single constant symbol suffices to existentially define all recursively enumerable relations. Since every existentially definable relation is clearly recursively enumerable (via a simple enumerative algorithm), this completely describes the expressiveness of existential formulas for $|A| \geq 3$. Despite the undecidability of the existential fragment [HSZ17], we find it surprising that all recursively enumerable relations—including relations like REV_A—are existentially definable.

Our result yields characterizations of the Σ_i-fragments for every $i \geq 2$: It implies that for each $i \geq 2$, the Σ_i-fragment over $(A^*, \preccurlyeq, (w)_{w \in A^*})$ can define exactly the relations in Σ_i^0, the i-th level of the arithmetical hierarchy, assuming $|A| \geq 3$. This also provides a description of Σ_i in the pure logic: It follows that in the Σ_i-fragment over $(A^*, \preccurlyeq)$, one can define exactly the relations in Σ_i^0 that are invariant under automorphisms of $(A^*, \preccurlyeq)$, if $|A| \geq 3$.

Since [HSZ17] shows that all recursively enumerable languages over one letter are definable in $(A^*, \preccurlyeq, (w)_{w \in A^*})$ if $|A| \geq 2$, it would suffice to define a bijection between a^* and A^* using subwords. However, since this seems hard to do directly, our proof follows a different route. We first show how to define rational transductions and then a special language from which one can build every recursively enumerable relation via rational transductions and intersections. In particular, a byproduct is a direct proof of undecidability of the existential fragment in the case of $|A| \geq 3$ that avoids using undecidability of Diophantine equations[2].

Key ingredients. The undecidability proof for the existential fragment from [HSZ17] shows that the relations ADD and MULT are definable, in addition to auxiliary predicates that are needed for this, such as concatenation and letter counting predicates of the form "$|u|_a = |v|_b$". With these methods, it is difficult to express that a certain property holds locally—by which we mean: at every position in a word. Using concatenation, we can define languages like $(a^n b)^*$ for each $n \in \mathbb{N}$ (see Section 3), which "locally look like $a^n b$". But if we want to express that, e.g., aba does not occur as an infix, this is of little help, because words avoiding an infix need not be periodic. The ability to disallow infixes would aid us in defining rational transductions via runs of transducers, as these are little more than configuration sequences where pairs of configurations that are not connected by a transition do not occur as infixes. Such local properties are often easy to state with universal quantification, but this is not available in existential formulas.

An important theme in our proof is to express such local properties by carefully constructing long words in which w has to embed in order for w to have the local property. For example, our first lemma says: Each set $X \subseteq A^{=\ell}$ can be characterized as the set of words (of length $\geq \ell$) that embed into each word in a finite set P. This allows us to define sets X^*.

---

[2]Our proof relies on the definability of concatenation and certain counting predicates (see Section 3), which was shown directly in [HSZ17], without using computational completeness of Diophantine equations.

Steps I–III of our proof use techniques of this type to express rational transductions. In Step IV, we then define the special language $G = \{a^n b^n \mid n \geq 0\}^*$, which has the property that all recursively enumerable languages can be obtained from G using rational transductions and intersection. This yields all recursively enumerable relations over two letters in Step V.

In sum, Steps I–V let us define all recursively enumerable relations over $\{a, b\}$, provided that the alphabet A contains an additional auxiliary letter. It then remains to define recursively enumerable relations that can also involve all other letters in A. We do this in Step VI by observing that each word $w \in A^*$ is determined by its projections to binary alphabets $B \subseteq A$. This allows us to compare words by looking at two letters at a time and use the other (currently unused) letters for auxiliary means.

A conference version of this paper appeared in [BGTZ22].

2. Main results

We say that u is a *subword* of v, written $u \preccurlyeq v$, if there exist words $u_1, \ldots, u_n$ and $v_0, \ldots, v_n$ such that $u = u_1 \cdots u_n$ and $v = v_0 u_1 v_1 \cdots u_n v_n$.

Subword logic. We consider first-order logic over the structure $(A^*, \preccurlyeq)$, first-order logic over the structure $(A^*, \preccurlyeq, w_1, \ldots, w_n)$ enriched with finitely many constant symbols $w_1, \ldots, w_n \in A^*$, and first-order logic over the structure $(A^*, \preccurlyeq, (w)_{w \in A^*})$ enriched with constant symbols w for every word $w \in A^*$. A first-order formula φ with free variables $x_1, \ldots, x_k$ *defines* a relation $R \subseteq (A^*)^k$ if R contains exactly those tuples of words $(v_1, \ldots, v_k)$ that satisfy[3] the formula φ.

Let us define the quantifier alternation fragments of first-order logic. A formula without quantifiers is called Σ_0-*formula* or Π_0-*formula*. For $i \geq 1$, a Σ_i-formula (resp. Π_i-formula) is one of the form $\exists x_1 \cdots \exists x_n \varphi$ (resp. $\forall x_1 \cdots \forall x_n \varphi$), where φ is a a Π_{i-1}-formula (resp. Σ_{i-1}-formula), $x_1, \ldots, x_n$ are variables, and $n \geq 0$. In other words, a Σ_i-formula is in prenex form and its quantifiers begin with a block of existential quantifiers and alternate at most $i-1$ times between universal and existential quantifiers. The Σ_i-*fragment* (Π_i-*fragment*) consists of the Σ_i-formulas (Π_i-formulas). In particular, the Σ_1-fragment (called the *existential fragment*) consists of the formulas in prenex form that only contain existential quantifiers.

Expressiveness with constants. Our main technical contribution is the following.

Theorem 2.1. *Let A be an alphabet with $|A| \geq 3$. A relation is definable in the Σ_1-fragment over $(A^*, \preccurlyeq, (w)_{w \in A^*})$ if and only if it is recursively enumerable.*

We prove Theorem 2.1 in Section 3. Theorem 2.1 in particular yields a description of what is expressible using Σ_i-formulas for each $i \geq 1$. Recall that the *arithmetical hierarchy* consists of classes $\Sigma_1^0, \Sigma_2^0, \ldots$, where $\Sigma_1^0 = \mathsf{RE}$ is the class of recursively enumerable relations, and for $i \geq 2$, we have $\Sigma_i^0 = \mathsf{RE}^{\Sigma_{i-1}^0}$. Here, for a class of relations $\mathcal{C}$, $\mathsf{RE}^{\mathcal{C}}$ denotes the class of relations recognized by oracle Turing machines with access to oracles over the class $\mathcal{C}$.

---

[3]The correspondence between the entries in the tuple and the free variables of φ will always be clear, because the variables will have an obvious linear order by sorting them alphabetically and by their index. For example, if φ has free variables x_i for $1 \leq i \leq k$ and y_j for $1 \leq j \leq \ell$, then we order them as $x_1, \ldots, x_k, y_1, \ldots, y_\ell$.

Corollary 2.2. *Let A be an alphabet with $|A| \geq 3$ and let $i \geq 1$. A relation is definable in the Σ_i-fragment over $(A^*, \preccurlyeq, (w)_{w \in A^*})$ if and only if it belongs to Σ_i^0.*

By [HSZ21, Theorem 3.5] the undecidability of the Σ_1-fragment already holds for $(A^*, \preccurlyeq, W)$ where $W \in A^*$ is a sufficiently complex constant. Using the same ideas we show that the characterizations from Theorem 2.1 and Corollary 2.2 also already hold for a single constant, which will be proven in Section 4.

Remark 2.3. Let $|A| \geq 3$. There exists a word $W \in A^*$ so that Theorem 2.1 and Corollary 2.2 still hold for $(A^*, \preccurlyeq, W)$ instead of $(A^*, \preccurlyeq, (w)_{w \in A^*})$.

Expressiveness of the pure logic. Corollary 2.2 completely describes the relations definable in the structure $(A^*, \preccurlyeq, (w)_{w \in A^*})$ if $|A| \geq 3$. We can use this to derive a description of the relations definable without constants, i.e., in the structure $(A^*, \preccurlyeq)$. The lack of constants slightly reduces the expressiveness; to make this precise, we need some terminology. An *automorphism (of $(A^*, \preccurlyeq)$)* is a bijection $\alpha \colon A^* \to A^*$ such that $u \preccurlyeq v$ if and only if $\alpha(u) \preccurlyeq \alpha(v)$. A relation $R \subseteq (A^*)^k$ is *automorphism-invariant* if for every automorphism α, we have $(v_1, \ldots, v_k) \in R$ if and only if $(\alpha(v_1), \ldots, \alpha(v_k)) \in R$. It is straightforward to check that every formula over $(A^*, \preccurlyeq)$ defines an automorphism-invariant relation. Thus, in the Σ_i-fragment over $(A^*, \preccurlyeq)$, we can only define automorphism-invariant relations inside Σ_i^0.

