Inclusion Problems for Patterns With a Bounded Number of Variables☆

Joachim Bremer[1], Dominik D. Freydenberger*

Goethe-University, Frankfurt am Main, Germany

Abstract

We study the inclusion problems for pattern languages that are generated by patterns with a bounded number of variables. This continues the work by Freydenberger and Reidenbach (Information and Computation 208 (2010)) by showing that restricting the inclusion problem to significantly more restricted classes of patterns preserves undecidability, at least for comparatively large bounds. For smaller bounds, we prove the existence of classes of patterns with complicated inclusion relations, and an open inclusion problem, that are related to the Collatz Conjecture. In addition to this, we give the first proof of the undecidability of the inclusion problem for NE-pattern languages that, in contrast to previous proofs, does not rely on the inclusion problem for E-pattern languages, and proves the undecidability of the inclusion problem for NE-pattern languages over binary and ternary alphabets.

1. Introduction

Patterns – finite strings that consist of *variables* and *terminals* – are compact and natural devices for the definition of formal languages. A pattern generates a word by a *substitution* of the variables with arbitrary strings of terminals from a fixed alphabet Σ (where all occurrences of a variable in the pattern must be replaced with the same word), and its language is the set of all words that can be obtained under substitutions. In a more formal manner, the language of a pattern can be understood as the set of all images under *terminal-preserving* morphisms; i. e., morphisms that map variables to terminal strings, and each terminal to itself. For example, the pattern $\alpha = x_1 x_1 \, \mathtt{a} \, \mathtt{b} \, x_2$ (where x_1 and x_2 are variables, and $\mathtt{a}$ and $\mathtt{b}$ are terminals) generates the language of all words that have a prefix that consists of a square, followed by the word $\mathtt{a}\,\mathtt{b}$.

☆A preliminary version of this article appeared at DLT 2010 [2].

*Corresponding author

Email addresses: `bremer@cs.uni-frankfurt.de` (Joachim Bremer), `freydenberger@em.uni-frankfurt.de` (Dominik D. Freydenberger)

[1]This author acknowledges the financial support by the German Research Foundation (DFG) under grant SCHW 837/3-3

The study of patterns in strings goes back to Thue [22] and is a central topic of combinatorics on words (cf. the survey by Choffrut and Karhumäki [4]), while the investigation of pattern languages was initiated by Angluin [1]. Angluin's definition of pattern languages permits only the use of *nonerasing* substitutions (hence, this class of pattern languages is called *NE-pattern languages*). Later, Shinohara [21] introduced *E-pattern languages* (E for 'erasing' or 'extended'), were erasing substitutions are permitted.

This small difference in the definitions leads to immense differences in the properties of these two classes. For example, while the equivalence problem for NE-pattern languages is trivially decidable, the equivalence problem for E-pattern languages is a hard open problem. Although both classes were first introduced in the context of *inductive inference* (which deals with the problem of learning patterns for given sets of strings, for a survey see Ng and Shinohara [17]), they have been widely studied in Formal Language Theory (cf. the surveys by Mitrana [14], Salomaa [20]). Due to their compact definition, patterns or their languages occur in numerous prominent areas of computer science and discrete mathematics, including *unavoidable patterns* (cf. Jiang et al. [9]), *practical regular expressions* (cf. Câmpeanu et al. [3]), or *word equations* and the *positive theory of concatenation* (cf. Choffrut and Karhumäki [4]).

One of the most notable results on pattern languages is the proof of the undecidability of the inclusion problem by Jiang et al. [10], a problem that was open for a long time and is of vital importance for the inductive inference of pattern languages. Unfortunately, this proof heavily depends on the availability of an unbounded number of terminals, which might be considered impractical, as pattern languages are mostly used in settings with fixed (or at least bounded) alphabets. But as shown by Freydenberger and Reidenbach [7], undecidability holds even if the terminal alphabet is bounded. As the proof by Jiang et al. and its modification by Freydenberger and Reidenbach require the number of variables of the involved patterns to be unbounded, we consider it a natural question whether the inclusion problems remain undecidable even if bounds are imposed on the number of variables in the pattern; especially as bounding the number of variables changes the complexity of the membership problem from NP-complete to P (cf. Ibarra et al. [8]). Similar restrictions have been studied in the theory of concatenation (cf. Durnev [5]).

Apart from potential uses in inductive inference or other areas, and the search for an approach that could provide the leverage needed to solve the equivalence problem for E-pattern languages, our main motivation for deeper research into the inclusion problems is the question how strongly patterns and their languages are connected. All known cases of (non-trivial) decidability of the inclusion problem for various classes of patterns rely on the fact that for these classes, inclusion is characterized by the existence of a terminal-preserving morphism mapping one pattern to the other. This is a purely syntactical condition that, although NP-complete (cf. Ehrenfeucht and Rozenberg [6]), can be straightforwardly verified. Finding cases of inclusion that are not covered by this condition, but still decidable, could uncover (or rule out) previously unknown phenomena, and be of immediate use for related areas of research.

Our results can be summarized as follows: We show that the inclusion problems for E- and NE-patterns with a bounded (but large) number of variables are indeed undecidable. For smaller bounds, we prove the existence of classes of patterns with complicated inclusion relations, and an open inclusion problem. Some of these inclusions can simulate iterations of the Collatz function, while others could (in principle) be used to settle an important part of the famous Collatz Conjecture. In contrast to the aforementioned previous proofs, our proof of the undecidability of the inclusion problem for NE-pattern languages is not obtained through a reduction of the inclusion problem for E-pattern languages. Apart from the technical innovation, this allows to prove the undecidability of the inclusion problem for NE-pattern languages over binary and ternary alphabets, which was left open by Freydenberger and Reidenbach.

2. Preliminaries

2.1. Basic Definitions and Pattern Languages

Let $\mathbb{N}_1 := \{1, 2, 3, \dots\}$ and $\mathbb{N}_0 := \mathbb{N}_1 \cup \{0\}$. The function div denotes the integer division, and mod its remainder. The symbols $\subseteq$, $\subset$, $\supseteq$ and $\supset$ refer to subset, proper subset, superset and proper superset relation, respectively. The symbol $\setminus$ denotes the set difference, and $\emptyset$ the empty set.

For an arbitrary alphabet A, a *string* (over A) is a finite sequence of symbols from A, and λ stands for the *empty string*. The symbol A^+ denotes the set of all nonempty strings over A, and $A^* := A^+ \cup \{\lambda\}$. For the *concatenation* of two strings w_1, w_2 we write $w_1 \cdot w_2$ or simply $w_1 w_2$. We say a string $v \in A^*$ is a *factor* of a string $w \in A^*$ if there are $u_1, u_2 \in A^*$ such that $w = u_1 v u_2$. If $u_1 = \lambda$ (or $u_2 = \lambda$), then v is a *prefix* of w (or a *suffix*, respectively).

For any alphabet A, a *language* L (over A) is a set of strings over A, i. e. $L \subseteq A^*$. A language L is *empty* if $L = \emptyset$; otherwise, it is *nonempty*.

The notation $|K|$ stands for the size of a set K or the length of a string K; the term $|w|_a$ refers to the number of occurrences of the symbol a in the string w. For any $w \in \Sigma^*$ and any $n \in \mathbb{N}_0$, w^n denotes the *n-fold concatenation of* w, with $w^0 := \lambda$. Furthermore, we use $\cdot$ and the regular operations $*$ and $+$ on sets and strings in the usual way.

For any alphabets A, B, a *morphism* is a function $h : A^* \to B^*$ that satisfies $h(vw) = h(v)h(w)$ for all $v, w \in A^*$. A morphism $h : A^* \to B^*$ is said to be *nonerasing* if $h(a) \neq \lambda$ for all $a \in A$. For any string $w \in C^*$, where $C \subseteq A$ and $|w|_a \geq 1$ for every $a \in C$, the morphism $h : A^* \to B^*$ is called a *renaming (of w)* if $h : C^* \to B^*$ is injective and $|h(a)| = 1$ for every $a \in C$.

Let Σ be a (finite or infinite) alphabet of so-called *terminals* and X an infinite set of *variables* with $\Sigma \cap X = \emptyset$. We normally assume $\{\mathtt{a}, \mathtt{b}, \dots\} \subseteq \Sigma$ and $\{x_1, x_2, x_3 \dots\} \subseteq X$. A *pattern* is a string over $\Sigma \cup X$, a *terminal-free pattern* is a string over X and a *terminal-string* is a string over Σ. For any pattern α, we refer to the set of variables in α as $\mathrm{var}(\alpha)$. The set of all patterns over $\Sigma \cup X$ is denoted by Pat_Σ; the set of all terminal-free patterns is denoted by $\mathrm{Pat}_{\mathtt{tf}}$. For every $n \geq 0$, let $\mathrm{Pat}_{n,\Sigma}$ denote the set of all patterns over Σ that contain at most n variables; that is, $\mathrm{Pat}_{n,\Sigma} := \{\alpha \in \mathrm{Pat}_\Sigma \mid |\mathrm{var}(\alpha)| \leq n\}$.

A morphism $\sigma : (\Sigma \cup X)^* \to (\Sigma \cup X)^*$ is called *terminal-preserving* if $\sigma(a) = a$ for every $a \in \Sigma$. A terminal-preserving morphism $\sigma : (\Sigma \cup X)^* \to \Sigma^*$ is called a *substitution*. The *E-pattern language* $L_{\mathrm{E},\Sigma}(\alpha)$ of α is given by

$$L_{\mathrm{E},\Sigma}(\alpha) := \{\sigma(\alpha) \mid \sigma : (\Sigma \cup X)^* \to \Sigma^* \text{ is a substitution}\},$$

and the *NE-pattern language* $L_{\mathrm{NE},\Sigma}(\alpha)$ of a pattern $\alpha \in \mathrm{Pat}_\Sigma$ is given by

$$L_{\mathrm{NE},\Sigma}(\alpha) := \{\sigma(\alpha) \mid \sigma : (\Sigma \cup X)^* \to \Sigma^* \text{ is a nonerasing substitution}\}.$$

If the intended meaning is clear, we write $L(\alpha)$ instead of $L_{\mathrm{E},\Sigma}(\alpha)$ or $L_{\mathrm{NE},\Sigma}(\alpha)$ for any $\alpha \in \mathrm{Pat}_\Sigma$. Furthermore, let ePAT_Σ denote the class of all E-pattern languages over Σ, and nePAT_Σ the class of all NE-pattern languages over Σ. Likewise, we define $\mathrm{ePAT}_{\mathrm{tf},\Sigma}$ as the class of all $L_{\mathrm{E},\Sigma}(\alpha)$ with $\alpha \in \mathrm{Pat}_{\mathrm{tf}}$, and, for any $n \geq 0$, $\mathrm{ePAT}_{n,\Sigma}$ as the class of all $L_{\mathrm{E},\Sigma}(\alpha)$ with $\alpha \in \mathrm{Pat}_{n,\Sigma}$. The classes $\mathrm{nePAT}_{\mathrm{tf},\Sigma}$ and $\mathrm{nePAT}_{n,\Sigma}$ are defined accordingly. Let P_1, P_2 be two classes of patterns, and $\mathrm{PAT}_1, \mathrm{PAT}_2$ be the corresponding classes of pattern languages (either the class of all E-pattern languages or the class of all NE-pattern languages over some alphabet Σ that are generated by patterns from P_1 or P_2). We say that the *inclusion problem for* PAT_1 *in* PAT_2 *is decidable* if there exists a total computable function χ such that, for every pair of patterns $\alpha \in P_1$ and $\beta \in P_2$, χ decides on whether or not $L(\alpha) \subseteq L(\beta)$. If no such function exists, this inclusion problem is *undecidable*. If both classes of pattern languages are the same class $\mathrm{PAT}_{\star,\Sigma}$, we simple refer to the *inclusion problem of* $\mathrm{PAT}_{\star,\Sigma}$.

2.2. The Universal Turing Machine U

Let U be the universal Turing machine $U_{15,2}$ with 2 symbols and 15 states described by Neary and Woods [16]. This machine has the state set $Q = \{q_1, \ldots, q_{15}\}$ and operates on the tape alphabet $\Gamma = \{0, 1\}$ (where 0 is the blank symbol). Its transition function $\delta : \Gamma \times Q \to (\Gamma \times \{L, R\} \times Q) \cup \mathrm{HALT}$ is depicted in Figure 1.

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8
0	$0, R, q_2$	$1, R, q_3$	$0, L, q_7$	$0, L, q_6$	$1, R, q_1$	$1, L, q_4$	$0, L, q_8$	$1, L, q_9$
1	$1, R, q_1$	$1, R, q_1$	$0, L, q_5$	$1, L, q_5$	$1, L, q_4$	$1, L, q_4$	$1, L, q_7$	$1, L, q_7$

	q_9	q_{10}	q_{11}	q_{12}	q_{13}	q_{14}	q_{15}
0	$0, R, q_1$	$1, L, q_{11}$	$0, R, q_{12}$	$0, R, q_{13}$	$0, L, q_2$	$0, L, q_3$	$0, R, q_{14}$
1	$1, L, q_{10}$	HALT	$1, R, q_{14}$	$1, R, q_{12}$	$1, R, q_{12}$	$0, R, q_{15}$	$1, R, q_{14}$

Figure 1: The transition table of the universal Turing machine U which is defined in Section 2.2. This machine is due to Neary and Woods [16] and is, to the author's knowledge, the smallest currently known universal Turing machine over a two letter tape alphabet.

In order to discuss configurations of U, we adopt the following conventions. The tape content of any configuration of U is characterized by the two infinite sequences $t_L = (t_{L,n})_{n \geq 0}$ and $t_R = (t_{R,n})_{n \geq 0}$ over Γ. Here, t_L describes the content of what we shall call the *left side of the tape*, the infinite word that starts at the position of the machine's head and extends to the left. Likewise, t_R describes the *right side of the tape*, the infinite word that starts immediately to the right of the head and extends to the right (cf. Figure 2).

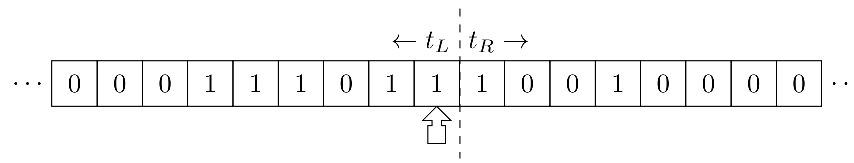

Figure 2: An illustration of tape words of some configuration of the universal Turing machine U (as defined in Section 2.2). The arrow below the tape symbolizes the position of the head, while the dashed lines show the borders between the left tape side and the right tape side. Assuming that all tape cells that are not shown contain 0, we observe the left tape word $t_L = 1101110^\omega$ and the right tape word $t_R = 10010^\omega$.

Encoding Computations of U. Next, we define the function $\mathrm{e} : \Gamma \to \mathbb{N}_0$ as $\mathrm{e}(0){:=}0$ and $\mathrm{e}(1){:=}1$, and extend this to an encoding of infinite sequences $t = (t_n)_{n\geq 0}$ over Γ by $\mathrm{e}(t){:=}\sum_{i=0}^{\infty} 2^i \,\mathrm{e}(t_i)$. As we consider only configurations where all but finitely many cells of the tape consist of the blank symbol 0 (which is encoded as 0), $\mathrm{e}(t)$ is always finite and well-defined. Note that for every side t of the tape, $\mathrm{e}(t) \bmod 2$ returns the encoding of the symbol that is closest to the head (the symbol under the head for t_L, and the symbol to the right of the head for t_R). Furthermore, each side can be lengthened or shortened by multiplying or dividing (respectively) its encoding $\mathrm{e}(t)$ by 2. The encodings $\mathrm{enc}_{\mathrm{E}}$ and $\mathrm{enc}_{\mathrm{NE}}$ of configurations of U are defined by

$$\mathrm{enc}_{\mathrm{E}}(q_i, t_L, t_R){:=}0\; 0^{\mathrm{e}(t_R)}\#0\; 0^{\mathrm{e}(t_L)}\#0^i,$$
$$\mathrm{enc}_{\mathrm{NE}}(q_i, t_L, t_R){:=}0^7\; 0^{\mathrm{e}(t_R)}\#0^7\; 0^{\mathrm{e}(t_L)}\#0^{i+6},$$

for every configuration (q_i, t_L, t_R). Note that both functions are almost identical; the only difference is that $\mathrm{enc}_{\mathrm{NE}}$ adds six additional occurrences of 0 to each of the three continuous blocks of 0.

We extend each of these encodings to an encoding of finite sequences of configurations $C = (C_i)_{i=1}^n$ by $\mathrm{enc}(C){:=}\#\#\,\mathrm{enc}(C_1)\#\#\ldots\#\#\,\mathrm{enc}(C_n)\#\#$ for $\mathrm{enc} = \mathrm{enc}_{\mathrm{E}}$ or $\mathrm{enc} = \mathrm{enc}_{\mathrm{NE}}$. Let I be any configuration of U. A *valid computation from I* is a finite sequence $C = (C_i)_{i=1}^n$ (with $n \geq 2$) of configurations of U such that $C_1 = I$, C_n is a halting configuration, and C_{i+1} is a valid successor configuration of C_i for every i with $1 \leq i < n$. We adopt the convention that any possible configuration where both tape sides have a finite value under e is a valid successor configuration of a halting configuration. This extended definition of succession does not change the acceptance behavior of U. Finally, let

$$\mathrm{VALC}_{\mathrm{E}}(I){:=}\{\mathrm{enc}_{\mathrm{E}}(C) \mid C \text{ is a valid computation from } I\},$$
$$\mathrm{VALC}_{\mathrm{NE}}(I){:=}\{\mathrm{enc}_{\mathrm{NE}}(C) \mid C \text{ is a valid computation from } I\}.$$

Each of the two sets is nonempty if and only if U accepts the input of the initial configuration I, and can thus be used to decide the halting problem of U. As U is universal, there can be no recursive function that, on input I, decides whether $\mathrm{VALC}_{\mathrm{E}}(I)$ is empty or not (the same holds for $\mathrm{VALC}_{\mathrm{NE}}(I)$).