Corollary 2.4. *Let A be an alphabet with $|A| \geq 3$ and let $i \geq 2$. A relation is definable in the Σ_i-fragment over $(A^*, \preccurlyeq)$ if and only if it is automorphism-invariant and belongs to Σ_i^0.*

To give some intuition on automorphism-invariant sets, let us recall the classification of automorphisms of $(A^*, \preccurlyeq)$, shown implicitly by Kudinov, Selivanov, and Yartseva in [KSY10] (for a short and explicit proof, see [HSZ21, Lemma 3.8]): A map $\alpha \colon A^* \to A^*$ is an automorphism of $(A^*, \preccurlyeq)$ if and only if (i) the restriction of α to A is a permutation of A, and (ii) α is either a *word morphism*, i.e., $\alpha(a_1 \cdots a_k) = \alpha(a_1) \cdots \alpha(a_k)$ for any $a_1, \ldots, a_k \in A$, or a *word anti-morphism*, i.e., $\alpha(a_1 \cdots a_k) = \alpha(a_k) \cdots \alpha(a_1)$ for any $a_1, \ldots, a_k \in A$.

Finally, Corollary 2.4 raises the question of whether the Σ_1-fragment over $(A^*, \preccurlyeq)$ also expresses exactly the automorphism-invariant recursively enumerable relations. It does not:

Observation 2.5. Let $|A| \geq 2$. There are undecidable binary relations definable in the Σ_1-fragment over $(A^*, \preccurlyeq)$. However, not every automorphism-invariant regular language is definable in it.

3. Existentially defining recursively enumerable relations

In this section, we prove Theorem 2.1. Therefore, we now concentrate on definability in the Σ_1-fragment. Moreover, for an alphabet A, we will sometimes use the phrase Σ_1-*definable over A* as a shorthand for definability in the Σ_1-fragment over the structure $(A^*, \preccurlyeq, (w)_{w \in A^*})$.

Notation. For an alphabet A, we write $A^{=k}$, $A^{\geq k}$, and $A^{\leq k}$ for the set of words over A that have length exactly k, at least k, and at most k, respectively. We write $|w|$ for the length of a word w. If $B \subseteq A$ is a subalphabet of A then $|w|_B$ denotes the number of occurrences of letters $a \in B$ in w, or simply $|w|_a$ if $B = \{a\}$ is a singleton. Furthermore, we write $\pi_B \colon A^* \to B^*$ for the projection morphism which keeps only the letters from B. If $B = \{a, b\}$, we also write $\pi_{a,b}$ for $\pi_{\{a,b\}}$. The *downward closure* of a word $v \in A^*$ is defined as $v{\downarrow} := \{u \in A^* \mid u \preccurlyeq v\}$.

Basic relations. We will use two kinds of relations, concatenation and counting letters, which are shown to be Σ_1-definable in $(A^*, \preccurlyeq, (w)_{w \in A^*})$ as part of the undecidability proof of the truth problem in [HSZ17, Theorem III.3]. The following relations are Σ_1-definable if $|A| \geq 2$.

Concatenation: The relation $\{(u, v, w) \in (A^*)^3 \mid w = uv\}$.

Counting letters: The relation $\{(u, v) \in (A^*)^2 \mid |u|_a = |v|_b\}$ for any $a, b \in A$.

Moreover, we will make use of a classical fact from word combinatorics: For $u, v \in A^*$, we have $uv = vu$ if and only if there is a word $r \in A^*$ with $u \in r^*$ and $v \in r^*$ [Ber79]. In particular, if p is *primitive*, meaning that $p \in A^+$ and there is no $r \in A^*$ with $|r| < |p|$ and $p \in r^*$, then $up = pu$ is equivalent to $u \in p^*$. Furthermore, note that by counting letters as above, and using concatenation, we can also say $|u|_a = |vw|_a$, i.e., $|u|_a = |v|_a + |w|_a$ for $a \in A$. With these building blocks, we can state arbitrary linear equations over terms $|u|_a$ with $u \in A^*$ and $a \in A$. For example, we can say $|u| = 3 \cdot |v|_a + 2 \cdot |w|_b$ for $u, v, w \in A^*$ and $a, b \in A$. This also allows us to state modulo constraints, such as $\exists v \colon |u|_a = 2 \cdot |v|_a$, i.e., "$|u|_a$ is even". Finally, counting letters lets us define projections: Note that for $B \subseteq A$ and $u, v \in A^*$, we have $v = \pi_B(u)$ if and only if $v \preccurlyeq u$ and $|v|_b = |u|_b$ for each $b \in B$ as well as $\neg(a \preccurlyeq v)$ for every $a \in A \setminus B$.

For any subalphabet $B \subseteq A$ one can clearly define B^* over A. Hence definability of a relation over B also implies definability over the larger alphabet A.

Finite state transducers. An important ingredient of our proof is to define regular languages in the subword order, and, more generally, rational transductions, i.e., relations recognized by finite state transducers.

For $k \in \mathbb{N}$, a *k-ary finite state transducer* $\mathcal{T} = (Q, A, \delta, q_0, Q_f)$ consists of a finite set of *states* Q, an input alphabet A, an *initial state* $q_0 \in Q$, a set of *final states* $Q_f \subseteq Q$, and a *transition relation* $\delta \subseteq Q \times (A \cup \{\varepsilon\})^k \times Q$. For a *transition* $(q, a_1, \ldots, a_k, q') \in \delta$, we also write $q \xrightarrow{(a_1, \ldots, a_k)} q'$.

The transducer $\mathcal{T}$ *recognizes* the k-ary relation $R(\mathcal{T}) \subseteq (A^*)^k$ containing precisely those k-tuples $(w_1, \ldots, w_k)$, for which there is a transition sequence $q_0 \xrightarrow{(a_{1,1}, \ldots, a_{k,1})} q_1 \xrightarrow{(a_{1,2}, \ldots, a_{k,2})} \ldots \xrightarrow{(a_{1,m}, \ldots, a_{k,m})} q_m$ with $q_m \in Q_f$ and $w_i = a_{i,1} a_{i,2} \cdots a_{i,m}$ for all $i \in \{1, \ldots, k\}$. Such a transition sequence is called an *accepting run* of $\mathcal{T}$. We sometimes prefer to think of the w_i as *produced output* rather than *consumed input* and thus occasionally use terminology accordingly. A relation T is called a *rational transduction* if it is recognized by some finite state transducer $\mathcal{T}$. Unary transducers (i.e., $k = 1$) recognize the *regular languages*.

Overview. As outlined in the introduction, our proof consists of six steps. In Steps I–III, we show that we can define all rational transductions $T \subseteq (A^*)^k$ over the alphabet B, if $|B| \geq |A| + 1$. In Step IV, we define the special language $G = \{a^n b^n \mid n \geq 0\}^*$. From G, all recursively enumerable languages can be obtained using rational transductions and intersection, which in Step V allows us to define over B all recursively enumerable relations over A, provided that $|B| \geq |A| + 1$. Finally, in Step VI, we use projections to binary alphabets to define arbitrary recursively enumerable relations over A, if $|A| \geq 3$.

Step I: Defining Kleene stars. We first define the languages X^*, where X consists of words of equal length. To this end, we establish an alternative representation for such sets.

Example 3.1. Before proving the general statement, let us illustrate how to define the language $\{ab, ba\}^*$ using an auxiliary symbol $\#$. It suffices to define the language $\{ab\#, ba\#\}^*$ and then project to $\{a, b\}$. The simple but key observation is that

$$u \in \{ab, ba\} \iff u \preccurlyeq bab \text{ and } u \preccurlyeq aba \text{ and } |u| \geq 2. \tag{3.1}$$

We claim that a word w belongs to $\{ab\#, ba\#\}^*$ if and only if

$$\exists n \in \mathbb{N} \colon w \preccurlyeq (aba\#)^n \wedge w \preccurlyeq (bab\#)^n \wedge |w|_\# = n \wedge |w| = 3n. \tag{3.2}$$

The "only if"-direction is immediate. For the "if"-direction consider a word w satisfying (3.2), i.e. $w = w_1\# \cdots w_n\#$ where each w_i belongs to $\{a, b\}^*$. Then each word w_i is a subword of aba and bab. Since $|w| = 3n$ either all words w_i have length 2 and therefore $w_i \in \{ab, ba\}$ by (3.1); or, there exists some w_i with $|w_i| > 2$. But then again $w_i \in \{ab, ba\}$ by (3.1), contradicting $|w_i| > 2$.

Lemma 3.2. *Every nonempty set $X \subseteq A^{=\ell}$ can be written as $X = A^{\geq \ell} \cap \bigcap_{p \in P} p\!\downarrow$ for some finite set $P \subseteq A^*$.*

Proof. We can assume $\ell \geq 1$ since otherwise $X = \{\varepsilon\} = A^{\geq 0} \cap \varepsilon\!\downarrow$. Let $w \in A^*$ be any permutation of A (i.e., each letter of A appears exactly once in w). If $a \in A$, then $(w \setminus a)$ denotes the word obtained from w by deleting a. For any nonempty word $u = a_1 \cdots a_k \in A^+$, $a_1, \ldots, a_k \in A$, define the word

$$p_u = (w \setminus a_1)(w \setminus a_1)\, a_1\, (w \setminus a_2)(w \setminus a_2)\, a_2 \cdots (w \setminus a_{k-1})(w \setminus a_{k-1})\, a_{k-1}\, (w \setminus a_k)(w \setminus a_k).$$

Note that p_u does not contain u as a subword: In trying to embed each letter a_i of u into p_u, the first possible choice for a_1 comes after the initial sequence $(w \setminus a_1)(w \setminus a_1)$. Similarly, the next possible choice for each subsequent a_i is right after $(w \setminus a_i)(w \setminus a_i)$. However, this only works until a_{k-1}, since there is no a_k at the end of p_u.

On the other hand, observe that p_u contains every word $v \in A^{\leq k} \setminus \{u\}$ as a subword: Suppose that $v = b_1 \cdots b_m$, $b_1, \ldots, b_m \in A$, and let $i \in [1, m+1]$ be the minimal position with $b_i \neq a_i$ or $i = m+1$. The prefix $b_1 \cdots b_{i-1} = a_1 \cdots a_{i-1}$ occurs as a subword of p_u, which in the case $i = m+1$ already is the whole word v. If $i \leq m$ then b_i occurs in $(w \setminus a_i)$, and $b_{i+1} \cdots b_m$ embeds into the subword $(w \setminus a_i)\, a_i \cdots (w \setminus a_{k-1})\, a_{k-1}$ of p_u. Thus, we can write

$$X = A^{\geq \ell} \cap \bigcap_{u \in (A^{=\ell} \setminus X) \cup A^{=\ell+1}} p_u\!\downarrow.$$

Here $u \in A^{=\ell+1}$ was added to also exclude all words of length greater than ℓ. □

Lemma 3.3. *Let $A \subseteq B$ be finite alphabets and $\# \in B \setminus A$. Let $X \subseteq A^{=k}$ and $Y \subseteq A^{=\ell}$ be sets. Then $(X\#Y\#)^*$ and X^* are Σ_1-definable over B.*

Proof. We can clearly assume that X, Y are nonempty. By Lemma 3.2 we can write $X = A^{\geq k} \cap \bigcap_{p \in P} p{\downarrow}$ and $Y = A^{\geq \ell} \cap \bigcap_{q \in Q} q{\downarrow}$ for some finite sets $P, Q \subseteq A^*$. We claim that $w \in (A \cup \{\#\})^*$ belongs to $(X\#Y\#)^*$ if and only if

$$\exists n \in \mathbb{N}: |w|_\# = 2n \land |w|_A = (k + \ell) \cdot n \land \bigwedge_{p \in P, q \in Q} w \preccurlyeq (p\#q\#)^n. \tag{3.3}$$

Observe that the number n is uniquely determined by $|w|_\#$. The "only if"-direction is clear. Conversely, suppose that $w \in (A \cup \{\#\})^*$ satisfies the formula. We can factorize $w = x_1\#y_1\# \ldots x_n\#y_n\#$ where each x_i is a subword of each word $p \in P$, and each y_i is a subword of each word $q \in Q$. If some word x_i were strictly longer than k, then it would belong to X by the representation of X, and in particular would have length k, contradiction. Therefore each word x_i has length at most k, and similarly each word y_i has length at most ℓ. However, since the total length of $x_1 y_1 \ldots x_n y_n$ is $(k + \ell) \cdot n$, we must have $|x_i| = k$ and $|y_i| = \ell$, and hence $x_i \in X$ and $y_i \in Y$ for all $i \in [1, n]$. This proves our claim.

Finally, (3.3) is equivalent to the following Σ_1-formula:

$$(k + \ell) \cdot |w|_\# = 2 \cdot |w|_A \;\land\; \bigwedge_{p \in P, q \in Q} \exists u \in (p\#q\#)^*: (w \preccurlyeq u \;\land\; |u|_\# = |w|_\#)$$

Here, we express $u \in (p\#q\#)^*$ as follows. If $p \neq q$, then $p\#q\#$ is primitive and $u \in (p\#q\#)^*$ is equivalent to $u(p\#q\#) = (p\#q\#)u$. If $p = q$, then $u \in (p\#q\#)^*$ is equivalent to $up\# = p\#u$ and $|u|_\#$ being even. Finally, to define X^* we set $Y = \{\varepsilon\}$ and obtain $X^* = \pi_A((X\#Y\#)^*)$. □

Step II: Blockwise transductions. On our way towards rational transductions, we work with a subclass of transductions. If $T \subseteq A^* \times A^*$ is any subset, then we define the relation

$$T^* = \{(x_1 \cdots x_n, y_1 \cdots y_n) \mid n \in \mathbb{N}, (x_1, y_1), \ldots, (x_n, y_n) \in T\}.$$

We call a transduction *blockwise* if it is of the form T^* for some $T \subseteq A^{=k} \times A^{=\ell}$ and $k, \ell \in \mathbb{N}$.

Lemma 3.4. *Let $A \subseteq B$ be finite alphabets with $|B| \geq |A| + 1$. Every blockwise transduction $R \subseteq A^* \times A^*$ is Σ_1-definable over B.*

Proof. Let $\# \in B \setminus A$ be a symbol. Suppose that $R = T^*$ for some $T \subseteq A^{=k} \times A^{=\ell}$. Define the language $L = \{x\#y\# \mid (x, y) \in T\}^*$. Note that

$$w \in L \iff w \in (A^{=k}\#A^{=\ell}\#)^* \;\land\; \pi_A(w) \in \{xy \mid (x, y) \in T\}^*,$$

and hence L is Σ_1-definable over B by Lemma 3.3. The languages $X = (A^{=k}\#\#)^*$ and $Y = (\#A^{=\ell}\#)^*$ are also definable over B by Lemma 3.3. Then $(x, y) \in R$ if and only if

$$\exists w \in L, \hat{x} \in X, \hat{y} \in Y: \hat{x}, \hat{y} \preccurlyeq w \;\land\; |w|_\# = |\hat{x}|_\# = |\hat{y}|_\# \;\land\; x = \pi_A(\hat{x}) \;\land\; y = \pi_A(\hat{y}). \quad □$$

Step III: Rational transductions. We are ready to define arbitrary rational transductions.

Lemma 3.5. *Let $A \subseteq B$ be finite alphabets where $|A| + 1 \le |B|$ and $|B| \ge 3$. Every rational transduction $T \subseteq (A^*)^k$ is Σ_1-definable over B.*

Proof. Let $a, b \in B$. Let us first give an overview. Suppose the transducer for T has n transitions. Of course, we may assume that every run contains at least one transition. The idea is that a sequence of transitions is encoded by a word, where transition $j \in \{1, \ldots, n\}$ is represented by $a^j b^{n+1-j}$. We will define predicates run and input_i for $i \in \{1, \ldots, k\}$ with

$$(w_1, \ldots, w_k) \in T \iff \exists w \in \{a, b\}^* : \mathsf{run}(w) \wedge \bigwedge_{i=1}^{k} \mathsf{input}_i(w, w_i).$$

Here, $\mathsf{run}(w)$ states that w encodes a sequence of transitions that is a run of the transducer. Moreover, $\mathsf{input}_i(w, w_i)$ states that $w_i \in A^*$ is the input of this run in the i-th coordinate.

We begin with the predicate run. Let us call the words in $X = \{a^j b^{n+1-j} \mid j \in \{1, \ldots, n\}\}$ the *transition codes*. Let Δ be the set of all words $a^i b^{n+1-i} a^j b^{n+1-j}$ for which the target state of transition i and the source state of transition j are the same. Note that a word $w \in X^*$ represents a run if

(1) w begins with a transition that can be applied in an initial state,
(2) w ends with a transition that leads to a final state, and
(3) either $w \in \Delta^* \cap X\Delta^* X$ or $w \in X\Delta^* \cap \Delta^* X$, depending on whether the run has an even or an odd number of transitions.

Thus, we can define $\mathsf{run}(w)$ using prefix and suffix relations and membership to sets Δ^*. The prefix and suffix relation can be defined over $\{a, b\}$ using concatenation, see also [HSZ17, Theorem III.3, step 14]. Finally, we can express $w \in X^*$, $w \in \Delta^*$ and similar with Lemma 3.3.