2.3. Collatz Iterations

The Collatz function $\mathcal{C} : \mathbb{N}_1 \to \mathbb{N}_1$ is defined by $\mathcal{C}(n){:=}\frac{1}{2}n$ if n is even, and $\mathcal{C}(n) := 3n + 1$ if n is odd. For any $i \geq 0$ and any $n \geq 1$, let $\mathcal{C}^0(n){:=}n$ and $\mathcal{C}^{i+1}(n){:=}\mathcal{C}(\mathcal{C}^i(n))$. A number n *leads* $\mathcal{C}$ *into a cycle* if there are i, j with $1 \leq i < j$ and $\mathcal{C}^i(n) = \mathcal{C}^j(n)$. The cycle is *non-trivial* if $\mathcal{C}^k(n) \neq 1$ for every $k \geq 0$; otherwise, it is the *trivial cycle.*

The Collatz Conjecture states that every natural number leads $\mathcal{C}$ into the trivial cycle $4, 2, 1$. Regardless of the considerable effort spent on this problem (see the bibliographies by Lagarias [11, 12]), the conjecture remains unsolved, as the iterated function often behaves rather unpredictably. For this reason, iterations of the Collatz function have been studied in the research of small Turing machines. Margenstern [13] conjectures that every class of Turing machines (as characterized by the number of states and symbols) that contains a machine that is able to simulate the iteration of the Collatz function, also contains a machine that has an undecidable halting problem.

Encoding Collatz Iterations. Similar to the definition of $\mathrm{VALC}_{\mathrm{E}}(I)$ and $\mathrm{VALC}_{\mathrm{NE}}(I)$, we encode those iterations of the Collatz function that lead to the number 1 (and thus, to the trivial cycle) in languages over the alphabet $\{0, \#\}$. For every $N \in \mathbb{N}_1$, let

$$\mathrm{TRIV}_{\mathrm{E}}(N){:=}\{\#0^{\mathcal{C}^0(N)}\#0^{\mathcal{C}^1(N)}\#\dots\#0^{\mathcal{C}^n(N)}\# \mid n \geq 1, \mathcal{C}^n(N) = 1\},$$
$$\mathrm{TRIV}_{\mathrm{NE}}(N){:=}\{\#0^{6+\mathcal{C}^0(N)}\#0^{6+\mathcal{C}^1(N)}\#\dots\#0^{6+\mathcal{C}^n(N)}\# \mid n \geq 1, \mathcal{C}^n(N) = 1\}.$$

By definition, $\mathrm{TRIV}_{\mathrm{E}}(N)$ (and $\mathrm{TRIV}_{\mathrm{NE}}(N)$) are empty if and only if N does not lead $\mathcal{C}$ into the trivial cycle. As we shall see, our constructions are able to express an even stronger problem, the question whether there are any numbers that lead $\mathcal{C}$ to a non-trivial cycle. We define $\mathrm{NTCC}_{\mathrm{E}}$ as the set of all strings $\#0^{\mathcal{C}^0(N)}\#0^{\mathcal{C}^1(N)}\#\dots\#0^{\mathcal{C}^n(N)}\#$, where $n, N \geq 1$, $\mathcal{C}^i(N) \neq 1$ for all $i \in \{0, \dots, n\}$, and $\mathcal{C}^j(N) = \mathcal{C}^n(N)$ for some $j < n$. Analogously, $\mathrm{NTCC}_{\mathrm{NE}}$ is defined to be the set of all strings $\#0^{6+\mathcal{C}^0(N)}\#0^{6+\mathcal{C}^1(N)}\#\dots\#0^{6+\mathcal{C}^n(N)}\#$, with the same restrictions on n and N. Obviously, both sets are nonempty if and only if there exist non-trivial cycles in the iteration of $\mathcal{C}$. This is one of the two possible cases that would disprove the Collatz Conjecture, the other being the existence of a number N with $\mathcal{C}^i(N) \neq \mathcal{C}^j(N)$ for all $i \neq j$.

3. Main Results

In this section, we study the inclusion problems of various classes of pattern languages generated by patterns with a bounded number of variables.

As shown by Jiang et al. [10], the *general inclusion problem for pattern languages* is undecidable, both in the case of E- and NE-patterns:

Theorem 3.1 (Jiang et al. [10]). *Let $Z \in \{\mathrm{E}, \mathrm{NE}\}$. There is no total computable function χ_Z which, for every alphabet Σ and for every pair of patterns $\alpha, \beta \in \mathrm{Pat}_\Sigma$, decides on whether or not $L_{Z,\Sigma}(\alpha) \subseteq L_{Z,\Sigma}(\beta)$.*

The proof for the E-case uses an involved construction that relies heavily on the unboundedness of the terminal alphabet Σ. For the NE-case, Jiang et al. give a complicated reduction of the inclusion problem for ePAT_Σ to the inclusion problem for $\mathrm{nePAT}_{\Sigma_2}$, where Σ_2 is an alphabet with two additional terminals. As shown by Freydenberger and Reidenbach [7], the inclusion problem remains undecidable for most cases of a fixed terminal alphabet:

Theorem 3.2 (Freydenberger and Reidenbach [7]). *Let Σ be a finite alphabet. If $|\Sigma| \geq 2$, the inclusion problem of ePAT_Σ is undecidable. If $|\Sigma| \geq 4$, the inclusion problem of nePAT_Σ is undecidable.*

The proof for the E-case consists of a major modification of the construction for the general inclusion problem for E-pattern languages, and relies on the presence of an unbounded number of variables in one of the patterns. The NE-case of the result follows from the same reduction as in the proof of Theorem 3.1 (thus, the difference in $|\Sigma|$), and also relies on an unbounded number of variables.

As patterns with an arbitrarily large number of variables might seem somewhat artificial for many applications, we consider it natural to bound this number in order to gain decidability of (or at least further insights on) the inclusion of pattern languages. We begin our considerations with an observation from two classical papers on pattern languages:

Theorem 3.3 (Angluin [1], Jiang et al. [9]). *The inclusion problem for nePAT_Σ in $\mathrm{nePAT}_{1,\Sigma}$ and the inclusion problem for ePAT_Σ in $\mathrm{ePAT}_{1,\Sigma}$ are decidable.*

The proofs for both cases of this theorem rely on the following sufficient condition for inclusion of pattern languages:

Theorem 3.4 (Jiang et al. [9], Angluin [1]). *Let Σ be an alphabet and $\alpha, \beta \in \mathrm{Pat}_\Sigma$. If there is a terminal-preserving morphism $\phi : (\Sigma \cup X)^* \to (\Sigma \cup X)^*$ with $\phi(\beta) = \alpha$, then $L_{\mathrm{E},\Sigma}(\alpha) \subseteq L_{\mathrm{E},\Sigma}(\beta)$. If ϕ is also nonerasing, then $L_{\mathrm{NE},\Sigma}(\alpha) \subseteq L_{\mathrm{NE},\Sigma}(\beta)$.*

In fact, the proofs of both parts of Theorem 3.3 show that, for every alphabet Σ and all patterns $\alpha \in \mathrm{Pat}_\Sigma$, $\beta \in \mathrm{Pat}_{1,\Sigma}$, $L(\alpha) \subseteq L(\beta)$ holds *if and only if* there is a terminal-preserving (and, in the NE-case, nonerasing) morphism ϕ with $\phi(\beta) = \alpha$. As the existence of such a morphism is a decidable property (although in general NP-complete, cf. Ehrenfeucht and Rozenberg [6]), the respective inclusion problems for these classes are decidable.

There are numerous other classes of pattern languages where this condition is not only sufficient, but characteristic; e. g. the terminal-free E-pattern languages (cf. Jiang et al. [10]), some of their generalizations (cf. Ohlebusch and Ukkonen [18]), and pattern languages over infinite alphabets (cf. Freydenberger and Reidenbach [7]). As far as we know, all non-trivial decidability results for pattern languages over non-unary alphabets rely on this property[2]. Contrariwise, the existence of patterns where inclusion is not characterized by the existence of an appropriate morphism between them is a necessary condition for an undecidable inclusion problem for this class.

The same phenomenon as in Theorem 3.3 does not occur if we swap the bounds. For the nonerasing case, this is illustrated by the following example:

Example 3.5 (Reidenbach [19], Example 3.2)**.** *Let* $\Sigma = \{\mathtt{a}_1, \dots, \mathtt{a}_n\}$ *with* $n \geq 2$*, and consider the pattern* $\alpha_n := x\, \mathtt{a}_1\, x\, \mathtt{a}_2\, x \dots x\, \mathtt{a}_n\, x$*,* $\beta := xyyz$*. Then there is no terminal-preserving morphism* ϕ *with* $\phi(\beta) = \alpha_n$*, but every word from* $L_{\mathrm{NE},\Sigma}(\alpha_n)$ *contains an inner square. Thus,* $L_{\mathrm{NE},\Sigma}(\alpha_n) \subseteq L_{\mathrm{NE},\Sigma}(\beta)$*.* ◇

Using a less straightforward approach, we observe an even tighter bound:

Proposition 3.6 (Angluin [1])**.** *For every finite alphabet* Σ*, there exist patterns* $\alpha \in \mathrm{Pat}_{1,\Sigma}$ *and* $\beta \in \mathrm{Pat}_{2,\Sigma}$ *such that* $L_{\mathrm{NE},\Sigma}(\alpha) \subseteq L_{\mathrm{NE},\Sigma}(\beta)$*, but there is no nonerasing terminal-preserving morphism* $\phi : (\Sigma \cup X)^+ \to (\Sigma \cup X)^+$ *with* $\phi(\beta) = \alpha$*.*

Proof. The proof for the case of binary terminal alphabets is due to Angluin [1] (Example 3.8). As Angluin only sketches the extension to ternary terminal alphabets and mentions that the construction can be extended to larger alphabets in a straightforward way, we give the whole proof.

First, we define the infinite terminal alphabet $\Sigma_\infty := \{\mathtt{a}_1, \mathtt{a}_2, \dots\}$, where all $\mathtt{a}_i$ are pairwise different. Next, we define an infinite sequence of patterns $(\hat{\alpha}_i)_{i=1}^{\infty}$ by

$$\hat{\alpha}_1 := \mathtt{a}_1\, x,$$
$$\hat{\alpha}_{i+1} := \hat{\alpha}_i\, \mathtt{a}_{i+1}\, \hat{\alpha}_i x$$

for every $i \geq 1$. In addition to this, we define a second sequence $(\alpha_i)_{i=1}^{\infty}$ by $\alpha_i := \hat{\alpha}_i\, \mathtt{a}_i$ for every $i \geq 1$. Thus, the first three patterns in the two sequences are

$$\begin{aligned} &\hat{\alpha}_1 := \mathtt{a}_1\, x, && \alpha_1 := \mathtt{a}_1\, x\, \mathtt{a}_1, \\ &\hat{\alpha}_2 := \mathtt{a}_1\, x\, \mathtt{a}_2\, \mathtt{a}_1\, xx, && \alpha_2 := \mathtt{a}_1\, x\, \mathtt{a}_2\, \mathtt{a}_1\, xx\, \mathtt{a}_2, \\ &\hat{\alpha}_3 := \mathtt{a}_1\, x\, \mathtt{a}_2\, \mathtt{a}_1\, xx\, \mathtt{a}_3\, \mathtt{a}_1\, x\, \mathtt{a}_2\, \mathtt{a}_1\, xxx, && \alpha_3 := \mathtt{a}_1\, x\, \mathtt{a}_2\, \mathtt{a}_1\, xx\, \mathtt{a}_3\, \mathtt{a}_1\, x\, \mathtt{a}_2\, \mathtt{a}_1\, xxx\, \mathtt{a}_3\,. \end{aligned}$$

We shall now show that, for every alphabet $\Sigma = \{\mathtt{a}_1, \dots, \mathtt{a}_n\} \subset \Sigma_\infty$ (with $n \geq 1$), the patterns α_n and $\beta := xxy$ prove the claim – i. e., $L_{\mathrm{NE},\Sigma}(\alpha_n) \subseteq L_{\mathrm{NE},\Sigma}(\beta)$,

[2]Non-trivial meaning that the involved classes are neither finite, nor restricted in some artificial way that leads to trivial decidability.

but there is no nonerasing terminal-preserving morphism ϕ with $\phi(\beta) = \alpha$. The proof relies on the following two claims:

Claim 1. For every $n \geq 1$, no nonempty prefix of $\hat{\alpha}_n$ is a square.

Proof of Claim 1. We prove this claim by induction. For $n = 1$, $\hat{\alpha}_n = \mathtt{a}_1\, x$. The only nonempty prefix of $\hat{\alpha}_n$ is $\hat{\alpha}_n$ itself, and this pattern is not a square.

Now assume the claim holds for some $n \geq 1$ (i. e., no nonempty prefix of $\hat{\alpha}_n$ is a square). By definition, $\hat{\alpha}_{n+1} = \hat{\alpha}_n\, \mathtt{a}_{n+1}\, \hat{\alpha}_n x$. Due to the definition of $\hat{\alpha}_n$, we know that the letter $\mathtt{a}_{n+1}$ does not occur therein, and by the induction assumption, no nonempty prefix of $\hat{\alpha}_n$ is a square. Thus, the claim holds for $\hat{\alpha}_{n+1}$ as well. □ (Claim 1)

In order to state the next claim, for every $i \geq 1$, we define S_i to be the set of all nonerasing substitutions $\sigma : (\Sigma_\infty \cup X)^+ \to (\Sigma_\infty)^+$ for which the leftmost letter of $\sigma(x)$ is $\mathtt{a}_i$.

Claim 2. For every $n \geq 1$, every i with $1 \leq i \leq n$ and every $\sigma \in S_i$, $\sigma(\hat{\alpha}_n)$ has a nonempty prefix that is a square.

Proof of Claim 2. Again, we show the claim by induction. First, let $n = 1$. In this case, we only need to consider the case of $\sigma \in S_1$. For every such σ, there is a $w \in (\Sigma_\infty)^*$ such that $\sigma(x) = \mathtt{a}_1\, w$. Accordingly, as $\sigma(\hat{\alpha}_1) = \mathtt{a}_1\, \mathtt{a}_1\, x$, the claim holds.

Now assume that, for some $n \geq 1$ and all i with $1 \leq i \leq n$, $\sigma(\hat{\alpha}_n)$ has a nonempty prefix that is a square. As $\hat{\alpha}_n$ is a prefix of $\hat{\alpha}_{n+1}$, this implies that, for every $\sigma \in S_1$ with $1 \leq i \leq n$, $\sigma(\hat{\alpha}_{n+1})$ has a nonempty square as a prefix. Therefore, we only need to consider the substitutions $\sigma \in S_{n+1}$. For every such σ, there is a $w \in (\Sigma_\infty)^*$ such that $\sigma(x) = \mathtt{a}_{n+1}\, w$, and

$$\begin{aligned}\sigma(\hat{\alpha}_{n+1}) &= \sigma(\hat{\alpha}_n\, \mathtt{a}_{n+1}\, \hat{\alpha}_n x)\\ &= \sigma(\hat{\alpha}_n)\, \mathtt{a}_{n+1}\, \sigma(\hat{\alpha}_n)\, \mathtt{a}_{n+1}\, w.\end{aligned}$$

Thus, $(\sigma(\hat{\alpha}_n)\, \mathtt{a}_{n+1})^2$ is a (nonempty) prefix of $\sigma(\hat{\alpha}_{n+1})$. □ (Claim 2)

Now, for every $n \geq 1$, consider the terminal alphabet $\Sigma{:=}\{\mathtt{a}_1, \ldots, \mathtt{a}_n\}$ and the patterns $\alpha{:=}\alpha_n$ and $\beta{:=}xxy$ (where x and y are distinct variables).

For every word $w \in L_{\mathrm{NE},\Sigma}(\alpha)$, there is a nonerasing substitution $\sigma : (\Sigma \cup X)^+ \to \Sigma^+$ with $\sigma(\alpha) = w$. Therefore, $\sigma \in S_i$ for some i with $1 \leq i \leq n$, depending on the leftmost letter of $\sigma(x)$. By Claim 2, $\sigma(\hat{\alpha}_n)$ has a nonempty prefix that is a square; i. e., there are a $u \in \Sigma^+$ and a $v \in \Sigma^*$ such that $w = uuv$. We now define the substitution $\tau : (\Sigma \cup X)^+ \to \Sigma^+$ by $\tau(x){:=}u$ and $\tau(y){:=}v\, \mathtt{a}_n$. Thus, $\tau(\beta) = uuv\, \mathtt{a}_n = \sigma(\hat{\alpha}_n)\, \mathtt{a}_n = \sigma(\alpha_n) = w$, and $L_{\mathrm{NE},\Sigma}(\alpha) \subseteq L_{\mathrm{NE},\Sigma}(\beta)$.