It remains to define the input_i predicate. In the case that every transition reads a single letter on each input (i.e., no ε input), we can simply replace each transition code in w by its i-th input letter using a blockwise transduction. To handle ε inputs, we define input_i in two steps. Fix i and let $A = \{a_1, \ldots, a_m\}$. We first obtain an encoded version u_i of the i-th input from w: For every transition that reads a_j, we replace its transition code with $ab^j ab^{m-j} a$. Moreover, for each transition that reads ε, we replace the transition code by b^{m+3}. Using Lemma 3.4, this replacement is easily achieved using a blockwise transduction. Hence, each possible input in $A \cup \{\varepsilon\}$ is encoded using a block from $Y \cup \{b^{m+3}\}$, where $Y = \{ab^j ab^{m-j} a \mid j \in \{1, \ldots, m\}\}$.

Suppose we have produced the encoded input $u_i \in (Y \cup \{b^{m+3}\})^*$. In the next step, we want to define the word $v_i \in Y^*$, which is obtained from u_i by removing each block b^{m+3} from u_i. We do this as follows:

$$v_i \in Y^* \wedge v_i \preccurlyeq u_i \wedge |v_i|_a = |u_i|_a.$$

Note that here, we can express $v_i \in Y^*$ because of Lemma 3.3. In the final step, we turn v_i into the input $w_i \in A$ by replacing each block $ab^j ab^{m-j} a$ with a_j for $j \in \{1, \ldots, m\}$. This is just a blockwise transduction and can be defined by Lemma 3.4 because $|B| \ge |A| + 1$. $\square$

Remark 3.6. We do not use this here, but Lemma 3.5 also holds without the assumption $|B| \ge 3$. Indeed, if $|B| = 2$, then this would imply $|A_i| = 1$ for every i. Then we can write $A_i = \{a_i\}$ for (not necessarily distinct) letters $a_1, \ldots, a_k$. Since T is rational, the set of all $(x_1, \ldots, x_k) \in \mathbb{N}^k$ with $(a_1^{x_1}, \ldots, a_k^{x_k}) \in T$ is semilinear, and thus Σ_1-definable in $(\mathbb{N}, +, 0)$. It follows from the known predicates that T is Σ_1-definable using subwords over $\{a_1, \ldots, a_k\}$.

Step IV: Generator language. Our next ingredient is to express a particular non-regular language G (and its variant $G_\#$):

$$G = \{a^n b^n \mid n \geq 0\}^*, \qquad\qquad G_\# = \{a^n b^n \# \mid n \geq 0\}^*.$$

This will be useful because from G, one can produce all recursively enumerable sets by way of rational transductions and intersection.

Lemma 3.7. *The language $\{ab, \#\}^*$ is Σ_1-definable over $\{a, b, \#\}$.*

Proof. Note that

$$u \in \{ab, \#\}^* \iff \exists v \in \#^* ab \#^*, \ w \in v^* : u \preccurlyeq w \ \wedge \ \pi_{a,b}(u) = \pi_{a,b}(w).$$

Here, the language $\#^* ab \#^*$ can be defined using concatenation. Moreover, since every word in $\#^* ab \#^*$ is primitive, we express $w \in v^*$ by saying $vw = wv$.   □

Lemma 3.8. *Let $\{a, b\} \subseteq A$ and $|A| \geq 3$. The language G is Σ_1-definable over A.*

Proof. Suppose $\# \in A \setminus \{a, b\}$. Since $G = \pi_{a,b}(G_\#)$, it suffices to define $G_\#$. We can define the language $a^* b^* \#$ as a concatenation of a^*, b^*, and $\#$. The next step is to define the language $K = (a^* b^* \#)^*$. To this end, notice that

$$w \in K \iff \exists u \in a^* b^* \#, \ v \in u^* : w \preccurlyeq v \ \wedge \ |w|_\# = |v|_\#.$$

Here, since the words in $a^* b^* \#$ are primitive, we can express $v \in u^*$ by saying $vu = uv$. Thus, we can define K. Using K and Lemma 3.7, we can define $G_\#$, since

$$w \in G_\# \iff w \in K \ \wedge \ \exists v \in \{ab, \#\}^* : \pi_{a,\#}(w) = \pi_{a,\#}(v) \ \wedge \ \pi_{b,\#}(w) = \pi_{b,\#}(v).\qquad □$$

Step V: Recursively enumerable relations over two letters. We are now ready to define all recursively enumerable relations over two letters in $(A^*, \preccurlyeq, (w)_{w \in A^*})$, provided that $|A| \geq 3$. For two rational transductions $T \subseteq A^* \times B^*$ and $S \subseteq B^* \times C^*$, and a language $L \subseteq A^*$, we denote *application* of T to L as $TL = \{v \in B^* \mid \exists u \in L : (u, v) \in T\} \subseteq B^*$, and we denote *composition* of S and T as $S \circ T = \{(u, w) \mid \exists v \in B^* : (u, v) \in T \wedge (v, w) \in S\} \subseteq A^* \times C^*$. The latter is again a rational transduction (see e.g. [Ber79]).

Lemma 3.9 (Hartmanis & Hopcroft 1970)**.** *Every recursively enumerable language L can be written as $L = \alpha(T_1 G_\# \cap T_2 G_\#)$ with a morphism α and rational transductions T_1, T_2.*

Proof. This follows directly from [HH70, Theorem 1] and the proof of [HH70, Theorem 2].   □

Let us briefly sketch the proof of Lemma 3.9. It essentially states that every recursively enumerable language can be accepted by a machine with access to two counters that work in a restricted way. The two counters have instructions to *increment*, *decrement*, and *zero test* (which correspond to the letters a, b, and $\#$ in $G_\#$). The restriction, which we call "locally one-reversal" (L1R) is that in between two zero tests of some counter, the instructions of that counter must be *one-reversal*: There is a phase of increments and then a phase of decrements (in other words: after a decrement, no increments are allowed until the next zero test).

   To show this, Hartmanis and Hopcroft use the classical fact that every recursively enumerable language can be accepted by a four counter machine (without the L1R property). Then, the four counter values p, q, r, s can be encoded as $2^p 3^q 5^r 7^s$ in a single integer register that can (i) multiply with, (ii) divide by, (iii) test non-divisibility by the constants $2, 3, 5, 7$.

Such a register, in turn, is easily simulated using two L1R-counters: For example, to multiply by $f \in \{2, 3, 5, 7\}$, one uses a loop that decrements the first counter and increments the second by f, until the first counter is zero. The other instructions are similar.

Lemma 3.10. *For every recursively enumerable relation $R \subseteq (\{a, b\}^*)^k$, there is a rational transduction $T \subseteq (\{a, b\}^*)^{k+2}$ such that*

$$(w_1, \ldots, w_k) \in R \iff \exists u, v \in G \colon (w_1, \ldots, w_k, u, v) \in T. \tag{3.4}$$

Proof. We shall build T out of several other transductions. These will be over larger alphabets, but since we merely compose them to obtain T, this is not an issue.

A standard fact from computability theory states that a relation is recursively enumerable if and only if it is the homomorphic image of some recursively enumerable language. In particular, there is a recursively enumerable language $L \subseteq B^*$ and morphisms $\beta_1, \ldots, \beta_k$ such that $R = \{(\beta_1(w), \ldots, \beta_k(w)) \mid w \in L\}$. By Lemma 3.9, we may write $L = \alpha(T_1 G_\# \cap T_2 G_\#)$ for a morphism $\alpha \colon C^* \to B^*$ and rational transductions $T_1, T_2 \subseteq \{a, b, \#\}^* \times C^*$.

Notice that if $\gamma \colon \{a, b, \#\}^* \to \{a, b\}^*$ is the morphism with $\gamma(a) = a$, $\gamma(b) = b$, and $\gamma(\#) = abab$, then $G_\# = (a^* b^* \#)^* \cap \gamma^{-1}(G)$. Taking the pre-image under a morphism (here γ) and then intersecting with the regular language (here $(a^* b^* \#)^*$) can be performed by a single rational transduction [Ber79, Theorem 3.2]. This means, there is a rational transduction $S \subseteq \{a, b\}^* \times \{a, b, \#\}^*$ with $G_\# = SG$. Therefore, we can replace $G_\#$ in the above expression for L and arrive at $L = \alpha((T_1(SG) \cap (T_2(SG)) = \alpha((T_1 \circ S)G \cap (T_2 \circ S)G)$. In sum, we observe that $(w_1, \ldots, w_k) \in R$ if and only if there exists a $w \in C^*$ with $w \in (T_1 \circ S)G$ and $w \in (T_2 \circ S)G$ such that $w_i = \beta_i(\alpha(w))$ for $i \in \{1, \ldots, k\}$. Consider the relation

$$T = \{(\beta_1(\alpha(w)), \ldots, \beta_k(\alpha(w)), u, v) \mid w \in C^*, \ (u, w) \in T_1 \circ S, \ (v, w) \in T_2 \circ S\}.$$

Note that T is rational: A transducer can guess w, letter by letter, and on track $i \in \{1, \ldots, k\}$, it outputs the image under $\beta_i(\alpha(\cdot))$ of each letter. To compute the output on tracks $k + 1$ and $k + 2$, it simulates transducers for $T_1 \circ S$ and $T_2 \circ S$. Moreover, we have $T \subseteq (\{a, b\}^*)^{k+2}$ and our observation implies that (3.4) holds. $\square$

Lemma 3.11. *Let A be an alphabet with $\{a, b\} \subseteq A$ and $|A| \geq 3$. Then every recursively enumerable relation $R \subseteq (\{a, b\}^*)^k$ is Σ_1-definable over A.*

Proof. Take the rational transduction T as in Lemma 3.10. Since $T \subseteq (\{a, b\}^*)^{k+2}$ and $|A| \geq |\{a, b\}| + 1$, Lemma 3.5 and Lemma 3.8 yield the result. $\square$

Step VI: Arbitrary recursively enumerable relations. We have seen that if $|A| \geq 3$, then we can define over A every recursively enumerable relation over two letters. In the proof, we use a third letter as an auxiliary letter. Our last step is to define all recursively enumerable relations that can use all letters of A freely. This clearly implies Theorem 2.1. To this end, we observe that every word is determined by its binary projections.

Lemma 3.12. *Let A be an alphabet with $|A| \geq 2$ and let $u, v \in A^*$ such that for every binary alphabet $B \subseteq A$, we have $\pi_B(u) = \pi_B(v)$. Then $u = v$.*

Proof. Towards a contradiction, suppose $u \neq v$. We clearly have $|u| = |v|$. Thus, if $w \in A^*$ is the longest common prefix of u and v, then $u = wau'$ and $v = wbv'$ for some letters $a \neq b$ and words $u', v' \in A^*$. But then the words $\pi_{a,b}(u)$ and $\pi_{a,b}(v)$ differ: After the common prefix $\pi_{a,b}(w)$, the word $\pi_{a,b}(u)$ continues with a and the word $\pi_{a,b}(v)$ continues with b. $\square$

We now fix $a, b \in A$ with $a \neq b$. For any binary alphabet $B \subseteq A$ let $\rho_B \colon A^* \to \{a, b\}^*$ be any morphism with $\rho_B(B) = \{a, b\}$ and $\rho_B(c) = \varepsilon$ for all $c \in A \setminus B$, i.e., ρ_B first projects a word over A to B and then renames the letters from B to $\{a, b\}$. Recall that $\binom{|A|}{2}$ is the number of binary alphabets $B \subseteq A$. We define the encoding function $e \colon A^* \to (\{a, b\}^*)^{\binom{|A|}{2}}$ which maps a word $u \in A^*$ to the tuple consisting of all words $\rho_B(u)$ for all binary alphabets $B \subseteq A$ (in some arbitrary order). Note that e is injective by Lemma 3.12.

Lemma 3.13. *If $|A| \geq 3$, then $e \colon A^* \to (\{a, b\}^*)^{\binom{|A|}{2}}$ is Σ_1-definable over A.*

Proof. For binary alphabets $B, C \subseteq A$, a map $\sigma \colon B^* \to C^*$ is called a *binary renaming* if (i) σ is a word morphism and (ii) σ restricted to B is a bijection of B and C. If, in addition, there is a letter $\# \in B \cap C$ such that $\sigma(\#) = \#$, then we say that σ *fixes a letter*.

Observe that if we can Σ_1-define all binary renamings, then the encoding function e can be Σ_1-defined using projections and binary renamings. Thus, it remains to define all binary renamings. For this, note that every binary renaming can be written as a composition of (at most three) binary renamings that each fix some letter. Hence, it suffices to define any binary renaming that fixes a letter. Suppose $\sigma \colon \{c, \#\}^* \to \{d, \#\}^*$ with $\sigma(c) = d$ and $\sigma(\#) = \#$. Without loss of generality, we assume $c \neq d$. Then σ is Σ_1-definable since

$$\sigma(u) = v \iff \exists w \in \{cd, \#\}^* \colon u = \pi_{c, \#}(w) \ \wedge \ v = \pi_{d, \#}(w).$$

and $\{cd, \#\}^*$ is definable by Lemma 3.7. $\qquad \square$

Theorem 3.14. *Let A be an alphabet with $|A| \geq 3$. Then every recursively enumerable relation $R \subseteq (A^*)^k$ is Σ_1-definable in $(A^*, \preccurlyeq, (w)_{w \in A^*})$.*

Proof. The encoding function e is clearly computable and injective by Lemma 3.12. Therefore a relation $R \subseteq (A^*)^k$ is recursively enumerable if and only if the image

$$e(R) = \{(e(w_1), \dots, e(w_k)) \mid (w_1, \dots, w_k) \in R\} \subseteq (\{a, b\}^*)^{k \cdot \binom{|A|}{2}}$$

is recursively enumerable. This means that $e(R)$ is Σ_1-definable over A by Lemma 3.11. Thus, we can define R as well, since we have

$$(w_1, \dots, w_k) \in R \iff (e(w_1), \dots, e(w_k)) \in e(R),$$

and the function e is Σ_1-definable over A by Lemma 3.13. $\qquad \square$

4. Restricting the signature to a single constant

In [HSZ17, Remark 3.4] the authors observe that their undecidability result for the existential fragment of subword logic with constants still holds, even if only a finite set of constant symbols is allowed. More precisely, they show that for any alphabet A with $|A| \geq 2$ there are finitely many words $w_1, \dots, w_n \in A^*$ such that the truth problem for the Σ_1-fragment over the structure $(A^*, \preccurlyeq, w_1, \dots, w_n)$ is undecidable. Furthermore, they remark that one can strengthen this result even more to only requiring a single constant $W \in A^*$. The proof of the latter can be found in the extended version [HSZ21, Theorem 3.5]. Using similar techniques, we can likewise strengthen our Σ_1-definability result for alphabets of size at least 3:

Theorem 4.1. *Let A be an alphabet with $|A| \geq 3$. Then there is a fixed word $W \in A^*$ such that every recursively enumerable relation $R \subseteq (A^*)^k$ is Σ_1-definable in $(A^*, \preccurlyeq, W)$.*

Like in [HSZ21], we make use of the fact that any word of length at least 3 is uniquely defined by its set of strict subwords:

Lemma 4.2 ([KS15]). *Let u, v be words with $|u| \geq 3$ and $|v| \geq 3$. Then $u = v$ if and only if $u{\downarrow} \setminus \{u\} = v{\downarrow} \setminus \{v\}$.*

Note that Lemma 4.2 does not hold for words of length 2 since $ab{\downarrow} \setminus \{ab\} = \{\varepsilon, a, b\} = ba{\downarrow} \setminus \{ba\}$.

Proof of Theorem 4.1. We begin with the case $A = \{a, b, c\}$, i.e. $|A| = 3$.
Recall that concatenation is Σ_1-definable in the structure $(A^*, \preccurlyeq, (w)_{w \in A^*})$. Let $w_1, \ldots, w_n$ be the constant symbols that appear in the formula defining the concatenation relation. We choose $W = a^{m+1} b^{m+2} c^{m+3}$, where $m = \max_{1 \leq i \leq n} |w_i|$ is the maximal length among these constants. Since every word is a concatenation of letters, it now suffices to show that the letters a, b, c and words $w_1, \ldots, w_n$ are all Σ_1-definable over $(A^*, \preccurlyeq, W)$.