On the other hand, assume that there is a nonerasing morphism $\phi : X^+ \to (\Sigma \cup X)^+$ with $\phi(\beta) = \alpha = \hat{\alpha}_n\, \mathtt{a}_n$. As τ is nonerasing, the rightmost letter of $\tau(y)$ must be $\mathtt{a}_n$. More formally, there is some $\gamma \in (\Sigma \cup X)^*$ with $\tau(y) = \gamma\, \mathtt{a}_n$. Thus, $\hat{\alpha}_n = (\tau(x))^2\gamma$; by definition of τ, this means that $\hat{\alpha}_n$ has a nonempty square as a prefix, which contradicts Claim 1. Therefore, no such ϕ exists. □

Thus, regardless of the size of $|\Sigma|$, even the inclusion problem of $\mathrm{nePAT}_{1,\Sigma}$ in $\mathrm{nePAT}_{3,\Sigma}$ is too complex to be characterized by the existence of a nonerasing terminal-preserving morphism between the patterns. A similar phenomenon can be observed for E-pattern languages:

Proposition 3.7. *For every finite alphabet Σ with $|\Sigma| \geq 2$, there are patterns $\alpha \in \mathrm{Pat}_{1,\Sigma}$ and $\beta \in \mathrm{Pat}_{2|\Sigma|+2,\Sigma}$ such that $L_{\mathrm{E},\Sigma}(\alpha) \subseteq L_{\mathrm{E},\Sigma}(\beta)$, but there is no terminal-preserving morphism $\phi : (\Sigma \cup X)^* \to (\Sigma \cup X)^*$ with $\phi(\beta) = \alpha$.*

Proof. The patterns α and β can be straightforwardly obtained from the patterns in the proof of Theorem 6 in [7], by replacing each variable in α with a single variable x, and removing a common prefix.

Let $\Sigma = \{\mathtt{a}_1, \dots, \mathtt{a}_n\}$ (where all $\mathtt{a}_i$ are distinct, i.e., $|\Sigma| = n$). Let $m{:=}n$ if n is odd, and $m{:=}n+1$ if n is even. If n is even, we also define $\mathtt{a}_m := \mathtt{a}_n$.

Next, we define

$$\begin{aligned}\alpha &{:=} \mathtt{a}_1\, x\, \mathtt{a}_1\, x \cdot \mathtt{a}_2\, x\, \mathtt{a}_2\, x \cdot \ldots \cdot \mathtt{a}_m\, x\, \mathtt{a}_m\, x,\\ \beta &{:=} \mathtt{a}_1\, \beta_1\, \mathtt{a}_1\, z_1 \cdot \mathtt{a}_2\, \beta_2\, \mathtt{a}_2\, z_2 \cdot \ldots \cdot \mathtt{a}_m\, \beta_m\, \mathtt{a}_m\, z_m,\end{aligned}$$

with, for $1 \leq i \leq m$,

$$\beta_i {:=} \begin{cases} y_i y_{i+1} & \text{if } 1 \leq i < m,\\ y_n y_1 & \text{if } i = m,\end{cases}$$

where $y_1, z_1, \dots, y_m, z_m$ are pairwise distinct variables.

In order to show $L_{\mathrm{E},\Sigma}(\alpha) \subseteq L_{\mathrm{E},\Sigma}(\beta)$, we prove that, for every substitution σ, there is a substitution τ with $\tau(\beta) = \sigma(\alpha)$. If $\sigma(x) = \lambda$, it is easy to see that $\sigma(\alpha)$ can be created from β by erasing all variables. Therefore, we can safely assume $\sigma(x) = \mathtt{a}_j\, u$ with $1 \leq j \leq n$ and $u \in \Sigma^*$.

We define the substitution τ by

$$\tau(z_i) {:=} \begin{cases} \sigma(x) & \text{if } i \neq j,\\ u\, \mathtt{a}_j\, \sigma(x) & \text{if } i = j,\end{cases}$$

for every $z_i \in \mathrm{var}(\beta)$, and by

$$\tau(y_i) {:=} \begin{cases} \lambda & \text{if } i \in \mathrm{ERASE}_j,\\ a_j u & \text{if } i \notin \mathrm{ERASE}_j,\end{cases}$$

for every $y_i \in \mathrm{var}(\beta)$, where the set $\mathrm{ERASE}_j \subset \mathrm{var}(\beta)$ is defined as

$$\mathrm{ERASE}_j := \{y_s \in \mathrm{var}(\beta) \mid s = j - 2i \text{ or } s = j + 1 + 2i \text{ for some } i \geq 0\}.$$

Note that, due to our definition of ERASE_j and τ, $\tau(\beta_j) = \lambda$ and $\tau(\beta_i) = \sigma(x)$ for every $i \neq j$ hold, as ERASE_j contains exactly those x_s with either $s \leq j$, and s has the same parity as j, or $s > j$, where s and j have different parities.

In order to prove $\phi(\beta) = \sigma(\alpha)$, it suffices to show that $\phi(\mathtt{a}_i\, \beta_i\, \mathtt{a}_i\, z_i) = \sigma(\mathtt{a}_i\, x\, \mathtt{a}_i\, x)$ for every i with $1 \leq i \leq m$ – then the claim follows by definition of α and β.

For every i with $1 \leq i \leq m$ and $i \neq j$, we use $\tau(\beta_i) = \sigma(x)$ to conclude

$$\begin{aligned}\tau(\mathtt{a}_i\, \beta_i\, \mathtt{a}_i\, z_i) &= \mathtt{a}_i\, \sigma(x)\, \mathtt{a}_i\, \sigma(x)\\ &= \sigma(\mathtt{a}_i\, x\, \mathtt{a}_i\, x).\end{aligned}$$

Likewise, for the special case of $i = j$, $\tau(\beta_j) = \lambda$ leads to

$$\begin{aligned}\tau(\mathtt{a}_j\, \beta_j\, \mathtt{a}_j\, z_j) &= \mathtt{a}_j \cdot \lambda \cdot \mathtt{a}_j\, u \cdot \mathtt{a}_j\, \sigma(x)\\ &= \mathtt{a}_j\, \sigma(x)\, \mathtt{a}_j\, \sigma(x)\\ &= \sigma(\mathtt{a}_j\, x\, \mathtt{a}_j\, x).\end{aligned}$$

Thus, $\phi(\beta) = \sigma(\alpha)$, and – as σ was chosen freely – $L_{\mathrm{E},\Sigma}(\alpha) \subseteq L_{\mathrm{E},\Sigma}(\beta)$.

We proceed to show that there is no terminal-preserving morphism $\phi : (\Sigma \cup X)^* \to (\Sigma \cup X)^*$ with $\phi(\beta) = \alpha$. Assume to the contrary that there is a terminal-preserving morphism ϕ with $\phi(\beta) = \alpha$. As α and β contain exactly the same occurrences of terminals, $\phi(\beta_i) = x$ and $\phi(z_i) = x$ must hold for every $i \in \{1, \dots, m\}$. We define $\beta' := \beta_1 \cdot \ldots \cdot \beta_m$, and observe $\phi(\beta') = x^m$. By definition of β_i, $|\beta'|_{z_i} = 2$ for $1 \leq i \leq m$, and thus, $|\beta'|$ is even. This contradicts the fact that m (and, thus, $|x^m|$) is odd by definition. □

The proof also shows that, if Σ has an odd number of letters, the bound on the number of variables in the second class of patterns can be lowered to $2|\Sigma|$. We do not know whether this lower bound is strict, or if there are patterns $\alpha \in \mathrm{Pat}_{1,\Sigma}$, $\beta \in \mathrm{Pat}_{n,\Sigma}$ with $n < 2|\Sigma|$ such that $L_{\mathrm{E},\Sigma}(\alpha) \subseteq L_{\mathrm{E},\Sigma}(\beta)$, but there is no terminal-preserving morphism mapping β to α.

For $|\Sigma| = 2$, according to Proposition 3.7, the inclusion of $\mathrm{ePAT}_{1,\Sigma}$ in $\mathrm{ePAT}_{6,\Sigma}$ is not characterized by the existence of such a morphism. As this bound (and the bound on NE-patterns from Example 3.5) are the lowest known bounds for 'morphism-free' inclusion, we want to emphasize the following problem:

Open Problem 1. *Let* $|\Sigma| = 2$*. Is the inclusion problem of* $\mathrm{ePAT}_{1,\Sigma}$ *in* $\mathrm{ePAT}_{6,\Sigma}$ *decidable? Is the inclusion problem of* $\mathrm{nePAT}_{1,\Sigma}$ *in* $\mathrm{nePAT}_{3,\Sigma}$ *decidable?*

In principle, both inclusion problems might be undecidable; but comparing these bounds to the ones in the following results, this seems somewhat improbable, and suggests that if these problems are undecidable, the proof would need to be far more complicated than the proofs in the present paper. On the other hand, these classes are promising candidates for classes of pattern languages where the inclusion is decidable, but not characterized by the existence of an appropriate morphism.

As evidenced by our first two main theorems, bounding the number of variables preserves the undecidability of the inclusion problem:

Theorem 3.8. *Let* $|\Sigma| = 2$*. The following problems are undecidable:*

1. *The inclusion problem of* $\mathrm{ePAT}_{3,\Sigma}$ *in* $\mathrm{ePAT}_{2854,\Sigma}$*,*
2. *the inclusion problem of* $\mathrm{ePAT}_{2,\Sigma}$ *in* $\mathrm{ePAT}_{2860,\Sigma}$*.*

Theorem 3.9. *Let $|\Sigma| = 2$. The following problems are undecidable:*

1. *The inclusion problem of* $\mathrm{nePAT}_{3,\Sigma}$ *in* $\mathrm{nePAT}_{2554,\Sigma}$*,*
2. *the inclusion problem of* $\mathrm{nePAT}_{2,\Sigma}$ *in* $\mathrm{nePAT}_{2558,\Sigma}$*.*

Note that the cases of all larger (finite) alphabets are handled in Section 5.1. The bounds presented in these two theorems are not optimal. Through additional effort and some encoding tricks, it is possible to reduce each bound on the number of variables in the second pattern by a few hundred variables. As the resulting number would still be far away from the bounds presented in the theorems further down in this section, we felt that these optimizations would only add additional complexity to the proofs, without providing deeper insight, and decided to give only the less optimal bounds present above.

The proofs for both theorems use the same basic approach as the proofs of the E-case in Theorems 3.1 and 3.2. We show that, for a given configuration I of U, one can effectively construct patterns α, β in the appropriate classes of patterns such that $L(\alpha) \subseteq L(\beta)$ if and only if U halts after starting in I. As this would decide the halting problem of the universal Turing machine U, the inclusion problems must be undecidable.

For the E-case, we show this using a nontrivial but comparatively straightforward modification of the proof for the E-case of Theorem 3.2. As this construction is still very complicated, a brief sketch can be found in Section 3.1, while the full construction is omitted due to space constraints.

For the NE-case, we show that a comparable construction can be realized with NE-patterns. This observation is less obvious than it might appear and requires extensive modifications to the E-construction. As previous results on the non-decidability of the inclusion problem for NE-patterns rely on an involved construction from [10], we consider the construction used for our proof of Theorem 3.9 a significant technical breakthrough; especially as this result (together with its extension following from the modification in Section 5.1) allows us to solve Open Problem 1 in [7], concluding that the inclusion problem for NE-patterns over binary and ternary alphabets is undecidable. Some remarks on the construction are sketched in Section 3.2, while the full construction is omitted.

Although encoding the correct operation of a Turing machine (or any similar device) in patterns requires a considerable amount of variables, the simple structure of iterating the Collatz function $\mathcal{C}$ can be expressed in a more compact form. With far smaller bounds, we are able to obtain the following two results using the same constructions as for the proof of Theorems 3.8 and 3.9:

Theorem 3.10. *Let Σ be a binary alphabet. Every algorithm that decides the inclusion problem of* $\mathrm{ePAT}_{2,\Sigma}$ *in* $\mathrm{ePAT}_{74,\Sigma}$ *can be converted into an algorithm that, for every $N \in \mathbb{N}_1$, decides whether N leads $\mathcal{C}$ into the trivial cycle.*

Theorem 3.11. *Let Σ be a binary alphabet. Every algorithm that decides the inclusion problem of* $\mathrm{nePAT}_{2,\Sigma}$ *in* $\mathrm{nePAT}_{97,\Sigma}$ *can be converted into an algorithm that, for every $N \in \mathbb{N}_1$, decides whether N leads $\mathcal{C}$ into the trivial cycle.*

The proofs are sketched in Sections 3.1 and 3.2. As mentioned in Section 2.3, this demonstrates that, even for these far tighter bounds, the inclusion problems are able to express comparatively complicated sets. Moreover, a slight modification of the result allows us to state the following far stronger results:

Theorem 3.12. *Let Σ be a binary alphabet. Every algorithm that decides the inclusion problem for* $\mathrm{ePAT}_{4,\Sigma}$ *in* $\mathrm{ePAT}_{80,\Sigma}$ *can be used to decide whether any number $N \geq 1$ leads $\mathcal{C}$ into a non-trivial cycle.*

Theorem 3.13. *Let Σ be a binary alphabet. Every algorithm that decides the inclusion problem for* $\mathrm{nePAT}_{4,\Sigma}$ *in* $\mathrm{nePAT}_{102,\Sigma}$ *can be used to decide whether any number $N \geq 1$ leads $\mathcal{C}$ into a non-trivial cycle.*

The proofs are sketched in Sections 3.1 and 3.2. These two results need to be interpreted very carefully. Of course, the existence of non-trivial cycles is trivially decidable (by a constant predicate); but these results are stronger than mere decidability, as the patterns are constructed effectively. Thus, deciding the inclusion of any of the two pairs of patterns defined in the proofs would allow us to prove the existence of a counterexample to the Collatz Conjecture, or to rule out the existence of one important class of counterexamples, and thus solve 'one half' of the Collatz Conjecture. More pragmatically, we think that these results give reason to suspect that the inclusion problems of these classes of pattern languages are probably not solvable (even if effectively, then not efficiently), and definitely very complicated.

3.1. Sketch of the Construction for E-Patterns

As the construction is rather involved, we only give a basic sketch, and omit the full technical details. In each of the proofs, our goal is to decide the emptiness of a set V, which is one of $\mathrm{TRIV}_{\mathrm{E}}(N)$ (for some $N \geq 1$), $\mathrm{NTCC}_{\mathrm{E}}$, or $\mathrm{VALC}_{\mathrm{E}}(I)$ (for some configuration I). For this, we construct two patterns α and β such that $L_{\mathrm{E},\Sigma}(\alpha) \setminus L_{\mathrm{E},\Sigma}(\beta) \neq \emptyset$ if and only if $V \neq \emptyset$. The pattern α contains two subpatterns α_1 and α_2, where α_2 is a terminal-free pattern with $\mathrm{var}(\alpha_2) \subseteq \mathrm{var}(\alpha_1) \cup \{y\}$, and y is a variable that occurs exactly once in α_2, but does not occur in α_1.

Glossing over details (and ignoring the technical role of α_2), the main goal is to define β in such a way that, for every substitution σ, $\sigma(\alpha) \in L_{\mathrm{E},\Sigma}(\beta)$ if and only if $\sigma(\alpha_1) \in \mathrm{V}$. More explicitly, the subpattern α_1 generates a set of possible strings, and β encodes a disjunction of predicates on strings that describe the complement of V through all possible errors. If one of these errors occurs in $\sigma(\alpha_1)$, we can construct a substitution τ with $\tau(\beta) = \sigma(\alpha)$. If $\mathrm{V} = \emptyset$, every $\sigma(\alpha)$ belongs to $L_{\mathrm{E},\Sigma}(\beta)$. Otherwise, any element of V can be used to construct a word $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$. The proof of Theorem 3.2 in [7] can be interpreted as a special case of this construction, using $\alpha_1 := x$ and $\alpha_2 := y$. Through our modification, we are able to exert more control on the elements of $L_{\mathrm{E},\Sigma}(\alpha_1)$, and use this to define required repetitions, prefixes or suffixes for all $\sigma(\alpha_1)$ with $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$. The variables in $\mathrm{var}(\alpha_2) \setminus \{y\}$ are even further restricted, and can only be mapped to 0^*.

3.2. Sketch of the Construction for NE-Patterns

Describing the NE-construction on the same level of detail as the E-construction, both appear to be identical, including the presence and the role of subpatterns α_1 and α_2 in α. But as evidenced in the full proof, the peculiarities of NE-patterns require considerable additional technical effort. For example, the E-construction heavily depends on being able to map most variables in β to the empty word; dealing with these 'superfluous' variables is the largest difficulty for the modification. In order to overcome this problem, the pattern α contains long terminal-strings, which makes it possible to map every variable in β to at least one terminal. These terminal-strings complicate one of the main proofs, as we have to ensure that these terminal-strings do not prevent a necessary mapping, while not allowing any unintended mappings. The E-construction uses a set of variables x_i of which, under some preconditions, all but one have to be mapped to the empty word. That variable is then used to enforce certain decompositions of β in a way that allows us to encode the predicates in a system of word equations. In the NE-construction, we use a different prefix-construction to obtain a set of variables, which (again under some preconditions) all but one have to be mapped to the terminal 0, while the single remaining variable has to be mapped to the terminal #. Sometimes the NE-construction needs additional variables in contrast to the E-construction. Some minor changes make sure that the number of different variables in β does not increase too much – this is one reason for the different definitions of the encoding sets for the erasing and the nonerasing case in Sections 2.2 and 2.3. As we use more often terminals in the NE-construction instead of variables, the number of different variables can be even smaller than in the E-construction. Through this displacement the number of different variables in Theorem 3.9 is less than in Theorem 3.8. Furthermore the modifications of the construction and the use of nonerasing substitutions make the implementation of the extensions in Section 5 simpler than for the erasing case.