In the following we use $u \prec v$ as a shorthand for $u \preccurlyeq v \wedge u \neq v$. The formula

$$v_{3m+5} \prec W \wedge \bigwedge_{i=0}^{3m+4} v_i \prec v_{i+1}$$

defines a sequence of subwords $v_0, \ldots, v_{3m+5}$ of W with $|v_i| = i$. In particular, we have $v_1 \in A$. If we repeat the same formula for two more sets of variables $u_0, \ldots, u_{3m+5}$, $t_0, \ldots, t_{3m+5}$, and additionally require

$$v_1 \neq u_1 \wedge v_1 \neq t_1 \wedge u_1 \neq t_1,$$

then we have defined a, b, and c, but only up to renaming of letters. To ensure $v_1 = c$ we use the following formula:

$$u_1 \not\preccurlyeq v'_{m+3} \wedge t_1 \not\preccurlyeq v'_{m+3} \wedge v'_{m+3} \preccurlyeq W \wedge \bigwedge_{i=0}^{m+2} v'_i \prec v'_{i+1}.$$

It defines a sequence of $v'_0, \ldots v'_{m+3}$ of subwords of W, among which the letters u_1 and t_1 do not occur. Therefore this sequence is comprised of words in v_1^*, and by choice of W a sequence of this length cannot exist for $v_1 = a$ or $v_1 = b$. Observe that also $|v'_i| = i$, which means that we have now additionally defined the words $c^0 = \varepsilon$ to c^{m+3}. Using two similar formulas with $m + 3$ and $m + 2$ variables, respectively, we can now define the words b^0 to b^{m+2} and a^0 to a^{m+1} as well.

Observe that for any word w and any letter a', $|w|_{a'} = \ell$ is equivalent to $a^\ell \preccurlyeq w \wedge a^{\ell+1} \not\preccurlyeq w$. Using this fact we can fix the number of occurrences of each letter a', and therefore also fix the total length of a word. We continue by defining the remaining words of length ≤ 2, which are ab, ba, ac, ca, bc, cb. The formula

$$a \preccurlyeq s_{01} \wedge aa \not\preccurlyeq s_{01} \wedge b \preccurlyeq s_{01} \wedge bb \not\preccurlyeq s_{01} \wedge c \not\preccurlyeq s_{01} \wedge s_{01} \preccurlyeq W$$

defines the word $s_{01} = ab$. By copying this formula for a new free variable s_{10} and replacing the last conjunct by $s_{10} \not\preccurlyeq W$, we can likewise define $s_{10} = ba$. Similarly, we can also distinguish ac from ca, as well as bc from cb, since in both cases one of them is a subword of W while the other is not.

Finally we inductively define all constants up to length m. Let w be a word of length $3 \leq |w| \leq m$. Then by induction hypothesis we have defined all constants up to length $|w| - 1$. Furthermore for every $a' \in A$ we have defined the constants $a'^{|w|_{a'}}$ and $a'^{|w|_{a'}+1}$,

since $|w|_{a'} \leq |w| \leq m$. By Lemma 4.2 the word w is uniquely defined by its set of strict subwords. Therefore the following formula defines $s = w$:

$$a^{|w|_a} \preccurlyeq s \wedge a^{|w|_a+1} \not\preccurlyeq s \wedge b^{|w|_b} \preccurlyeq s \wedge b^{|w|_b+1} \not\preccurlyeq s \wedge c^{|w|_c} \preccurlyeq s \wedge c^{|w|_c+1} \not\preccurlyeq s$$

$$\wedge \bigwedge_{|w'|\leq|w|,w'\preccurlyeq w} w' \preccurlyeq s \quad \wedge \bigwedge_{|w'|\leq|w|,w'\not\preccurlyeq w} w' \not\preccurlyeq s.$$

Since a, b, c and $w_1, \ldots, w_n$ have at most length m, we have successfully defined all of the required constants. To conclude the case $|A| = 3$, we add existential quantifiers for all the variables representing constants that we do not use outside of these auxiliary formulas. Since we only ever used existential quantification, we have shown definability in the Σ_1-fragment of $(A^*, \preccurlyeq, W)$.

The case $A = \{a_1, \ldots, a_k\}$ for $k \geq 4$ is very similar. We define the number m in the same way as before and choose $W = a_1^{m+1} \cdots a_k^{m+k}$. Then we can again define all constants a_i^0 to a_i^{m+1} for every letter a_i. We also proceed to define all remaining words of length 2, in the same way as in the previous case. Finally, Lemma 4.2 and the fact $|w|_{a_i} = \ell \iff a_i^\ell \preccurlyeq w \wedge a_i^{\ell+1} \not\preccurlyeq w$ allow us to inductively define all constants w up to length m, like before. $\qquad\square$

Remark 4.3. In the proof of Theorem 4.1 we show that there is a number $m \in \mathbb{N}$ such that for the alphabet $A = \{a_1, \ldots, a_k\}$ with $k \geq 4$ the word $a_1^{m+1} \cdots a_k^{m+k}$ is a valid choice for the constant symbol W. Moreover, the proof still works for any number $m' > m$ and $W = a_1^{m'+1} \cdots a_k^{m'+k}$. Therefore the theorem does not just hold for one fixed word W, but an infinite family of words $W_{m'}$.

5. Further consequences

In this section, we prove Corollary 2.2, Corollary 2.4 and Observation 2.5. When working with higher levels (Σ_i^0 for $i \geq 2$) of the arithmetic hierarchy, it will be convenient to use a slightly different definition than the one using oracle Turing machines: [Koz10, Theorem 35.1] implies that for $i \geq 1$, a relation $R \subseteq (A^*)^k$ belongs to Σ_{i+1}^0 if and only if it can be written as $R = \pi((A^*)^{k+\ell} \setminus S)$, where $S \subseteq (A^*)^{k+\ell}$ is a relation in Σ_i^0 and $\pi\colon (A^*)^{k+\ell} \to (A^*)^k$ is the projection to the first k coordinates.

Proof of Corollary 2.2. It is immediate that every predicate definable in the Σ_i-fragment of $(A^*, \preccurlyeq, (w)_{w\in A^*})$ belongs to Σ_i^0, because the subword relation is recursively enumerable. We show the converse using induction on i, such that Theorem 2.1 is the base case.

Now suppose that every relation in Σ_i^0 is definable in the Σ_i-fragment of $(A^*, \preccurlyeq, (w)_{w\in A^*})$ and consider a relation $R \subseteq (A^*)^k$ in Σ_{i+1}^0. Then we can write $R = \pi((A^*)^{k+\ell} \setminus S)$ for some $\ell \geq 0$, where $\pi\colon (A^*)^{k+\ell} \to (A^*)^k$ is the projection to the first k coordinates, and $S \subseteq (A^*)^{k+\ell}$ is a relation in Σ_i^0. By induction, S is definable by a Σ_i-formula φ over $(A^*, \preccurlyeq, (w)_{w\in A^*})$. By negating φ and moving all negations inwards, we obtain a Π_i-formula ψ that defines $(A^*)^{k+\ell} \setminus S$. Finally, adding existential quantifiers for the variables corresponding to the last ℓ coordinates yields a Σ_{i+1}-formula for $R = \pi((A^*)^{k+\ell} \setminus S)$. $\qquad\square$

Note that if instead of Theorem 2.1 we use Theorem 4.1 as the base case in the induction above, it follows that there exists a word $W \in A^*$ such that Corollary 2.2 also holds for the

structure $(A^*, \preccurlyeq, W)$ instead of $(A^*, \preccurlyeq, (w)_{w \in A^*})$. This together with Theorem 4.1 yields Remark 2.3.

Finally, we look at the expressive power of the pure logic $(A^*, \preccurlyeq)$. We start by proving Corollary 2.4, which characterizes relations definable in the Σ_i-fragment of $(A^*, \preccurlyeq)$ for $i \geq 2$.

Proof of Corollary 2.4. Clearly, every relation definable with a Σ_i-formula over $(A^*, \preccurlyeq)$ must be automorphism-invariant and must define a relation in Σ_i^0.

Conversely, consider an automorphism-invariant relation $R \subseteq (A^*)^k$ in Σ_i^0. Then R is definable using a Σ_i-formula φ with free variables $x_1, \ldots, x_k$ over $(A^*, \preccurlyeq, (w)_{w \in A^*})$ by Corollary 2.2. Let $w_1, \ldots, w_\ell$ be the constants occurring in φ. From φ, we construct the Σ_i-formula φ' over $(A^*, \preccurlyeq)$, by replacing each occurrence of w_j by a fresh variable y_j.

It was shown in [KS15, Sections 4.1 and 4.2] that from the tuple $(w_1, \ldots, w_\ell) \in (A^*)^\ell$, one can construct a Σ_2-formula ψ with free variables $y_1, \ldots, y_\ell$ over $(A^*, \preccurlyeq)$ such that $\psi(u_1, \ldots, u_\ell)$ is true if and only if there exists an automorphism of $(A^*, \preccurlyeq)$ mapping u_j to w_j for each j. We claim that the formula $\chi = \exists y_1, \ldots, y_\ell \colon \psi \wedge \varphi'$ defines the set R. Since ψ belongs to Σ_2 and thus χ belongs to Σ_i, this implies Corollary 2.4.