4. Proofs of the Main Theorems

4.1. The Construction for E-Patterns

In this section, we describe the construction that is common to the proofs of Theorems 3.8, 3.10 and 3.12, and describe how the number of necessary variables can be derived from each actual instantiation of the construction. The actual proofs for Theorems 3.8, 3.10 and 3.12 can be found in Section 4.2, 4.3 and 4.4, respectively.

Let $\Sigma = \{0, \#\}$. For each of the proofs, the goal is to decide the emptiness of a set V, which is one of $\mathrm{TRIV}_{\mathrm{E}}(N)$ (for some $N \geq 1$), $\mathrm{NTCC}_{\mathrm{E}}$, or $\mathrm{VALC}_{\mathrm{E}}(I)$ (for some configuration I). For this, we construct two patterns α and β such that $L_{\mathrm{E},\Sigma}(\alpha) \setminus L_{\mathrm{E},\Sigma}(\beta) = \emptyset$ if and only if $V \neq \emptyset$.

Basically, α generates a list of possible strings and provides some technical infrastructure, while β encodes a list of predicates π_1 to π_μ that describe all possible errors in the strings generated by α by describing the complement of

V. Due to the right choice of α and β, $L_{\mathrm{E},\Sigma}(\alpha) \subset L_{\mathrm{E},\Sigma}(\beta)$ holds if some word in $L_{\mathrm{E},\Sigma}(\alpha)$ satisfies none of the predicates.

Depending on the intended proof, we choose a structural parameter $\kappa \in \{2,3\}$ and a $\mu \geq 4$. The parameter κ has two purposes: First, it determines the maximal number of parameters in each predicate, and second, if none of the predicates is satisfied, the encoded word must not contain a factor $\#^\kappa$.

In addition to this, also depending on the actual proof, we select patterns α_1 and α_2, where α_1 is a pattern that does not contain $\#^\kappa$ as a factor, and α_2 is a terminal-free pattern with $\mathrm{var}(\alpha_2) \subseteq \mathrm{var}(\alpha_1) \cup \{y\}$, where y is a variable that occurs exactly once in α_2, but does not occur in α_1.

We define

$$\alpha{:=}v\; v\; \#^4\; v\; \alpha_1\; v\; \alpha_2\; v\; \#^4\; v\; u\; v,$$

where $v = 0\#^3 0$ and $u = 0\#\#0$. The pattern α_1 will be used to generate the set of possible members of V, while α_2 serves more technical purposes.

Note that the construction in [7] can be seen as a special case of the present construction, by selecting $\alpha_1{:=}x$, $\alpha_2{:=}y$ and $\kappa{:=}3$. Our more general approach allows us describe the intended starting and ending values of the encoded computation in α_1 without the use of additional predicates. Furthermore, as we shall see soon, the variables in $\mathrm{var}(\alpha_1) \cap \mathrm{var}(\alpha_2)$ provide us with greater control on the shape of the images of α_1.

Furthermore, let

$$\beta{:=}(x_1)^2 \dots (x_\mu)^2 \#^4 \hat{\beta}_1 \dots \hat{\beta}_\mu \#^4 \ddot{\beta}_1 \dots \ddot{\beta}_\mu,$$

with, for all $i \in \{1, \dots, \mu\}$, $\hat{\beta}_i{:=}x_i\; \gamma_i\; x_i\; \delta_i\; x_i$ and $\ddot{\beta}_i{:=}x_i\; \eta_i\; x_i$, where $x_1, \dots, x_\mu$ are pairwise distinct variables and all $\gamma_i, \delta_i, \eta_i \in X^*$ are terminal-free patterns. The patterns γ_i and δ_i shall be defined later; for now, we only mention:

1. $\eta_i{:=}z_i(\hat{z}_i)^2 z_i$ and $z_i \neq \hat{z}_i$ for all $i \in \{1, \dots, \mu\}$,
2. $\mathrm{var}(\gamma_i\delta_i\eta_i) \cap \mathrm{var}(\gamma_j\delta_j\eta_j) = \emptyset$ for all $i, j \in \{1, \dots, \mu\}$ with $i \neq j$,
3. $x_k \notin \mathrm{var}(\gamma_i\delta_i\eta_i)$ for all $i, k \in \{1, \dots, \mu\}$.

Thus, for every i, the elements of $\mathrm{var}(\gamma_i\delta_i\eta_i)$ appear nowhere but in these three factors. Let H be the set of all substitutions $\sigma : (\Sigma \cup \mathrm{var}(\alpha_1\alpha_2))^* \to \Sigma^*$. We interpret each triple $(\gamma_i, \delta_i, \eta_i)$ as a predicate $\pi_i : H \to \{0,1\}$ in such a way that $\sigma \in H$ *satisfies* π_i if there exists a morphism $\tau : \mathrm{var}(\gamma_i\delta_i\eta_i)^* \to \Sigma^*$ with $\tau(\gamma_i) = \sigma(\alpha_1)$, $\tau(\delta_i) = \sigma(\alpha_2)$ and $\tau(\eta_i) = u$. As we shall see, $L_{\mathrm{E},\Sigma}(\alpha) \setminus L_{\mathrm{E},\Sigma}(\beta)$ exactly contains those $\sigma(\alpha)$ for which σ does not satisfy any of π_1 to π_μ. Our goal is a selection of predicates that describe the complement of V, where the predicates π_4 to π_μ provide an exhaustive list of sufficient criteria for 'non-membership' in V. We continue with further technical preparations.

A substitution σ is of *κ-E-bad form* if $\sigma(\alpha_1)$ contains $\#^\kappa$ as a factor, or if $\sigma(\alpha_2)$ contains $\#$. Otherwise, σ is of *κ-E-good form*. For $\kappa = 3$, this notion is equivalent to the concept of bad form and good form in [7].

The predicates π_1 and π_2 describe the cases where σ is of κ-E-bad form and are defined by

$$\gamma_1 := y_{1,1}(\hat{z}_1)^\kappa y_{1,2}, \qquad \gamma_2 := y_2,$$
$$\delta_1 := \hat{y}_1, \qquad \delta_2 := \hat{y}_{2,1}\ \hat{z}_2\ \hat{y}_{2,2},$$

where $y_{1,1}$, $y_{1,2}$, y_2, $\hat{y}_1$, $\hat{y}_{2,1}$, $\hat{y}_{2,2}$, $\hat{z}_1$ and $\hat{z}_2$ are pairwise distinct variables.

Recall that $\eta_i = z_i(\hat{z}_i)^2 z_i$ for all i. It is not very difficult to see that π_1 and π_2 characterize the morphisms that are of κ-E-bad form:

Lemma 4.1. *A substitution $\sigma \in H$ is of κ-E-bad form if and only if σ satisfies π_1 or π_2.*

Proof. Apart from the changed definition of α_1 and α_2, this proof is identical to the proof of Lemma 1 in [7].

We begin with the *only if* direction. If $\sigma(\alpha_1) = w_1\#^\kappa w_2$ for some $w_1, w_2 \in \Sigma^*$, choose $\tau(y_{1,1}) := w_1$, $\tau(y_{1,2}) := w_2$, $\tau(\hat{z}_1) := \#$, $\tau(\hat{y}_1) := \sigma(\alpha_2)$ and $\tau(z_1) := 0$. Then $\tau(\gamma_1) = \sigma(\alpha_1)$, $\tau(\delta_1) = \sigma(\alpha_2)$ and $\tau(\eta_1) = u$; thus, σ satisfies π_1.

If $\sigma(\alpha_2) = w_1\#w_2$ for some $w_1, w_2 \in \Sigma^*$, let $\tau(y_2) := \sigma(\alpha_1)$, $\tau(\hat{y}_{2,1}) := w_1$, $\tau(\hat{y}_{2,2}) := w_2$ and $\tau(\hat{z}_2) := \#$, and $\tau(z_2) := 0$. It is easy to see that σ satisfies π_2.

For the *if* direction, if σ satisfies π_1, then there exists a morphism τ with $\tau(\gamma_1) = \sigma(\alpha_1)$ and $\tau(\eta_1) = 0\#^2 0$. Thus, $\tau(\hat{z}_1) = \#$ and $\tau(z_1) = 0$ must hold. Then, by definition of γ_1, $\sigma(\alpha_1) = \tau(y_{1,1})\#^\kappa\tau(y_{1,2})$, which means that σ is of κ-E-bad form.

Analogously, if σ satisfies π_2, then $\sigma(\alpha_2)$ contains the letter $\#$, and σ is of κ-E-bad form. □

Note that, if σ is of κ-E-good form, $\sigma(x) \in 0^*$ for all variables $x \in \mathrm{var}(\alpha_1) \cap \mathrm{var}(\alpha_2)$. Thus, these variables provide us with greater control on the shape of $\sigma(\alpha_1)$ for the remaining predicates.

As in the original, Lemma 4.1 leads us to the central part of the construction:

Lemma 4.2. *For every substitution $\sigma \in H$, $\sigma(\alpha) \in L_{\mathrm{E},\Sigma}(\beta)$ if and only if σ satisfies one of the predicates π_1 to π_μ.*

Proof. This proof is also almost identical to the proof of Lemma 2 in [7]. We begin with the *if* direction. Assume $\sigma \in H$ satisfies some predicate π_i. Then there exists a morphism $\tau : (\mathrm{var}(\gamma_i\delta_i\eta_i))^* \to \Sigma^*$ such that $\tau(\gamma_i) = \sigma(\alpha_1)$, $\tau(\delta_i) = \sigma(\alpha_2)$ and $\tau(\eta_i) = u$. We extend τ to a substitution τ' defined by

1. $\tau'(x) := \tau(x)$ for all $x \in \mathrm{var}(\gamma_i\delta_i\eta_i)$,
2. $\tau'(x_i) := 0\#^3 0 = v$,
3. $\tau'(0) := 0$ and $\tau'(\#) := \#$,
4. $\tau'(x) := \lambda$ in all other cases.

By definition, none of the variables in $\mathrm{var}(\gamma_i\delta_i\eta_i)$ appears outside of these factors. Thus, τ' can always be defined this way. We obtain

$$\begin{aligned}
\tau'(\hat\beta_i) &= \tau'(x_i\,\gamma_i\,x_i\,\delta_i\,x_i)\\
&= v\,\tau(\gamma_i)\,v\,\tau(\delta_i)\,v\\
&= v\,\sigma(\alpha_1)\,v\,\sigma(\alpha_2)\,v,\\
\tau'(\ddot\beta_i) &= \tau'(x_i\,\eta_i\,x_i)\\
&= v\,\tau(\eta)\,v\\
&= v\,u\,v.
\end{aligned}$$

As $\tau'(\gamma_j) = \tau'(\delta_j) = \tau'(\eta_j) = \tau'(\hat\beta_j) = \tau'(\ddot\beta_j) = \lambda$ for all $j \neq i$, this leads to

$$\begin{aligned}
\tau'(\beta) &= \tau'\left((x_1)^2\ldots(x_\mu)^2\#^4\hat\beta_1\ldots\hat\beta_\mu\#^4\ddot\beta_1\ldots\ddot\beta_\mu\right)\\
&= \tau'\left((x_i)^2\right)\#^4\tau'(\hat\beta_i)\#^4\tau'(\ddot\beta_i)\\
&= v\,v\,\#^4\,v\,\sigma(\alpha_1)\,v\,\sigma(\alpha_2)\,v\,\#^4\,v\,u\,v\\
&= \sigma(\alpha).
\end{aligned}$$

This proves $\sigma(\alpha) \in L_{\mathrm{E},\Sigma}(\beta)$.

For the other direction, assume $\sigma(\alpha) \in L_{\mathrm{E},\Sigma}(\beta)$. If σ is of κ-E-bad form, then by Lemma 4.1, σ satisfies π_1 or π_2. Thus, assume $\sigma(\alpha_1)$ does not contain $\#^\kappa$ as a factor, and $\sigma(\alpha_2) \in 0^*$. Let τ be a substitution with $\tau(\beta) = \sigma(\alpha)$.

Now, as σ is of κ-E-good form, $\sigma(\alpha)$ contains exactly two occurrences of $\#^4$, and these are non-overlapping. As $\sigma(\alpha) = \tau(\beta)$, the same holds for $\tau(\beta)$. Thus, the equation $\sigma(\alpha) = \tau(\beta)$ can be decomposed into the system consisting of the following three equations:

$$0\#^3 0\,0\#^3 0 = \tau\left((x_1)^2\ldots(x_\mu)^2\right), \tag{1}$$

$$0\#^3 0\,\sigma(\alpha_1)\,0\#^3 0\,\sigma(\alpha_2)\,0\#^3 0 = \tau(\hat\beta_1\ldots\hat\beta_\mu), \tag{2}$$

$$0\#^3 0\,u\,0\#^3 0 = \tau(\ddot\beta_1\ldots\ddot\beta_\mu). \tag{3}$$

First, consider equation (1) and choose the smallest i for which $\tau(x_i) \neq \lambda$. Then $\tau(x_i)$ has to start with 0, and as

$$\tau\left((x_i)^2\ldots(x_\mu)^2\right) = 0\#^3 0\,0\#^3 0,$$

it is easy to see that $\tau(x_i) = 0\#^3 0 = v$ and $\tau(x_j) = \lambda$ for all $j \neq i$ must hold.

Note that u does not contain $0\#^3 0$ as a factor, and does neither begin with $\#^3 0$, nor end on $0\#^3$. But as $\tau(\ddot\beta_i)$ begins with and ends on $0\#^3 0$, we can use equation (3) to obtain $0\#^3 0\,u\,0\#^3 0 = \tau(\ddot\beta_i)$ and $\tau(\ddot\beta_j) = \lambda$ for all $j \neq i$. As $\ddot\beta_i = x_i\eta_i x_i$ and $\tau(x_i) = 0\#^3 0$, $\tau(\eta_i) = u$ must hold.

As σ is of κ-E-good form, $\sigma(0\#^3 0\,\alpha_1\,0\#^3 0\,\alpha_2\,0\#^3 0)$ contains exactly three occurrences of $\#^3$. But there are already three occurrences of $\#^3$ in $\tau(\hat\beta_i) = 0\#^3 0\,\tau(\gamma_i)\,0\#^3 0\,\tau(\delta_i)\,0\#^3 0$. This, and equation (2), lead to $\tau(\hat\beta_j) = \lambda$ for all $j \neq i$ and, more importantly, $\tau(\gamma_i) = \sigma(\alpha_1)$ and $\tau(\delta_i) = \sigma(\alpha_2)$. Therefore, σ satisfies the predicate π_i. $\square$

Thus, we can select predicates π_1 to π_μ in such a way that $L_{\mathrm{E},\Sigma}(\alpha) \setminus L_{\mathrm{E},\Sigma}(\beta) = \emptyset$ if and only if $\mathrm{V} = \emptyset$ by describing $\overline{\mathrm{V}}$ through a disjunction of predicates on H. The proof of Lemma 4.2 shows that if $\sigma(\alpha) = \tau(\beta)$ for substitutions σ and τ; where σ is of κ-E-good form, there exists exactly one i ($3 \leq i \leq \mu$) such that $\tau(x_i) = 0\#^3 0$.

Due to technical reasons, we need a predicate π_3 that, if unsatisfied, sets a lower bound to the length of $\sigma(\alpha_2)$, defined by

$$\gamma_3 := y_{3,1}\ \hat{y}_{3,1}\ y_{3,2}\ \hat{y}_{3,2}\ y_{3,3}, \qquad \delta_3 := \hat{y}_{3,1}\ \hat{y}_{3,2},$$

if $\kappa = 2$, or by

$$\gamma_3 := y_{3,1}\ \hat{y}_{3,1}\ y_{3,2}\ \hat{y}_{3,2}\ y_{3,3}\ \hat{y}_{3,3}\ y_{3,4}, \qquad \delta_3 := \hat{y}_{3,1}\ \hat{y}_{3,2}\ \hat{y}_{3,3},$$

if $\kappa = 3$; where in either case all of $y_{3,1}$ to $y_{3,4}$ and $\hat{y}_{3,1}$ to $\hat{y}_{3,3}$ are pairwise distinct variables.

Clearly, if some $\sigma \in H$ satisfies π_3, $\sigma(\alpha_2)$ is a concatenation of κ (possibly empty) factors of $\sigma(\alpha_1)$. Thus, if σ satisfies none of π_1 to π_3, $\sigma(\alpha_2)$ has to be longer than the κ longest non-overlapping sequences of 0s in $\sigma(\alpha_1)$. This allows us to identify a class of predicates definable by a rather simple kind of expression, which we use to define π_4 to π_μ in a less technical way. Note that any meaningful use of this construction requires α_2 to contain at least one variable that does not occur in α_1, as otherwise, π_3 would always be satisfied.

Let $X_\kappa := \{\hat{x}_1, \ldots, \hat{x}_\kappa\} \subset X$, let G_κ denote the set of those substitutions in H that are of κ-E-good form and let R be the set of all substitutions $\rho : (\Sigma \cup X_\kappa)^* \to \Sigma^*$ for which $\rho(\hat{x}_i) \in 0^*$ for all i with $1 \leq i \leq \kappa$. For patterns $\zeta \in (\Sigma \cup X_\kappa)^*$, we define $R(\zeta) := \{\rho(\zeta) \mid \rho \in R\}$.

Definition 1. A predicate $\pi : G_\kappa \to \{0, 1\}$ is called a *κ-simple predicate for* α_1 if there exist a pattern $\zeta \in (\Sigma \cup X_\kappa)^*$ and languages $L_1, L_2 \in \{\Sigma^*, \{\lambda\}\}$ such that a substitution σ satisfies π if and only if $\sigma(\alpha_1) \in L_1\ R(\zeta)\ L_2$. If $L_1 = L_2 = \Sigma^*$, we call π an *infix-predicate.* If only $L_1 = \Sigma^*$ and $L_2 = \{\lambda\}$, π is called a *suffix-predicate*, and if $L_1 = \{\lambda\}$ and $L_2 = \Sigma^*$, a *prefix-predicate.*

¿From a slightly different point of view, the elements of X_κ can be understood as numerical parameters describing (concatenational) powers of 0, with substitutions $\rho \in R$ acting as assignments. For example, if $\sigma \in G_\kappa$ satisfies a κ-simple predicate π if and only if $\sigma(\alpha_1) \in \Sigma^* R(\#\hat{x}_1\#\hat{x}_2 0\#\hat{x}_1)$, we can also write that σ satisfies π if and only if $\sigma(\alpha_1)$ has a suffix of the form $\#0^m\#0^n 0\#0^m$ (with $m, n \in \mathbb{N}_0$), which could also be written as $\#0^m\#0^* 0\#0^m$, as n occurs only once in this expression. Although these predicates do not explicitly allow arithmetical operations on the numerical parameters, we use expressions like 0^{m+2n+1} as a shorthand for $0^m 0^n 0^n 0$.

As in the original construction, the predicate π_3 allows us to express all κ-simple predicates:

Lemma 4.3. *For every κ-simple predicate π_S having n numerical parameters with $n \leq \kappa$, there exists a predicate π defined by terminal-free patterns γ, δ, η such that for all substitutions $\sigma \in G_\kappa$:*

1. *if σ satisfies π_S, then σ also satisfies π or π_3,*
2. *if σ satisfies π, then σ also satisfies π_S.*

Proof. This proof is a variation of the proof of Lemma 3 in [7].

We first consider the case of $L_1 = L_2 = \Sigma^*$. Assume π_S is a κ-simple predicate, and $\zeta \in (\Sigma \cup X_\kappa)^*$ is a pattern such that $\sigma \in G_\kappa$ satisfies π_S if and only if $\sigma(\alpha_1) \in L_1 R(\zeta) L_2$. Then define $\gamma := y_1\, \zeta'\, y_2$, where ζ' is obtained from ζ by replacing all occurrences of 0 with a new variable z and all occurrences of # with a different variable $\hat{z}$, while leaving all present elements of X_κ unchanged. Furthermore, $\delta := \hat{x}_1 \ldots \hat{x}_\kappa \hat{y}$. Finally, in order to stay consistent with the η_i appearing in β, let $\eta := z(\hat{z})^2 z$. Note that $\hat{x}_1$, $\hat{x}_2$, $\hat{x}_3$, y_1, y_2, z and $\hat{z}$ are pairwise distinct variables.

Now, assume $\sigma \in G_\kappa$ satisfies π_S. Then there exist words $w_1, w_2 \in \Sigma^*$ and a substitution $\rho \in R$ such that $\sigma(\alpha_1) = w_1\, \rho(\zeta)\, w_2$. If $\sigma(\alpha_2)$ is not longer than any κ non-overlapping factors of the form 0^* of $\sigma(\alpha_1)$ combined, π_3 is satisfied. Otherwise, we can define τ by setting $\tau(y_1) := w_1$, $\tau(y_2) := w_2$, $\tau(z) := 0$, $\tau(\hat{z}) := \#$, $\tau(\hat{x}_i) := \rho(\hat{x}_i)$ for all $i \in \{1, \ldots, k\}$ where $\hat{x}_i$ appears in ζ and $\tau(\hat{x}_i) := \lambda$ where $\hat{x}_i$ does not appear in ζ. Finally, let $\tau(\hat{y}) := 0^m$, where

$$m := |\sigma(\alpha_2)| - \sum_{\hat{x} \in \operatorname{var}(\zeta)} |\tau(\hat{x})|$$

($m > 0$ holds, as σ does not satisfy π_3). Then $\tau(\zeta') = \rho(\zeta)$, and

$$\begin{aligned}
\tau(\gamma) &= \tau(y_1)\, \tau(\zeta')\, \tau(y_2) \\
&= w_1\, \rho(\zeta)\, w_2 = \sigma(\alpha_1), \\
\tau(\delta) &= 0^{|\sigma(\alpha_2)|} = \sigma(\alpha_2), \\
\tau(\eta) &= \tau(z\, (\hat{z})^2\, z) \\
&= 0\#\#0 = u.
\end{aligned}$$

Therefore, σ satisfies π, which concludes this direction.

For the other direction, assume $\sigma \in G_\kappa$ satisfies π. Then there is a morphism τ such that $\sigma(\alpha_1) = \tau(\gamma)$, $\sigma(\alpha_2) = \tau(\delta)$ and $\tau(\eta) = u$. As $\eta = z\, (\hat{z})^2\, z$ and $u = 0\#\#0$, $\tau(z) = 0$ and $\tau(\hat{z}) = \#$ must hold. By definition $\tau(y_1), \tau(y_2) \in \Sigma^*$. If we define $\rho(\hat{x}_i) := \tau(\hat{x}_i)$ for all $\hat{x}_i \in \operatorname{var}(\delta)$, we see that $\sigma(\alpha_1) \in L_1 R(\zeta) L_2$ holds. Thus, σ satisfies π_S as well.

The other three cases for choices of L_1 and L_2 can be handled analogously by omitting y_1 or y_2 as needed. Note that this proof also works in the case $\zeta = \lambda$. □

Intuitively, if σ does not satisfy π_3, then $\sigma(\alpha_2)$ (which is in 0^*, due to $\sigma \in G_\kappa$) is long enough to provide building blocks for κ-simple predicates using variables from X_κ.

All that remains for each of the proofs is to choose an appropriate set of predicates.

Then it is easy to see how many variables each predicate in β requires. First, every predicate π_i has a corresponding variable x_i, for μ variables in total. The predicates π_1 and π_2 each use five further variables, π_3 uses $2\kappa + 3$ additional variables. In total, β contains $\mu + 2\kappa + 13$ variables for the predicates π_1 to π_3 and the variables x_i, and the additional variables that are required to encode the remaining predicates π_4 to π_μ.

Each of these predicates requires:

1. three variables for y_i, z_i and $\hat{z}_i$,
2. one variable for each numerical parameter (or star),
3. one additional variable if it is a prefix or a suffix predicate,
4. two additional variables if it is an infix predicate.

Thus, each predicate requires at least 3 and at most 8 variables.

4.2. Proof of Theorem 3.8.

Proof. For both claims of the proof, we show that, given any configuration I of U, we can construct patterns α and β from the appropriate classes such that $L_{\mathrm{E},\Sigma}(\alpha) \setminus L_{\mathrm{E},\Sigma}(\beta) = \emptyset$ if and only if $\mathrm{VALC}_{\mathrm{E}}(I) = \emptyset$. The predicates for the proofs of the two claims of this theorem are very similar, they differ only at the choice of α_1 and α_2, and an additional predicate that is required for the second case. For the first claim, we chose $\mu = 333$, for the second, $\mu = 334$. In either case, we choose $\kappa = 3$.

For the first claim of the theorem, we choose

$$\alpha_1 := \#\#\,\mathrm{enc}_{\mathrm{E}}(I)\#\#x_1\#00x_2x_2\#0^{10}\#\#, \qquad\qquad \alpha_2 := x_2 y,$$

where x_1, x_2 and y are pairwise distinct variables; for the second,

$$\alpha_1 := \#\#\,\mathrm{enc}_{\mathrm{E}}(I)\#\#x\#0^{10}\#\#, \qquad\qquad \alpha_2 := y,$$

where x and y are distinct variables. Ultimately, if $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$, $\sigma(\alpha_1)$ is supposed to contain an encoding of a valid computation that starts in the configuration I, and leads to an accepting configuration. The variable x_2 in the subpattern α_1 of the first claim will have an image from 0^*, which means that the left tape of the final configuration has an odd encoding, and thus contains 1, while the machine is in state q_{10}. For the second claim, this condition will be checked by an additional predicate, which requires 6 additional variables in β.

Our first intermediate goal is a set of predicates that (if unsatisfied) forces $\sigma(\alpha_1)$ into a basic shape common to all elements of $\mathrm{VALC}_{\mathrm{E}}(I)$. In other words, we want to remove all cases where

$$\sigma(\alpha_1) \notin (\#\#0^+\#0^+\#0^+)^+\#\#,$$

or $\sigma(\alpha_1)$ contains a factor $0^{16}\#\#$ and thus, an encoding of a state q_n with $n > 15$ (such a state does not exist in U).

To achieve this goal, we define predicates π_4 to π_7 by κ-simple predicates as follows:

$$\pi_4 : \sigma(\alpha_1) \text{ contains a factor } \#\#0^+\#\#,$$
$$\pi_5 : \sigma(\alpha_1) \text{ contains a factor } \#\#0^+\#0^+\#\#,$$
$$\pi_6 : \sigma(\alpha_1) \text{ contains a factor } \#\#0^+\#0^+\#0^+\#0,$$
$$\pi_7 : \sigma(\alpha_1) \text{ contains a factor } 0^{16}\#\#.$$

Due to Lemma 4.3, the predicates π_1 to π_7 do not strictly give rise to a characterization of substitutions with images that are not an encoding of a sequence of configurations of U, as there are $\sigma \in G_\kappa$ where $\sigma(\alpha_1)$ is of the right shape, but π_3 is satisfied due to $\sigma(\alpha_2)$ being too short. But this problem can be avoided by choosing $\sigma(\alpha_2)$ long enough to leave π_3 unsatisfied.

Thus, if σ satisfies none of the predicates π_1 to π_7, $\sigma(\alpha_1)$ is an encoding of a sequence of configurations of U that starts with I, and ends in a halting configuration (for the first claim we prove), or a configuration in state q_{10} (for the second claim).

The remaining predicates will describe all errors where one of the encoded configurations is not a valid successor of its preceding configuration[3]. We will first consider all errors in state transitions, and then all errors in the tape contents.

In principle, we could now define predicates that, for every state $q_i \in Q$, every input letter $a \in \Gamma$, list all states that are not the successor state of q_i on input a. In order to save predicates (and thereby variables), our approach is a little bit more involved. Every state has at most two legal successor states, and the states q_6, q_{10} and q_{15} have only one successor. Thus, we can first exclude forbidden successor states regardless of the input letter, and then handle the few remaining cases. Furthermore, we are able to express the fact that a successor state has a larger number than possible.

In order to determine a good choice of predicates, it helps to visualize the relations of possible predecessor and successor states in a matrix. We define the 15×15 matrix $S = (s_{i,j})_{i,j=1}^{15}$ by

$$s_{i,j} := \begin{cases} 1 & \text{if there is an } a \in \Gamma \text{ with } \delta(q_i, a) = q_j, \\ 0 & \text{otherwise.} \end{cases}$$

For a graphical representation of S and the predicate that are derived from it, see Figure 3. Intuitively, $s_{i,j}$ equals 0 if and only if q_j can never be a valid immediate successor of q_j, regardless of the input letter.

[3]Note that, at this point, the construction uses 5 infix predicates (in addition to π_1 to π_3); one for each possible number of numerical parameters from 0 to 3. Even this small number of predicates requires 52 variables in β, and is only able to express the basic shape of encoded configurations.

$$S = \begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}$$

Figure 3: The matrix S, listing the possible (and impossible) immediate successors and predecessors of the states. Lines denote the sets of impossible state pairs that are described by the predicates π_8 to π_{34}. The remaining occurrences of 0 are handled by the predicates π_{35} to π_{66}.

First, we construct a predicate

$$\pi_8 : \sigma(\alpha_1) \text{ contains a factor } \#0^1\#\#0^+\#0^+\#0^3.$$

This predicates handles all cases where the encoding contains a configuration with state q_1, where the next state is some q_j with $j \geq 3$. In the same spirit, we can define a predicate that handles all configurations where q_1 is preceded by a state q_j with $j \geq 10$, which is also impossible in a valid computation:

$$\pi_9 : \sigma(\alpha_1) \text{ contains a factor } 0^{10}\#\#0^+\#0^+\#0^1\#.$$

Intuitively, π_8 describes all occurrences of 0 in the first row of S, while π_9 describes the bottom block of 6 occurrences of 0 in the first column.

We define similar predicates π_{10} to π_{33} for all states q_2 to q_{13}; each predicate handles the longest continuous block of 0s when reading a row from the right, or a column from the bottom.

Using the matrix S it is easy to see that this is not possible for q_{14}, as this state has q_{15} as successor and as predecessor. Similarly, the state q_{15} is handled by a single predicate

$$\pi_{34} : \sigma(\alpha_1) \text{ contains a factor } \#0^{15}\#\#0^+\#0^+\#0^{15}\#$$

that describes the lone 0 in the bottom right corner of S. Each of the 27 predicates π_8 to π_{34} is an infix predicate with 2 numerical parameters.

It seems like reordering the states could transform the matrix and reduce the number of predicates for single occurrences of 0. But after some experimentation, we decided that the expected small savings would not warrant the considerable effort. Further (but still comparatively small) savings might be achieved by the use of a machine with a different matrix.

There are still 32 occurrences of 0 that have at least one 1 between them an the right side or the bottom of S. Thus, for each $s_{i,j}$ with this property, we define a predicate

$$\pi_k : \sigma(\alpha_1) \text{ contains } \#0^i\#\#0^+\#0^+\#0^j\#$$

for an appropriate k. This leads to the 32 predicates π_{35} to π_{66}, also infix predicates with 2 numerical parameters.

Now, only 24 possible errors need to be considered. For every state $q_i \in Q \setminus \{q_6, q_{10}, q_{15}\}$, and every input letter $a \in \Gamma$, we need to describe the error that the succeeding state is the one possible successor state that would have been reached from q_i by reading the complement of a. This leads to the predicates π_{67} to π_{90}; as an example, we define the two predicates that handle the invalid successor states of q_1:

$$\pi_{67} : \sigma(\alpha_1) \text{ contains } \#00^{2m}\#0^1\#\#0^+\#0^+\#0^1\#; \qquad m \in \mathbb{N}_0,$$
$$\pi_{68} : \sigma(\alpha_1) \text{ contains } \#00^{2m+1}\#0^1\#\#0^+\#0^+\#0^2\#; \qquad m \in \mathbb{N}_0.$$

The first of these two predicates describes all cases where the machine is in the state q_1, reads 0 (as $\mathrm{enc}(t_L) \bmod 2 = 0 = \mathrm{e}(0)$) and stays in the state q_1, while π_{68} describes all cases where the machine transitions to q_2 upon reading 1 in state q_1.

No such predicates are required for the states q_6 and q_{15}, as these have only one possible successor state. As we permitted the machine to continue working after reaching a halting computation, the same applies to q_{10}. The 24 predicates π_{67} to π_{90} are infix predicates with three numerical parameters (as the starts count as numerical parameters that occur only once).

Thus, if σ satisfies none of the predicates π_1 to π_{90}, $\sigma(\alpha_1)$ encodes a sequence of configurations that starts with the initial configuration I and ends on the state q_{10} (as mentioned before, we also know that in the proof of the first claim, the final configuration is an accepting configuration, but this fact will be discussed later). Furthermore, we know that all transitions of the states are correct. Therefore, all that remains is to define a set of predicates that handle errors in the handling of the tape.

For this, we need to distinguish between left movements and right movements. Before we proceed to the definition of the predicates for tape error in each of these cases, we take a closer look at the intended behavior of valid computations, and their encodings in $\mathrm{VALC}_{\mathrm{E}}(I)$. Assume U is in some state q_i, while the tape contains t_L on the left and t_R on the right side. Let a denote the input letter, i. e., $\mathrm{e}(a) = (\mathrm{e}(t_L) \bmod 2)$. Let t'_L and t'_R denote the left and the right tape side of the succeeding valid configuration, respectively.

First, consider the case that $\delta(q_i, a) = (d, L, q_j)$ for some state $q_j \in Q$ and an output letter $d \in \Gamma$. In this case,

$$\begin{aligned}\mathrm{e}(t'_L) &= \mathrm{e}(t_L) \operatorname{div} 2,\\ \mathrm{e}(t'_R) &= 2(\mathrm{e}(t_R)) + \mathrm{e}(d).\end{aligned}$$

Thus, every tape error can be understood as a difference between the supposed e-value of the encoded side, and the actual e-value. As we shall see, all these differences can be described by a finite number of simple predicates, simulating arithmetic operations with the numerical parameters.

We begin with predicates for values that are too large, which can be defined more straightforwardly than for too small values. For some appropriate $k > 90$, define the predicates

$$\begin{aligned}\pi_k &: \sigma(\alpha_1) \text{ contains } \#00^{2m+\mathrm{e}(a)}\#0^i\#\#0^+\#00^{m+1}; && m \in \mathbb{N}_0,\\ \pi_{k+1} &: \sigma(\alpha_1) \text{ contains } \#00^m\#00^{2n+\mathrm{e}(a)}\#0^i\#\#00^{2m+\mathrm{e}(d)+1}; && m, n \in \mathbb{N}_0.\end{aligned}$$

These capture all cases where, upon reading a in state q_i, the left or the right side of the tape (respectively) in the succeeding configuration contains more than it is supposed to (more meaning that its image under e is larger).