Clearly, every $(v_1, \ldots, v_k) \in R$ satisfies χ. Moreover, if $\chi(v_1, \ldots, v_k)$, then there are $u_1, \ldots, u_\ell \in A^*$ with $\varphi'(v_1, \ldots, v_k, u_1, \ldots, u_\ell)$ and an automorphism α mapping u_j to w_j for each j. Since α is an automorphism, the formula φ' is also satisfied on the tuple $(\alpha(v_1), \ldots, \alpha(v_k), \alpha(u_1), \ldots, \alpha(u_\ell)) = (\alpha(v_1), \ldots, \alpha(v_k), w_1, \ldots, w_\ell)$ and thus we have $(\alpha(v_1), \ldots, \alpha(v_k)) \in R$. Since R is automorphism-invariant, this implies $(v_1, \ldots, v_k) \in R$. □

The expressiveness of the existential fragment of $(A^*, \preccurlyeq)$ is not well understood. Partial results are summarized in Observation 2.5.

Proof of Observation 2.5. Take a recursively enumerable, but undecidable subset $S \subseteq \mathbb{N}$. Fix a letter $a \in A$ and define the unary language $L = \{a^n \mid n \in S\}$. By [HSZ21, Theorem 3.5] there exists a word $W \in A^*$ and a Σ_1-formula $\varphi(x)$ over $(A^*, \preccurlyeq, W)$ which defines L. Consider the formula φ' in the Σ_1-fragment of $(A^*, \preccurlyeq)$ obtained by replacing each occurrence of W by a fresh variable y. Then (v, W) satisfies φ' if and only if $v \in L$. Thus, φ' defines an undecidable relation.

For the second statement, we claim that every language $L \subseteq A^*$ that is Σ_1-definable in $(A^*, \preccurlyeq)$ satisfies $A^*LA^* \subseteq L$. Hence, many automorphism-invariant regular languages such as $\bigcup_{a \in A} a^*$ are not definable. Note that for $a \in A$ and $u, v \in A^*$, we have $u \preccurlyeq v$ if and only if $au \preccurlyeq av$. Thus, every Σ_0-definable relation $R \subseteq (A^*)^k$ satisfies $(w_1, \ldots, w_k) \in R$ if and only if $(aw_1, \ldots, aw_k) \in R$. Symmetrically, $(w_1, \ldots, w_k) \in R$ is equivalent to $(w_1a, \ldots, w_ka) \in R$. As a projection of a Σ_0-definable relation, L thus satisfies $A^*LA^* \subseteq L$. □

Observation 2.5 raises the question whether there are undecidable Σ_1-definable *languages* in $(A^*, \preccurlyeq)$, which we leave as an open problem. In fact, all examples of Σ_1-definable languages that we have constructed are regular. While each individual Σ_1-definable language could be decidable, the following observation implies that the membership problem for Σ_1-definable languages is *not* decidable, if the formula is part of the input.

Observation 5.1. Let $|A| \geq 2$. There exists a word $W \in A^*$ such that the following problem is undecidable: Given a Σ_1-formula $\varphi(x)$ over $(A^*, \preccurlyeq)$, does W satisfy $\varphi(x)$?

Proof. By [HSZ21, Theorem 3.5] there exists a word $W \in A^*$ so that the *truth problem* for Σ_1-formulas over $(A^*, \preccurlyeq, W)$ is undecidable, which asks whether a given a sentence φ (formula without free variables) is true. In other words, testing whether W satisfies the

formula obtained from φ by replacing each occurrence of W by a fresh free variable is undecidable. $\square$

6. Conclusion

We have shown how to define all recursively enumerable relations in the existential fragment of the subword order with constants for each alphabet A with $|A| \geq 3$. If $|A| = 1$, then the relations definable in $(A^*, \preccurlyeq, (w)_{w \in A^*})$ correspond to relations over $\mathbb{N}$ definable in $(\mathbb{N}, \leq)$ with constants. Hence, this case is very well understood: This structure admits quantifier elimination [Pél92, Theorem 2.2(b)], which implies that the Σ_1-fragment is expressively complete and also that a subset of A^* is only definable if it is finite or co-finite. In particular, Theorem 2.1 does not hold for $|A| = 1$.

We leave open whether Theorem 2.1 still holds over a binary alphabet. If this is the case, then we expect that substantially new techniques are required. In order to express non-trivial relations over two letters, our proof often uses a third letter as a separator and marker for "synchronization points" in subword embeddings.

References

[ABQ11]   Mohamed Faouzi Atig, Ahmed Bouajjani, and Shaz Qadeer. Context-bounded analysis for concurrent programs with dynamic creation of threads. *Log. Methods Comput. Sci.*, 7(4), 2011. `doi:10.2168/LMCS-7(4:4)2011`.

[ACH⁺16]   Mohamed Faouzi Atig, Dmitry Chistikov, Piotr Hofman, K. Narayan Kumar, Prakash Saivasan, and Georg Zetzsche. The complexity of regular abstractions of one-counter languages. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 207–216. ACM, 2016. `doi:10.1145/2933575.2934561`.

[AJ96]   Parosh Aziz Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. *information and computation*, 127(2):91–101, 1996.

[AMMS17]   Mohamed Faouzi Atig, Roland Meyer, Sebastian Muskalla, and Prakash Saivasan. On the upward/downward closures of Petri nets. In Kim G. Larsen, Hans L. Bodlaender, and Jean-François Raskin, editors, *42nd International Symposium on Mathematical Foundations of Computer Science, MFCS 2017, August 21-25, 2017 - Aalborg, Denmark*, volume 83 of *LIPIcs*, pages 49:1–49:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. `doi:10.4230/LIPIcs.MFCS.2017.49`.

[AZ23]   Ashwani Anand and Georg Zetzsche. Priority downward closures. In Guillermo A. Pérez and Jean-François Raskin, editors, *34th International Conference on Concurrency Theory, CONCUR 2023, September 18-23, 2023, Antwerp, Belgium*, volume 279 of *LIPIcs*, pages 39:1–39:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. `doi:10.4230/LIPICS.CONCUR.2023.39`.

[BCCP20]   David Barozzini, Lorenzo Clemente, Thomas Colcombet, and Pawel Parys. Cost automata, safe schemes, and downward closures. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 109:1–109:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. `doi:10.4230/LIPIcs.ICALP.2020.109`.

[Ber79]   Jean Berstel. *Transductions and Context-Free Languages*. Teubner, 1979.

[BFH⁺20]   Laura Barker, Pamela Fleischmann, Katharina Harwardt, Florin Manea, and Dirk Nowotka. Scattered factor-universality of words. In *Developments in Language Theory - 24th International Conference, DLT 2020, Tampa, FL, USA, May 11-15, 2020, Proceedings*, volume 12086 of *Lecture Notes in Computer Science*, pages 14–28. Springer, 2020. `doi:10.1007/978-3-030-48516-0\_2`.

[BGM⁺23a] Pascal Baumann, Moses Ganardi, Rupak Majumdar, Ramanathan S. Thinniyam, and Georg Zetzsche. Checking refinement of asynchronous programs against context-free specifications. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 110:1–110:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. `doi:10.4230/LIPICS.ICALP.2023.110`.

[BGM⁺23b] Pascal Baumann, Moses Ganardi, Rupak Majumdar, Ramanathan S. Thinniyam, and Georg Zetzsche. Context-bounded analysis of concurrent programs (invited talk). In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 3:1–3:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. `doi:10.4230/LIPICS.ICALP.2023.3`.

[BGTZ22] Pascal Baumann, Moses Ganardi, Ramanathan S. Thinniyam, and Georg Zetzsche. Existential definability over the subword ordering. In Petra Berenbrink and Benjamin Monmege, editors, *39th International Symposium on Theoretical Aspects of Computer Science, STACS 2022, March 15-18, 2022, Marseille, France (Virtual Conference)*, volume 219 of *LIPIcs*, pages 7:1–7:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. `doi:10.4230/LIPICS.STACS.2022.7`.

[BK15] Karl Bringmann and Marvin Künnemann. Quadratic conditional lower bounds for string problems and dynamic time warping. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 79–97. IEEE Computer Society, 2015. `doi:10.1109/FOCS.2015.15`.

[BK18] Karl Bringmann and Marvin Künnemann. Multivariate fine-grained complexity of longest common subsequence. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1216–1235. SIAM, 2018. `doi:10.1137/1.9781611975031.79`.

[BMTZ20] Pascal Baumann, Rupak Majumdar, Ramanathan S. Thinniyam, and Georg Zetzsche. The complexity of bounded context switching with dynamic thread creation. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 111:1–111:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. `doi:10.4230/LIPICS.ICALP.2020.111`.

[BMTZ22] Pascal Baumann, Rupak Majumdar, Ramanathan S. Thinniyam, and Georg Zetzsche. Context-bounded verification of thread pools. *Proc. ACM Program. Lang.*, 6(POPL):1–28, 2022. `doi:10.1145/3498678`.