The following predicate describes all cases where the encoding of the left side of tape is too small:

$$\pi_{k+2} : \sigma(\alpha_1) \text{ contains } \#00^{2(m+n+1)+\mathrm{e}(a)}\#0^i\#\#0^+\#00^m\#; \quad m, n \in \mathbb{N}_0.$$

We capture the same case for the right side of the tape by the following two cases:

$$\begin{aligned}\pi_{k+3} &: \sigma(\alpha_1) \text{ contains } \#00^{2m+\mathrm{e}(a)}\#0^i\#\#00^{2n+(1-\mathrm{e}(d))}\#; && m, n \in \mathbb{N}_0,\\ \pi_{k+4} &: \sigma(\alpha_1) \text{ contains } \#00^{l+m+1}\#00^{2n+\mathrm{e}(a)}\#0^i\#\#00^{2m+\mathrm{e}(d)}\#; && l, m, n \in \mathbb{N}_0.\end{aligned}$$

As $\mathrm{e}(t'_R) = 2(\mathrm{e}(t_R)) + \mathrm{e}(d)$ holds, we know that every case with $\mathrm{e}(t'_R) \bmod 2 \neq \mathrm{e}(d)$ contains an error, which is described by π_{k+3}. Assuming that this predicate is not satisfied, we can use π_{k+4} to capture all cases where $\mathrm{e}(t'_R) \bmod 2$ equals $\mathrm{e}(d) \bmod 2$, but is too small.

This concludes the definitions of tape error for L movements. Every combination of q_i and a that results in an L-movement requires 5 infix predicates π_k to π_{k+4}; the first two use 2 parameters, the other three use 3 parameters. In total, U has 15 combinations (q_i, a) that lead to an L-movement. Therefore, we need 75 predicates for tape errors of L-movements, which brings us to an intermediate total of 165 predicates.

Next, assume $\delta(q_i, a) = (d, R, q_j)$ for some state $q_j \in Q$ and an output letter $d \in \Gamma$. Then

$$\begin{aligned}\mathrm{e}(t'_L) &= 2(2(\mathrm{e}(t_L) \operatorname{div} 2) + \mathrm{e}(d)) + (\mathrm{e}(t_R) \bmod 2)\\ &= 4(\mathrm{e}(t_L) \operatorname{div} 2) + 2\,\mathrm{e}(d) + (\mathrm{e}(t_R) \bmod 2),\\ \mathrm{e}(t'_R) &= \mathrm{e}(t_R) \operatorname{div} 2.\end{aligned}$$

Although the second of these equations should be clear, the first is comparatively involved and is best understood by examining it from the inside. The intermediate result $2(\mathrm{e}(t_L) \operatorname{div} 2) + \mathrm{e}(d)$ sets the tape cell under the head to the letter d, multiplying this number by 2 shifts the whole left side of the tape one cell to the left and appends a new cell containing the blank symbol 0. This symbol is then overwritten with the first letter of the right side of the tape by adding $(\mathrm{e}(t_R) \bmod 2)$. Thus, $\mathrm{e}(t'_L)$ is indeed an encoding of the left side of the tape after the step (d, R, q_j).

For fixed q_i and a, encoding R-steps is more involved than encoding L-steps, as we need to distinguish the two possible cases for $t_R \bmod 2$. This is the reason we chose to count the head of U to the left side of the tape, as we have only 14 R-movements, but 15 L-movements. Larger savings could be achieved by using a different machine with a larger difference in the number of L- and R-movements; but as mentioned before, we do not think that these slight improvements warrant the effort.

For an appropriate $k > 165$, we define the following four predicates for cases where one of the sides of the tapes contains too much:

$$\begin{aligned}
&\pi_k : \sigma(\alpha_1) \text{ contains } \#00^{2m}\#00^{2n+\mathrm{e}(a)}\#0^i\#\#0^+\#00^{2(2n+\mathrm{e}(d))+1}; && m, n \in \mathbb{N}_0,\\
&\pi_{k+1} : \sigma(\alpha_1) \text{ contains } \#00^{2m+1}\#00^{2n+\mathrm{e}(a)}\#0^i\#\#0^+\#00^{2(2n+\mathrm{e}(d))+2}; && m, n \in \mathbb{N}_0,\\
&\pi_{k+2} : \sigma(\alpha_1) \text{ contains } \#00^{2m}\#00^{2n+\mathrm{e}(a)}\#0^i\#\#00^{m+1}; && m, n \in \mathbb{N}_0,\\
&\pi_{k+3} : \sigma(\alpha_1) \text{ contains } \#00^{2m+1}\#00^{2n+\mathrm{e}(a)}\#0^i\#\#00^{m+1}; && m, n \in \mathbb{N}_0.
\end{aligned}$$

The first two describe the cases where t'_L is too large (with $\mathrm{e}(t_R)$ being even or odd, respectively), the second two the cases where $\mathrm{e}(t'_R)$ is too large.

Next, we define two predicates that are satisfied if t'_R is too small:

$$\begin{aligned}
&\pi_{k+4} : \sigma(\alpha_1) \text{ contains } \#00^{2(l+m+1)}\#00^{2n+\mathrm{e}(a)}\#0^i\#\#00^l\#; && l, m, n \in \mathbb{N}_0,\\
&\pi_{k+5} : \sigma(\alpha_1) \text{ contains } \#00^{2(l+m+1)+1}\#00^{2n+\mathrm{e}(a)}\#0^i\#\#00^l\#; && l, m, n \in \mathbb{N}_0.
\end{aligned}$$

Again, we need to distinguish whether $\mathrm{e}(t_R)$ is even (π_{k+4}) or odd (π_{k+5}). This concludes the definition of predicates for t'_R.

As $t'_L = 4(\mathrm{e}(t_L) \operatorname{div} 2) + 2\,\mathrm{e}(d) + (\mathrm{e}(t_R) \bmod 2)$, we know that for every R-movement in a valid computation, the congruence class of $\mathrm{e}(t'_L)$ modulo 4 is either $2\,\mathrm{e}(d)$ or $2\,\mathrm{e}(d) + 1$, depending on $t_{R,0}$ (recall that $t_{R,0}$ is the first cell to the right of the head). Thus, regardless of that tape cell, the congruence classes of $2 - \mathrm{e}(d)$ and $3 - \mathrm{e}(d)$ modulo 4 can be excluded with the following two predicates:

$$\begin{aligned}
&\pi_{k+6} : \sigma(\alpha_1) \text{ contains } \#00^{2m+\mathrm{e}(a)}\#0^i\#\#0^+\#00^{4n+(2-\mathrm{e}(d))}\#; && m, n \in \mathbb{N}_0,\\
&\pi_{k+7} : \sigma(\alpha_1) \text{ contains } \#00^{2m+\mathrm{e}(a)}\#0^i\#\#0^+\#00^{4n+(3-\mathrm{e}(d))}\#; && m, n \in \mathbb{N}_0.
\end{aligned}$$

Furthermore, depending on $t_{R,0}$, we can also exclude the class $2\,\mathrm{e}(d) + (1 - \mathrm{e}(t_{R,0}))$ modulo 4. For this, we need to distinguish the two possible cases for

$\mathrm{e}(t_{R,0})$ and define the predicates

$\pi_{k+8} : \sigma(\alpha_1)$ contains $\#00^{2l}\#00^{2m+\mathrm{e}(a)}\#0^i\#\#00^l\#00^{4n+2\,\mathrm{e}(d)+1}\#;\quad l,m,n \in \mathbb{N}_0,$

$\pi_{k+9} : \sigma(\alpha_1)$ contains $\#00^{2l+1}\#00^{2m+\mathrm{e}(a)}\#0^i\#\#00^l\#00^{4n+2\,\mathrm{e}(d)}\#;\quad l,m,n \in \mathbb{N}_0.$

Finally, the last two predicates handle the case where $\mathrm{e}(t'_L)$ is of the right congruence class modulo 4, but too small. Again, we need to distinguish the two possible values of $\mathrm{e}(t_{R,0})$:

$\pi_{k+10} : \sigma(\alpha_1)$ contains $\#00^{2l}\#00^{2(m+n+1)\,\mathrm{e}(a)}\#0^i\#\#00^l\#00^{4m+2\,\mathrm{e}(d)}\#;\quad l,m,n \in \mathbb{N}_0,$

$\pi_{k+11} : \sigma(\alpha_1)$ contains $\#00^{2l+1}\#00^{2(m+n+1)\,\mathrm{e}(a)}\#0^i\#\#00^l\#00^{4m+2\,\mathrm{e}(d)+1}\#;\quad l,m,n \in \mathbb{N}_0.$

Note that the last four predicates already assume t'_R has transitioned correctly. This is acceptable, as errors on this side of the tape are handled by the previous predicates.

We see that every one of the 14 R-movements of U requires 12 infix predicates π_k to π_{k+11}. Of these, π_{k+2} and π_{k+3} use 2 parameters, all others use 3 parameters. Adding these 168 predicates allows us to conclude that $\mu = 333$ was indeed a correct choice for the first claim.

For the second claim, we also add the suffix predicate

$$\pi_{334} : \sigma(\alpha_1) \text{ ends on } \#0^{2n+1}\#0^{10}\#\#; \qquad n \in \mathbb{N}_0.$$

This predicate eliminates all computations where the last configuration is not accepting.

Now, if there is a $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$, $\sigma(\alpha_1)$ encodes a computation of U that starts in I and reaches the state q_{10}, while $\mathrm{e}(t_L)$ is odd. That means that the machine reads 1 in q_{10} and halts. On the other hand, if there is a valid computation $(C_i)_{i=0}^n$ with $C_0 = I$, we can define σ by $\sigma(\alpha_1) := \mathrm{enc}(C)$ and (for example) $\sigma(\alpha_2) := 0^{|\sigma(\alpha_1)|}$. Then none of the predicates is satisfied, and $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$.

Thus, for both claims, $L_{\mathrm{E},\Sigma}(\alpha) \setminus L_{\mathrm{E},\Sigma}(\beta) = \emptyset$ if and only if $\mathrm{VALC}_{\mathrm{E}}(I) = \emptyset$. As I was chosen freely, this question must be undecidable.

All that remains is to count the number of variables in β. For the first claim, the types of predicates are distributed as follows:

1. 1 infix predicate with no parameter (π_7),
2. 1 infix predicate with one parameter (π_4),
3. 133 infix predicates with two parameters (π_5, π_8 to π_{66}, 3 per L-instruction, 2 per R-instruction),
4. 195 infix predicates with three parameters (π_6, π_{67} to π_{90}, 2 per L-instruction, 10 per R-instruction).

Therefore, in the first case, we have

$$\begin{aligned} |\operatorname{var}(\beta)| &= \mu + 2\kappa + 13 + 5 + 6 + 133 \cdot 7 + 195 \cdot 8 \\ &= 333 + 6 + 13 + 5 + 6 + 931 + 1560 = 2854. \end{aligned}$$

Thus, our construction proves that the inclusion problem for $\mathrm{ePAT}_{3,\Sigma}$ in $\mathrm{ePAT}_{2854,\Sigma}$ is undecidable.

The suffix predicate π_{334} uses one parameter and requires 6 additional variables (as μ needs to be increased by one), bringing the total amount of variables in β to 2860. This demonstrates undecidability of the inclusion problem for $\mathrm{ePAT}_{2,\Sigma}$ in $\mathrm{ePAT}_{2860,\Sigma}$. □

4.3. Proof of Theorem 3.10

Proof. Here, for any given $N \geq 1$, we use the construction to decide the emptiness of $\mathrm{TRIV}_{\mathrm{E}}(N)$.

Let $\kappa{:=}2$, $\mu{:=}10$, $\alpha_1{:=}\#0^N\#\,x\,\#0\#$ and $\alpha_2{:=}y$, where x and y are distinct variables. Due to the results in Section 4.1, we know that if there is a substitution σ with $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$, then

$$\sigma(\alpha_1) \subseteq \#0^N\#\,(0^+\#)^+\,0\#.$$

Therefore, every word from this set difference is already an encoding of a finite sequence over $\mathbb{N}_1$, with N as the first, and 1 as the last number. All that remains is to choose predicates π_4 to π_μ that describe every pair of successive numbers n_i and n_{i+1} where $n_{i+1} \neq \mathcal{C}(n_i)$.

We begin with the cases where $n_{i+1} > \mathcal{C}(n_i)$, which are handled by the following two predicates:

$$\pi_4 : \sigma(\alpha_1) \text{ contains a factor } \#0^{2m}\#0^{m+1} \text{ for some } m \in \mathbb{N}_0,$$
$$\pi_5 : \sigma(\alpha_1) \text{ contains a factor } \#0^{2m+1}\#0^{6m+3+2} \text{ for some } m \in \mathbb{N}_0.$$

It is easy to see that π_4 is satisfied if and only if the encoded sequence contains successive numbers n_i and n_{i+1} where n_i is even, and $n_{i+1} > \frac{1}{2}n_i = \mathcal{C}(n_i)$. Likewise, π_5 does the same for odd n_i: If n_i is odd, there is an $m \in \mathbb{N}_0$ with $n_i = 2m+1$, and $\mathcal{C}(n_i) = 3n_i + 1 = 6m + 3 + 1$.

Next, we define a predicate that describes all cases where n_i is even, and $n_{i+1} < \mathcal{C}(n_i)$:

$$\pi_6 : \sigma(\alpha_1) \text{ contains a factor } \#0^{2m+2n+2}\#0^m\# \text{ for some } m, n \in \mathbb{N}_0.$$

Obviously, if this predicate is satisfied, n_i is even, and $n_{i+1} < \mathcal{C}(n_i)$. For the other direction, let n_i be even, $n_{i+1} < \mathcal{C}(n_i)$, and define $m{:=}n_i$, $n{:=}\frac{1}{2}n_i - n_{i+1} - 1$. Then $2m + 2n + 2 = n_i$, which means that the corresponding substitution satisfies this predicate.

Capturing all cases where n_i is odd and $n_{i+1} < \mathcal{C}(n_i)$ is a little bit more involved. We define the following four predicates:

$$\pi_7 : \sigma(\alpha_1) \text{ contains a factor } \#0^{2m+1}\#0^{2n+1}\# \text{ for some } m, n \in \mathbb{N}_0,$$
$$\pi_8 : \sigma(\alpha_1) \text{ contains a factor } \#0^{2m+1}\#0^{6n}\# \text{ for some } m, n \in \mathbb{N}_0,$$
$$\pi_9 : \sigma(\alpha_1) \text{ contains a factor } \#0^{2m+1}\#0^{6n+2}\# \text{ for some } m, n \in \mathbb{N}_0,$$
$$\pi_{10} : \sigma(\alpha_1) \text{ contains a factor } \#0^{2m+2n+3}\#0^{6n+4}\# \text{ for some } m, n \in \mathbb{N}_0.$$

By definition of the Collatz function, if n_i is odd, then $\mathcal{C}(n_i)$ must be congruent to 4 modulo 6. The first three of these predicates handle all the cases where n_i is odd, but n_{i+1} is in the wrong congruence class modulo 6; i. e., either n_{i+1} is odd (π_7) or division by 6 leads to a remainder of 0 or 2 (π_8 and π_9, respectively). The remaining predicate π_{10} is satisfied if and only if n_i is odd, n_{i+1} is congruent to 4 modulo 6, and $n_{i+1} < \mathcal{C}(n_i)$.

Thus, if there is a $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$, $\sigma(\alpha_1)$ contains an encoding of a sequence $n_0, \ldots, n_l$ for some $l \geq 2$ with $n_i = \mathcal{C}^i(N)$ for every i, and $n_l = 1$. This means that N leads the Collatz function to the trivial cycle, and thus, $\mathrm{TRIV}_{\mathrm{E}}(N) \neq \emptyset$.

On the other hand, assume $\mathrm{TRIV}_{\mathrm{E}}(N) \neq \emptyset$. Then there is an $l \geq 2$ with $\mathcal{C}^l(N) = 1$. Let $\sigma(x){:=}0^{\mathcal{C}^1(N)}\#0^{\mathcal{C}^2(N)}\#\ldots\#0^{\mathcal{C}^{l-1}(N)}$ and $\sigma(y){:=}0^m$, where $m{:=}|\sigma(\alpha_1)|$. As we have seen, σ satisfies none of the predicates π_1 to π_{10}, and thus, $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$.

The total number of variables in β can be calculated as follows: First, we require $\mu + 2\kappa + 13$ variables from the basic construction and π_1 to π_3. As π_4 and π_5 are infix predicates with one numerical parameter, they each require 6 additional variables. Likewise, the predicates π_6 to π_{10} require 7 variables each. Thus, β contains $\mu + 2\kappa + 13 + 12 + 35 = 74$ different variables. □

4.4. Proof of Theorem 3.12

Proof. In order to decide the emptiness of $\mathrm{NTCC}_{\mathrm{E}}$, we choose $\kappa{:=}2$, $\mu{:=}11$ $\alpha_1{:=}\#x_1\#x_2\#x_3\#x_2\#$ and $\alpha_2{:=}x_2y$, where x_1, x_2, x_3 and y are pairwise distinct variables.

We use the same predicates π_4 to π_{10} as in the previous section for the encoding of $\mathrm{TRIV}_{\mathrm{E}}(N)$, and the additional predicate

$$\pi_{11} : \sigma(\alpha_1) \text{ contains the factor } \#0\#.$$

Considering the previous section, it is easy to see that $L_{\mathrm{E},\Sigma}(\alpha) \setminus L_{\mathrm{E},\Sigma}(\beta) \neq \emptyset$ if and only if there is a number leading to a non-trivial cycle: Assume there is a substitution σ with $\sigma(\alpha) \notin L_{\mathrm{E},\Sigma}(\beta)$. This substitution satisfies none of the predicates π_1 to π_{10}, and must be of 2-E-good form. Therefore, $\sigma(x_2) \in 0^+$, which means that the sequence encoded in $\sigma(\alpha_1)$ contains the number $|\sigma(x_2)|$ at least twice. Due to π_{11}, this sequence does not contain the number 1, which means that the encoded sequence contains a non-trivial cycle of the Collatz function. Thus, $\mathrm{NTCC}_{\mathrm{E}}$ is empty if and only if $L_{\mathrm{E},\Sigma}(\alpha) \setminus L_{\mathrm{E},\Sigma}(\beta)$ is empty.