[BY91] Ricardo A. Baeza-Yates. Searching subsequences. *Theoretical Computer Science*, 78(2):363–376, 1991.

[CPSW16] Lorenzo Clemente, Pawel Parys, Sylvain Salvati, and Igor Walukiewicz. The diagonal problem for higher-order recursion schemes is decidable. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 96–105. ACM, 2016. `doi:10.1145/2933575.2934527`.

[DFK⁺21] Joel D. Day, Pamela Fleischmann, Maria Kosche, Tore Koß, Florin Manea, and Stefan Siemer. The edit distance to k-subsequence universality. In *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference)*, volume 187 of *LIPIcs*, pages 25:1–25:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. `doi:10.4230/LIPIcs.STACS.2021.25`.

[ERW08] Cees Elzinga, Sven Rahmann, and Hui Wang. Algorithms for subsequence combinatorics. *Theoretical Computer Science*, 409(3):394–404, 2008.

[FK18] Lukas Fleischer and Manfred Kufleitner. Testing Simon's congruence. In *43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018, August 27-31, 2018, Liverpool, UK*, volume 117 of *LIPIcs*, pages 62:1–62:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. `doi:10.4230/LIPIcs.MFCS.2018.62`.

[FKK⁺23] Pamela Fleischmann, Sungmin Kim, Tore Koß, Florin Manea, Dirk Nowotka, Stefan Siemer, and Max Wiedenhöft. Matching patterns with variables under simon's congruence. In Olivier Bournez, Enrico Formenti, and Igor Potapov, editors, *Reachability Problems - 17th International Conference, RP 2023, Nice, France, October 11-13, 2023, Proceedings*, volume 14235 of *Lecture*

Notes in Computer Science, pages 155–170. Springer, 2023. `doi:10.1007/978-3-031-45286-4\_12`.

[GKK⁺21]   Pawel Gawrychowski, Maria Kosche, Tore Koß, Florin Manea, and Stefan Siemer. Efficiently testing Simon's congruence. In *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference)*, volume 187 of *LIPIcs*, pages 34:1–34:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. `doi:10.4230/LIPIcs.STACS.2021.34`.

[GLHK⁺20] Jean Goubault-Larrecq, Simon Halfon, Prateek Karandikar, K. Narayan Kumar, and Philippe Schnoebelen. *The Ideal Approach to Computing Closed Subsets in Well-Quasi-orderings*, pages 55–105. Springer International Publishing, Cham, 2020. `doi:10.1007/978-3-030-30229-0_3`.

[HH70]   Juris Hartmanis and John E Hopcroft. What makes some language theory problems undecidable. *Journal of Computer and System Sciences*, 4(4):368–376, 1970.

[HKO16]   Matthew Hague, Jonathan Kochems, and C.-H. Luke Ong. Unboundedness and downward closures of higher-order pushdown automata. In Rastislav Bodík and Rupak Majumdar, editors, *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016*, pages 151–163. ACM, 2016. `doi:10.1145/2837614.2837627`.

[HMW10]   Peter Habermehl, Roland Meyer, and Harro Wimmel. The downward-closure of Petri net languages. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, volume 6199 of *Lecture Notes in Computer Science*, pages 466–477. Springer, 2010. `doi:10.1007/978-3-642-14162-1\_39`.

[HSZ17]   Simon Halfon, Philippe Schnoebelen, and Georg Zetzsche. Decidability, complexity, and expressiveness of first-order logic over the subword ordering. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017*, pages 1–12. IEEE Computer Society, 2017. `doi:10.1109/LICS.2017.8005141`.

[HSZ21]   Simon Halfon, Philippe Schnoebelen, and Georg Zetzsche. Decidability, complexity, and expressiveness of first-order logic over the subword ordering. *CoRR*, abs/1701.07470, 2021. URL: `http://arxiv.org/abs/1701.07470`, `arXiv:1701.07470`.

[KKS15]   Prateek Karandikar, Manfred Kufleitner, and Philippe Schnoebelen. On the index of Simon's congruence for piecewise testability. *Inf. Process. Lett.*, 115(4):515–519, 2015. `doi:10.1016/j.ipl.2014.11.008`.

[Koz10]   Dexter C. Kozen. *Theory of computation*. Springer Verlag London Limited, 2010.

[KS15]   Prateek Karandikar and Philippe Schnoebelen. Decidability in the logic of subsequences and supersequences. In *35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2015, December 16-18, 2015, Bangalore, India*, volume 45 of *LIPIcs*, pages 84–97. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015. `doi:10.4230/LIPIcs.FSTTCS.2015.84`.

[KS19]   Prateek Karandikar and Philippe Schnoebelen. The height of piecewise-testable languages and the complexity of the logic of subwords. *Log. Methods Comput. Sci.*, 15(2), 2019. `doi:10.23638/LMCS-15(2:6)2019`.

[KS20]   Dietrich Kuske and Christian Schwarz. Complexity of Counting First-Order Logic for the Subword Order. In Javier Esparza and Daniel Král, editors, *45th International Symposium on Mathematical Foundations of Computer Science (MFCS 2020)*, volume 170 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 61:1–61:12, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.MFCS.2020.61`.

[KSY10]   Oleg V. Kudinov, Victor L. Selivanov, and Lyudmila V. Yartseva. Definability in the subword order. In *Conference on Computability in Europe*, pages 246–255. Springer, 2010.

[Kus06]   Dietrich Kuske. Theories of orders on the set of words. *RAIRO-Theoretical Informatics and Applications*, 40(01):53–74, 2006.

[KZ19]   Dietrich Kuske and Georg Zetzsche. Languages ordered by the subword order. In *Foundations of Software Science and Computation Structures - 22nd International Conference, FOSSACS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS*

2019, Prague, Czech Republic, April 6-11, 2019, Proceedings, volume 11425 of *Lecture Notes in Computer Science*, pages 348–364. Springer, 2019. `doi:10.1007/978-3-030-17127-8\_20`.

[Mai78] David Maier. The complexity of some problems on subsequences and supersequences. *Journal of the ACM (JACM)*, 25(2):322–336, 1978.

[Mat93] Yuri Matiyasevich. *Hilbert's tenth problem*. MIT press, 1993.

[MTZ22] Rupak Majumdar, Ramanathan S. Thinniyam, and Georg Zetzsche. General decidability results for asynchronous shared-memory programs: Higher-order and beyond. *Log. Methods Comput. Sci.*, 18(4), 2022. `doi:10.46298/LMCS-18(4:2)2022`.

[Pél92] Pierre Péladeau. Logically defined subsets of N^k. *Theoretical computer science*, 93(2):169–183, 1992.

[SS97] Jacques Sakarovitch and Imre Simon. Subwords. In M. Lothaire, editor, *Combinatorics on Words*, Cambridge Mathematical Library, chapter 6, pages 105–142. Cambridge University Press, 2nd edition, 1997. `doi:10.1017/CBO9780511566097.009`.

[SV23] Philippe Schnoebelen and Julien Veron. On arch factorization and subword universality for words and compressed words. In Anna E. Frid and Robert Mercas, editors, *Combinatorics on Words - 14th International Conference, WORDS 2023, Umeå, Sweden, June 12-16, 2023, Proceedings*, volume 13899 of *Lecture Notes in Computer Science*, pages 274–287. Springer, 2023. `doi:10.1007/978-3-031-33180-0\_21`.

[TMW15] Salvatore La Torre, Anca Muscholl, and Igor Walukiewicz. Safety of parametrized asynchronous shared-memory systems is almost always decidable. In Luca Aceto and David de Frutos-Escrig, editors, *26th International Conference on Concurrency Theory, CONCUR 2015, Madrid, Spain, September 1.4, 2015*, volume 42 of *LIPIcs*, pages 72–84. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015. `doi:10.4230/LIPICS.CONCUR.2015.72`.

[Zet15a] Georg Zetzsche. An approach to computing downward closures. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 440–451. Springer, 2015. `doi:10.1007/978-3-662-47666-6\_35`.

[Zet15b] Georg Zetzsche. Computing downward closures for stacked counter automata. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, volume 30 of *LIPIcs*, pages 743–756. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015. `doi:10.4230/LIPIcs.STACS.2015.743`.

[Zet16] Georg Zetzsche. The complexity of downward closure comparisons. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *Proc. of the 43rd International Colloquium on Automata, Languages and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 123:1–123:14, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Zet18] Georg Zetzsche. Separability by piecewise testable languages and downward closures beyond subwords. In Anuj Dawar and Erich Grädel, editors, *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 929–938. ACM, 2018. `doi:10.1145/3209108.3209201`.