As π_{11} is a 2-simple infix predicate with no numerical parameters, its subpatterns require five new variables in β (in addition to x_{11}), bringing the total number of variables in β to 80.

Therefore, any algorithm that decides the inclusion problem of $\mathrm{ePAT}_{4,\Sigma}$ in $\mathrm{ePAT}_{80,\Sigma}$ can be used to determine in finite time whether there exists any non-trivial cycle of the Collatz function by deciding whether $L_{\mathrm{E},\Sigma}(\alpha) \subseteq L_{\mathrm{E},\Sigma}(\beta)$. □

4.5. The Construction for NE-Patterns

This construction is used by the proofs of Theorem 3.9, Theorem 3.11 and Theorem 3.13, which can be found in Section 4.6, 4.7 and 4.8 (respectively).

Let $\Sigma{:=}\{0,\#\}$ and let V be the respective set of valid computations, i. e., $\mathrm{TRIV}_{\mathrm{NE}}(N)$, $\mathrm{NTCC}_{\mathrm{NE}}$ or $\mathrm{VALC}_{\mathrm{NE}}(I)$, and let $\overline{\mathrm{V}}$ denote the corresponding complement. Our goal is to construct patterns $\alpha, \beta \in \mathrm{Pat}_\Sigma$ such that $L_{\mathrm{NE},\Sigma}(\alpha) \subseteq L_{\mathrm{NE},\Sigma}(\beta)$ if and only if $\overline{\mathrm{V}} = \emptyset$.

In this section, β is defined first, because a part of β is needed to define α.

We define

$$\beta{:=}a\ b\ \#^5\ a\ x_1 \ldots x_\mu\ b\ \#^5\ r_1\ \hat{\beta}_1\ r_2\ \hat{\beta}_2 \ldots r_\mu\ \hat{\beta}_\mu\ r_{\mu+1}$$

and for all $i \in \{1, \ldots, \mu\}$,

$$\hat{\beta}_i{:=}0x_i^4 0\ \gamma_i\ 0x_i^4 0\ \delta_i\ 0x_i^4 0,$$

where a, b, $r_{\mu+1}$ and all r_i and x_i are distinct variables and all $\gamma_i, \delta_i \in \mathrm{Pat}_\Sigma$ are patterns. All variables r_i and $r_{\mu+1}$ occur only once and the variables a and b occur only twice in the whole pattern β. The patterns γ_i and δ_i shall be defined later; for now we only mention:

1. $\mathrm{var}(\gamma_i\delta_i) \cap \mathrm{var}(\gamma_j\delta_j) = \emptyset$ for all $i, j \in \{1, \ldots, \mu\}$ with $i \neq j$,
2. $x_k \notin \mathrm{var}(\gamma_i\delta_i)$ for all $i, k \in \{1, \ldots, \mu\}$.

Any variable in $\mathrm{var}(\gamma_i\delta_i)$ does not appear outside these two factors. In contrast to the E-construction, the patterns γ_i and δ_i are not terminal-free, and the patterns η_i are not used.

Now define

$$\alpha{:=}0^{\mu+1}\ \#^5\ 0^\mu\#0^\mu\ \#^5\ t\ v\ 0\ \alpha_1\ 0\ v\ 0\ \alpha_2\ 0\ v\ t,$$

where $v{:=}0\#^5 0$, t is another terminal-string, α_1 is a pattern not containing $\#^3$ as a factor, and α_2 is a pattern not containing $\#$. To define t we need the nonerasing substitution $\psi : (\mathrm{var}(\beta) \cup \Sigma)^* \to \Sigma^*$ with $\psi(x) = 0$ for all $x \in \mathrm{var}(\beta)$. Now $t{:=}\psi(r_1\hat{\beta}_1 \ldots r_\mu\hat{\beta}_\mu r_{\mu+1})$.

Lemma 4.4. *All $\psi(\hat{\beta}_i)$ with $i \in \{1, \ldots, \mu\}$ and t begin and end with 0 and do not contain $\#^4$ as a factor.*

Proof. For this proof all predicates have to be already defined.

Outside of γ_i and δ_i with $i \in \{1, \ldots, \mu\}$ no $\#$ occurs in $\hat{\beta}_i$. The only δ_i with a factor $\#$ is δ_2 and the factor occurs only once. The γ_i with the longest factor of $\#$s is γ_1 with one factor $\#^\kappa$ and $\kappa \in \{2, 3\}$. None of the $\psi(\hat{\beta}_i)$ contains the factor $\#^4$, as ψ maps all variables to 0s, and thus,

$$\psi(\hat{\beta}_i) = \psi\left(0x_i^4 0\ \gamma_i\ 0x_i^4 0\ \delta_i\ 0x_i^4 0\right) = 0^6\ \psi(\gamma_i)\ 0^6\ \psi(\delta_i)\ 0^6.$$

Thus, t also does not contain the factor $\#^4$ and begins and ends with 0. $\square$

Let H^+ be the set of all nonerasing substitutions $\sigma : (\Sigma \cup \mathrm{var}(\alpha_1\alpha_2))^* \to \Sigma^*$. We interpret each pair (γ_i, δ_i) as a predicate $\pi_i : H^+ \to \{0, 1\}$ in such a way that $\sigma \in H^+$ *satisfies* π_i if there exists a nonerasing substitution τ :

$(\mathrm{var}(\gamma_i\delta_i)\cup\Sigma)^* \to \Sigma^*$ with $\tau(\gamma_i) = \sigma(0\alpha_1 0)$ and $\tau(\delta_i) = \sigma(0\alpha_2 0)$. Later, we shall see that $L_{\mathrm{NE},\Sigma}(\alpha)\backslash L_{\mathrm{NE},\Sigma}(\beta)$ contains exactly those $\sigma(\alpha)$ for which σ does not satisfy any of π_1 to π_μ, and choose these predicates to describe $\overline{\mathrm{V}}$. The encoding of $\overline{\mathrm{V}}$ shall be handled by π_4 to π_μ, as these predicates describe a complete list of sufficient criteria for membership in $\overline{\mathrm{V}}$. Again we need a considerable amount of technical preparations.

Choose a fixed $\kappa \in \{2,3\}$. A nonerasing substitution σ is of *κ-NE-bad form* if $\sigma(0\alpha_1 0)$ contains $\#^\kappa$ as a factor, or if $\sigma(0\alpha_2 0)$ contains $\#$. Otherwise, σ is of *κ-NE-good form.*

The predicates π_1 and π_2 handle all cases where σ is of κ-NE-bad form and are defined by

$$\begin{aligned} &\gamma_1 := y_{1,1}\ \#^\kappa\ y_{1,2}, &\qquad \gamma_2 := 0 y_2 0,\\ &\delta_1 := 0\hat{y}_1 0, &\qquad \delta_2 := \hat{y}_{2,1}\ \#\ \hat{y}_{2,2}, \end{aligned}$$

where $y_{1,1}$, $y_{1,2}$, $\hat{y}_1$, y_2, $\hat{y}_{2,1}$ and $\hat{y}_{2,2}$ are pairwise distinct variables.

Lemma 4.5. *A nonerasing substitution $\sigma \in H^+$ is of κ-NE-bad form if and only if σ satisfies π_1 or π_2.*

Proof. We begin with the *only if* direction. If $\sigma(0\alpha_1 0) = w_1\#^\kappa w_2$ for some $w_1 \in 0\Sigma^*$ and $w_2 \in \Sigma^* 0$, choose $\tau(y_{1,1}) := w_1$, $\tau(y_{1,2}) := w_2$ and $\tau(0\hat{y}_1 0) := \sigma(0\alpha_2 0)$. Then $\tau(\gamma_1) = \sigma(0\alpha_1 0)$ and $\tau(\delta_1) = \sigma(0\alpha_2 0)$; thus, σ satisfies π_1.

If $\sigma(0\alpha_2 0) = w_1\# w_2$ for some $w_1 \in 0\Sigma^*$ and $w_2 \in \Sigma^* 0$, let $\tau(0y_2 0) := \sigma(0\alpha_1 0)$, $\tau(\hat{y}_{2,1}) := w_1$ and $\tau(\hat{y}_{2,2}) := w_2$. It is easy to see that σ satisfies π_2.

For the *if* direction, if σ satisfies π_1, then there exists a nonerasing substitution τ with $\tau(\gamma_1) = \sigma(0\alpha_1 0)$. Then, by definition of γ_1,

$$\sigma(0\alpha_1 0) = \tau(y_{1,1})\#^\kappa \tau(y_{1,2,}),$$

which means that σ is of κ-NE-bad form.

Analogously, if σ satisfies π_2, then $\sigma(0\alpha_2 0)$ contains the terminal $\#$, and σ is of κ-NE-bad form. $\square$

The reason for putting the additional 0 left and right of α_1 and α_2 is to ensure that the predicates can be almost the same as in the erasing case. In the erasing case, γ_i and δ_i often had separate variables at the borders. For example, γ_1 has the border-variables $y_{1,1}$ and $y_{1,2}$. If π_1 is satisfied by σ, then one factor $\#^\kappa$ in $\sigma(\alpha_1)$ can be chosen and $y_{1,1}$ can be mapped to the terminal-string in $\sigma(\alpha_1)$ to the left of this $\#^\kappa$, and $y_{1,1}$ to the terminal-string to the right of this $\#^\kappa$. In the erasing case, the variables can even be mapped to the empty word, which is obviously not possible in the nonerasing case. If we now used the same predicate π_1 for nonerasing substitutions without the additional 0 to the left and to the right of α_1, and if the only factor $\#^\kappa$ in $\sigma(\alpha_1)$ were on a border of $\sigma(\alpha_1)$, then π_1 would not be satisfied by σ.

With the additional 0s, the border-variable for such σ could be mapped to only 0 and σ satisfies π_1. If we want to reuse a predicate π_i, where a separate border-variable does not exist, we have to add a 0 at the left and/or right end of the corresponding patterns γ_i or δ_i. For example in δ_1, 0 was added at the left and at the right end.

Lemmas 4.4 and 4.5 allow us to make the following observation, which – as in the E-construction – serves as the central part of the construction and is independent of the exact shape of π_3 to π_μ:

Lemma 4.6. *For every nonerasing substitution $\sigma \in H^+$, $\sigma(\alpha) \in L_{\mathrm{NE},\Sigma}(\beta)$ if and only if σ satisfies one of the predicates π_1 to π_μ.*

Proof. We begin with the *if* direction. Assume $\sigma \in H^+$ satisfies some predicate π_i with $i \in \{1, \dots, \mu\}$. Then there exists a nonerasing substitution $\tau : (var(\gamma_i\delta_i) \cup \Sigma)^* \to \Sigma^*$ with

$$\tau(\gamma_i) = \sigma(0\alpha_1 0), \qquad \tau(\delta_i) = \sigma(0\alpha_2 0).$$

We extend τ to a nonerasing substitution τ' defined by

1. $\tau'(x) := \begin{cases} \tau(x) & \text{for all } x \in \mathrm{var}(\gamma_i\delta_i) \\ 0 & \text{for all } x \in \mathrm{var}(\gamma_j\delta_j) \text{ with } j \neq i \end{cases}$,
2. $\tau'(x_j) := \begin{cases} \# & \text{for } j = i \\ 0 & \text{for } j \neq i \end{cases}$,
3. $\tau'(r_j) := \begin{cases} \psi(r_i\hat{\beta}_i \dots r_\mu\hat{\beta}_\mu r_{\mu+1}) & \text{for } j = i \\ \psi(r_1\hat{\beta}_1 \dots r_i\hat{\beta}_i r_{i+1}) & \text{for } j = i+1 \\ 0 & \text{else} \end{cases}$,
4. $\tau'(a) := 0^{\mu-i+1}$,
5. $\tau'(b) := 0^i$.

By definition, none of the variables in $\mathrm{var}(\gamma_i\delta_i)$ appears outside the factors γ_i and δ_i. Thus, τ' can always be defined in this way. We obtain

$$\tau'(\gamma_i) = \tau(\gamma_i) = \sigma(0\alpha_1 0)$$

and

$$\tau'(\delta_i) = \tau(\delta_i) = \sigma(0\alpha_2 0).$$

In addition, it follows that

$$\begin{aligned} \tau'(a\ b\ \#^5\ a\ x_1 \dots x_\mu\ b\ \#^5) &= 0^{\mu-i+1}\ 0^i\ \#^5\ 0^{\mu-i+1}\ 0^{i-1}\ \#\ 0^{\mu-i}\ 0^i\ \#^5 \\ &= 0^{\mu+1}\ \#^5\ 0^\mu\ \#\ 0^\mu\ \#^5. \end{aligned}$$

Also

$$\begin{aligned} \tau'(\hat{\beta}_i) &= \tau'(0x_i^4 0\ \gamma_i\ 0x_i^4 0\ \delta_i\ 0x_i^4 0) \\ &= 0\#^4 0\ \tau'(\gamma_i)\ 0\#^4 0\ \tau'(\delta_i)\ 0\#^4 0 \\ &= v\ \sigma(0\alpha_1 0)\ v\ \sigma(0\alpha_2 0)\ v. \end{aligned}$$

As $\tau'(x) = \psi(x)$ for all $x \in \mathrm{var}(\hat{\beta}_j)$ with $j \neq i$, we get for all $j \neq i$

$$\tau'(\hat{\beta}_j) = \psi(\hat{\beta}_j).$$

Now we obtain

$$\begin{aligned}
\tau'(\beta) &= \tau'(ab\#^5ax_1 \dots x_\mu b\#^5\ r_1\ \hat{\beta}_1 \dots r_\mu\ \hat{\beta}_\mu\ r_{\mu+1}) \\
&= 0^{\mu+1}\#^5 0^\mu \# 0^\mu \#^5\ \tau'(r_1\ \hat{\beta}_1 \dots r_\mu\ \hat{\beta}_\mu\ r_{\mu+1}) \\
&= 0^{\mu+1}\#^5 0^\mu \# 0^\mu \#^5\ \tau'(r_1\hat{\beta}_1 \dots r_{i-1}\hat{\beta}_{i-1})\ \tau'(r_i)\ \tau'(\hat{\beta}_i)\ \dots \\
&\qquad\qquad\qquad \dots\ \tau'(r_{i+1})\ \tau'(\hat{\beta}_{i+1} \dots r_\mu\hat{\beta}_\mu r_{\mu+1}) \\
&= 0^{\mu+1}\#^5 0^\mu \# 0^\mu \#^5\ \psi(r_1\hat{\beta}_1 \dots r_{i-1}\hat{\beta}_{i-1})\ \tau'(r_i)\ \tau'(\hat{\beta}_i)\ \dots \\
&\qquad\qquad\qquad \dots\ \tau'(r_{i+1})\ \psi(\hat{\beta}_{i+1} \dots r_\mu\hat{\beta}_\mu r_{\mu+1}) \\
&= 0^{\mu+1}\#^5 0^\mu \# 0^\mu \#^5\ \psi(r_1\hat{\beta}_1 \dots r_{i-1}\hat{\beta}_{i-1})\ \psi(r_i\hat{\beta}_i \dots r_\mu\hat{\beta}_\mu r_{\mu+1})\ \tau'(\hat{\beta}_i)\ \dots \\
&\qquad\qquad \dots\ \psi(r_1\hat{\beta}_1 \dots r_i\hat{\beta}_i r_{i+1})\ \psi(\hat{\beta}_{i+1} \dots r_\mu\hat{\beta}_\mu r_{\mu+1}) \\
&= 0^{\mu+1}\#^5 0^\mu \# 0^\mu \#^5\ \psi(r_1\hat{\beta}_1 \dots r_\mu\hat{\beta}_\mu r_{\mu+1})\ \tau'(\hat{\beta}_i)\ \psi(r_1\hat{\beta}_1 \dots r_\mu\hat{\beta}_\mu r_{\mu+1}) \\
&= 0^{\mu+1}\#^5 0^\mu \# 0^\mu \#^5\ t\ \tau'(\hat{\beta}_i)\ t \\
&= 0^{\mu+1}\#^5 0^\mu \# 0^\mu \#^5\ t\ v\ \sigma(0\alpha_1 0)\ v\ \sigma(0\alpha_2 0)\ v\ t \\
&= \sigma(\alpha).
\end{aligned}$$

This proves $\sigma(\alpha) \in L_{\mathrm{NE},\Sigma}(\beta)$.

For the other direction, assume $\sigma(\alpha) \in L_{\mathrm{NE},\Sigma}(\beta)$. If σ is of κ-NE-bad form, then by Lemma 4.5, σ satisfies π_1 or π_2. Thus, assume $\sigma(0\alpha_1 0)$ does not contain $\#^\kappa$ as a factor, and $\sigma(0\alpha_2 0) \in 00^+0$. Let τ be a nonerasing substitution with $\tau(\beta) = \sigma(\alpha)$.

Now, as σ is of κ-NE-good form and, by Lemma 4.4, t begins and ends with 0 and does not contain $\#^4$ as a factor, $\sigma(\alpha)$ contains the factor $\#^5$ exactly twice. As $\sigma(\alpha) = \tau(\beta)$, the same holds for $\tau(\beta)$. Thus the equation $\sigma(\alpha) = \tau(\beta)$ can be decomposed into the system consisting of the following three equations:

$$0^{\mu+1} = \tau(a\ b), \tag{1}$$

$$0^\mu\ \#\ 0^\mu = \tau(a\ x_1 \dots x_\mu\ b), \tag{2}$$

$$t\ v\ \sigma(0\alpha_1 0)\ v\ \sigma(0\alpha_2 0)\ v\ t = \tau(r_1\ \hat{\beta}_1 \dots r_\mu\ \hat{\beta}_\mu\ r_{\mu+1}). \tag{3}$$

In equation (2) the image $\tau(x_1 \dots x_\mu)$ hast to contain the single $\#$ and has to be of length μ, as else equation (1) would not be satisfied. Then each $\tau(x_i)$ with $i \in \{1, \dots, \mu\}$ is a single terminal and thus there exist an $i \in \{1, \dots, \mu\}$ with $\tau(x_i) = \#$ and $\tau(x_j) = 0$ for all $j \neq i$. Now this i we obtain

$$\begin{aligned}
\tau(\hat{\beta}_i) &= \tau(0x_i^4 0\ \gamma_i\ 0x_i^4 0\ \delta_i\ 0x_i^4 0) \\
&= 0\#^4 0\ \tau(\gamma_i)\ 0\#^4 0\ \tau(\delta_i)\ 0\#^4 0 \\
&= v\ \tau(\gamma_i)\ v\ \tau(\delta_i)\ v.
\end{aligned}$$

The right side of equation (3) can be converted to

$$\begin{aligned}\tau(r_1\ \hat\beta_1 \ldots r_\mu\ \hat\beta_\mu\ r_{\mu+1}) &= \tau(r_1\ \hat\beta_1 \ldots r_{i-1}\ \hat\beta_{i-1}\ r_i)\ \tau(\hat\beta_i)\ \tau(r_{i+1}\ \hat\beta_{i+1} \ldots r_\mu\ \hat\beta_\mu\ r_{\mu+1})\\ &= \tau(r_1\ \hat\beta_1 \ldots r_{i-1}\ \hat\beta_{i-1}\ r_i)\ v\ \tau(\gamma_i)\ v\ \tau(\delta_i)\ v\ \tau(r_{i+1}\ \hat\beta_{i+1} \ldots r_\mu\ \hat\beta_\mu\ r_{\mu+1}),\end{aligned}$$

and thus,

$$\begin{aligned}&t\ v\ \sigma(0\alpha_1 0)\ v\ \sigma(0\alpha_2 0)\ v\ t =\\ &\qquad \tau(r_1\ \hat\beta_1 \ldots r_{i-1}\ \hat\beta_{i-1}\ r_i)\ v\ \tau(\gamma_i)\ v\ \tau(\delta_i)\ v\ \tau(r_{i+1}\ \hat\beta_{i+1} \ldots r_\mu\ \hat\beta_\mu\ r_{\mu+1}).\end{aligned}$$

As σ is of κ-NE-good form and t does not contain the factor $\#^4$, the left side of the equation contains exactly three times the factor $v = 0\#^4 0$. As the right side also contains three times this factor, the equation can be decomposed into the system consisting of the following four equations:

$$t = \tau(r_1\ \hat\beta_1 \ldots r_{i-1}\ \hat\beta_{i-1}\ r_i), \tag{4}$$

$$\sigma(0\alpha_1 0) = \tau(\gamma_i), \tag{5}$$

$$\sigma(0\alpha_2 0) = \tau(\delta_i), \tag{6}$$

$$t = \tau(r_{i+1}\ \hat\beta_{i+1} \ldots r_\mu\ \hat\beta_\mu\ r_{\mu+1}). \tag{7}$$

Due to equations (5) and (6), σ satisfies the predicate π_i. □

Thus, we can select predicates π_1 to π_μ such that $L_{\mathrm{NE},\Sigma}(\alpha) \backslash L_{\mathrm{NE},\Sigma}(\beta) = \emptyset$ if and only if $\mathrm{V} = \emptyset$. As in the E-construction, the corresponding complement $\overline{\mathrm{V}}$ of V can be described by a disjunction of predicates. The proof of Lemma 4.6 shows that if $\sigma(\alpha) = \tau(\beta)$ for nonerasing substitutions σ and τ, where σ is of κ-NE-good form, there exists exactly one i with $i \in \{3, \ldots, \mu\}$ fulfilling $\tau(0x_i^4 0) = 0\#^4 0 = v$.

Again due to technical reasons, we need a predicate that, if unsatisfied, sets a lower bound to the length of $\sigma(\alpha_2)$. If $\kappa = 2$, the predicate π_3 is defined by

$$\begin{aligned}\gamma_3 &:= y_{3,1}\ \hat y_{3,1}\ y_{3,2}\ \hat y_{3,2}\ y_{3,3},\\ \delta_3 &:= 0\ \hat y_{3,1}\ \hat y_{3,2}\ 0.\end{aligned}$$

Recall that the 0s in δ_3 are necessary due to the additional 0s to the left and to the right of α_1 and α_2. In γ_3 the 0s are missing; the reason is that the images of the border-variables will include the 0s. Hence no problem occurs if the longest sequences of 0s are on the borders of $\sigma(\alpha_1)$.

If $\kappa = 3$, we use a different predicate π_3 defined by

$$\begin{aligned}\gamma_3 &:= y_{3,1}\ \hat y_{3,1}\ y_{3,2}\ \hat y_{3,2}\ y_{3,3}\ \hat y_{3,3}\ y_{3,4},\\ \delta_3 &:= 0\ \hat y_{3,1}\ \hat y_{3,2}\ \hat y_{3,3}\ 0.\end{aligned}$$

In either case, all of $y_{3,1}$ to $y_{3,4}$ and $\hat y_{3,1}$ to $\hat y_{3,3}$ are pairwise distinct variables.

We do not need to cover cases of less than κ non-overlapping and non-touching strings of 0s in $\sigma(\alpha_1)$, as the predicates π_1 and π_2 and the later defined exact construction of α_1 ensure that there are at least κ non-overlapping, non-touching, nonempty factors of 0s in $\sigma(\alpha_1)$. The special case $|\sigma(\alpha_2)| < \kappa$, has not to be covered, because $|\alpha_2|$ shall be at least κ.

If some $\sigma \in H^+$ satisfies π_3, $\sigma(\alpha_2)$ is a concatenation of κ nonempty factors of $\sigma(\alpha_1)$. Thus, if σ does not satisfy any of π_1 to π_3, then $\sigma(\alpha_2)$ has to be longer than the κ longest non-overlapping, non-touching sequences of 0s in $\sigma(\alpha_1)$. This again allows to create a class of predicates definable by a rather simple kind of expression, which we shall use to define π_4 to π_μ in a less technical way. Note that any reasonable use of this construction requires α_2 to contain at least one variable that does not occur in α_1, as otherwise, every σ of κ-NE-good form would satisfy π_3.

Let $X_\kappa := \{\hat{x}_1, \ldots, \hat{x}_\kappa\} \subset X$, let G^+_κ denote the set of those nonerasing substitutions in H^+ that are of κ-NE-good form and let R be the set of all nonerasing substitutions $\rho : (\Sigma \cup X_\kappa)^* \to \Sigma^*$ for which $\rho(\hat{x}_i) \in 0^+$ for all $i \in \{1, \ldots, \kappa\}$. For patterns $\zeta \in (\Sigma \cup X_\kappa)^*$, we define $R(\zeta) := \{\rho(\zeta) | \rho \in R\}$.

Definition 2. A predicate $\pi : G^+_\kappa \to \{0, 1\}$ is called a κ-NE-simple predicate for $0\alpha_1 0$, if there exists a pattern $\zeta \in (\Sigma \cup X_\kappa)^*$ and languages $L_1 \in \{0\Sigma^*, \{0\}\}$ and $L_2 \in \{\Sigma^* 0, \{0\}\}$ such that a nonerasing substitution σ satisfies π if and only if $\sigma(0\alpha_1 0) \in L_1\ R(\zeta)\ L_2$. If $L_1 = 0\Sigma^*$ and $L_2 = \Sigma^* 0$, we call π an infix-predicate. If only $L_1 = \{0\}$ or $L_2 = \{0\}$, we call π a prefix-predicate or a suffix-predicate, respectively.

Again, the elements of X_κ can be understood as numerical parameters describing (concatenational) powers of 0, with now nonerasing substitutions $\rho \in R$ acting as assignments. In contrast to the E-construction, the power 0^0 is not allowed. For example, $\sigma \in G^+_\kappa$ satisfies a κ-NE-simple predicate π if an only if $\sigma(0\alpha_1 0) \in 0\Sigma^* R(\#\hat{x}_1\#\hat{x}_2 0\#\hat{x}_1)0$, means σ satisfying π if and only if $\sigma(\alpha_1)$ has a suffix of the form $\#0^m\#0^n 0\#0^m$, but now with $m, n \in \mathbb{N}_1$. This could also be written as $\#0^m\#0^+0\#0^m$, as n occurs only once in this expression. To replace the simple predicates which were used in the erasing case, where for the above example $m, n \in \mathbb{N}_0$, we could use multiple simple predicates in the nonerasing case. In the above example, this could be done by three additional simple predicates where $m = 0$ and $n \in \mathbb{N}_1$ or $m \in \mathbb{N}_1$ and $n = 0$ or $m, n = 0$. Later we shall see how these simple predicates can be regardless combined into one predicate, which will lead to almost the same predicates as in the E-construction.

Using π_3, our construction is able to express all κ-NE-simple predicates:

Lemma 4.7. *For every κ-NE-simple predicate π_S over n numerical parameters with $n \leq \kappa$, there exists a predicate π defined by patterns γ and δ such that for all nonerasing substitutions $\sigma \in G^+_\kappa$:*

1. *if σ satisfies π_S, then σ also satisfies π or π_3,*
2. *if σ satisfies π, then σ also satisfies π_S.*

Proof. We first consider the case of $L_1 = 0\Sigma^*$ and $L_2 = \Sigma^* 0$. Assume π_S is a κ-NE-simple predicate, and $\zeta \in (\Sigma \cup X_\kappa)^*$ is a pattern such that $\sigma \in G^+_\kappa$ satisfies π_S if and only if $\sigma(0\alpha_1 0) \in L_1 R(\zeta) L_2$. Then define $\gamma := y_1 \zeta y_2$. Furthermore, let

$\delta{:=}0\theta\hat{y}0$, whereas θ is the concatenation of all $\hat{x} \in \mathrm{var}(\zeta)$. Note that $\hat{x}_1$, $\hat{x}_2$, $\hat{x}_3$, y_1 and y_2 are pairwise distinct variables.

Now, assume $\sigma \in G_\kappa^+$ satisfies π_S. Then there exist words $w_1 \in 0\Sigma^*$ and $w_2 \in \Sigma^*0$ and a nonerasing substitution $\rho \in R$ such that $\sigma(0\alpha_1 0) = w_1\rho(\zeta)w_2$. If $\sigma(\alpha_2)$ is not longer than any κ non-overlapping, non-touching factors of the form 0^+ of $\sigma(\alpha_1)$ combined, π_3 is satisfied. Otherwise, we can define τ by setting $\tau(y_1){:=}w_1$, $\tau(y_2){:=}w_2$ and $\tau(\hat{x}_i){:=}\rho(\hat{x}_i)$ for all $i \in \{1, \ldots, \kappa\}$. Finally, let $\tau(\hat{y}){:=}0^m$, where

$$m{:=}|\sigma(\alpha_2)| - \sum_{\hat{x} \in \mathrm{var}(\zeta)} |\tau(\hat{x})|$$

($m > 0$ holds, as σ does not satisfy π_3). Then

$$\begin{aligned}\tau(\gamma) &= \tau(y_1)\tau(\zeta)\tau(y_2) = w_1\rho(\zeta)w_2 = \sigma(0\alpha_1 0),\\ \tau(\delta) &= 00^{|\sigma(\alpha_2)|}0 = \sigma(0\alpha_2 0).\end{aligned}$$

Therefore, σ satisfies π, which concludes this direction.

For the other direction, assume $\sigma \in G_\kappa^+$ satisfies π. Then there is a nonerasing substitution τ such that $\sigma(0\alpha_1 0) = \tau(\gamma)$ and $\sigma(0\alpha_2 0) = \tau(\delta)$. By definition $\tau(y_1) \in 0\Sigma^*$ and $\tau(y_2) \in \Sigma^*0$. If we define $\rho(\hat{x}_i){:=}\tau(\hat{x}_i)$ for all $\hat{x}_i \in \mathrm{var}(\delta)$, we see that $\sigma(0\alpha_1 0) \in L_1 R(\zeta) L_2$ holds. Thus, σ satisfies π_S as well.

The other three cases for choices of L_1 and L_2 can be handled analogously by omitting y_1 or y_2 as needed. □

Roughly speaking, if σ does not satisfy π_3, $\sigma(\alpha_2)$ (which is in 0^+, due to $\sigma \in G_\kappa^+$) is long enough to provide building blocks for κ-NE-simple predicates using variables from X_κ.

Using almost the same predicates as in the E-construction, we need six additional predicates. These predicates are necessary, as we use some slightly different definitions. Numbers $i \in \mathbb{N}_0$ or $j \in \mathbb{N}_1$ are encoded as 00^i or 0^j in the erasing case, but shall be encoded as $0^6 00^i$ or $0^6 0^j$ in the present, nonerasing case. Because of these changes we need predicates, which are satisfied by all $\sigma \in H^+$ with κ-NE-good form, whereas $\sigma(\alpha_1)$ contains a factor $\#0^n\#$ with $1 \leq n \leq 6$. Only factors of this form have to be covered, considering the κ-NE-good form of σ and the exact construction of α_1. Each of the six predicates π_4 to π_9 covers one of the six options of n:

$$\begin{aligned}\pi_4 &: \sigma(\alpha_1) \text{ contains } \#0^1\#,\\ \pi_5 &: \sigma(\alpha_1) \text{ contains } \#0^2\#,\\ \pi_6 &: \sigma(\alpha_1) \text{ contains } \#0^3\#,\\ \pi_7 &: \sigma(\alpha_1) \text{ contains } \#0^4\#,\\ \pi_8 &: \sigma(\alpha_1) \text{ contains } \#0^5\#,\\ \pi_9 &: \sigma(\alpha_1) \text{ contains } \#0^6\#.\end{aligned}$$

If $\sigma \in H^+$ is of κ-NE-good form and does not satisfy any predicate π_4 to π_9, then every nonempty string of 0s between two #s in $\sigma(\alpha_1)$ has at least a length of seven.

The predicates π_4 to π_9 are the only predicates that were newly defined, instead of being obtained by modifying predicates from the E-construction.

The additional six 0s in every nonempty factor of 0s, cause additional 0s in the definition of the predicates π_{10} to π_μ. For example, the predicate π_8 in Section 4.3 was defined to be

$$\sigma(\alpha_1) \text{ contains a factor } \#0^{2m+1}\#0^{6n}\# \text{ for some } m,n \in \mathbb{N}_0.$$

In the nonerasing case we add six 0s to every nonempty factor of 0s. Note that 0^{6n} does not count as possibly empty, as $n \neq 0$ if σ is of 2-E-good form. We now would like to define the corresponding predicate by

$$\sigma(\alpha_1) \text{ contains a factor } \#0^6 0^{2m+1}\#0^6 0^{6n}\# \text{ for some } m,n \in \mathbb{N}_0,$$

but, as said before, only $m, n \in \mathbb{N}_1$ is possible in the nonerasing case. Normally we would have to split the predicate into multiple predicates, but thanks to the additional 0s we can define the predicate by

$$\sigma(\alpha_1) \text{ contains a factor } \#0^{6-2}0^{2m+1}\#0^{6-6}0^{6n}\# \text{ for some } m,n \in \mathbb{N}_1.$$

Whenever using a numerical parameter, we reduce the additional 0s by one and the numerical parameters are in $\mathbb{N}_1$. In all cases of the E-construction, the number of occurrences of numerical parameters in a factor of 0s is never larger than six (for example, $0^{2n+2}0^{3m}$ would have five occurrences). So with six additional 0s we can use almost the same predicates as in the erasing case.

Now we can count the number of different variables outside the predicates π_{10} to π_μ. Every predicate has corresponding x_i and r_i. Additional we have the variables $r_{\mu+1}$, a and b. Each of the predicates π_1, π_2 and π_4 to π_9 uses three more variables, π_3 uses $2\kappa + 1$ additional variables. In total, β without γ_{10} to γ_μ and δ_{10} to δ_μ contains $2\mu + 2\kappa + 28$ variables.

Each of the remaining predicates π_{10} to π_μ requires:

1. one variable for y_i,
2. one variable for each numerical parameter (or star/plus),
3. one additional variable if it is a prefix or a suffix predicate,
4. two additional variables if it is an infix predicate.

Thus every predicate requires at least 1 and at most 6 variables.

4.6. Proof of Theorem 3.9

Proof. Let I be any configuration of U. Analogously to the proof of Theorem 3.8, we construct patterns to decide whether $\mathrm{VALC}_{\mathrm{NE}}(I) = \emptyset$. The predicates for the proofs of the two claims of this theorem are almost similar, they differ only in the choice of α_1 and α_2 and in an additional predicate for the second claim. In either case, we choose $\kappa{:=}3$.

For the first claim of the theorem, we choose

$$\alpha_1{:=}\#\#enc(I)\#\#x_1\#x_2x_2\#0^6 0^{10}\#\#, \qquad \alpha_2{:=}x_2y0,$$

where x_1, x_2 and y are pairwise distinct variables; for the second,

$$\alpha_1{:=}\#\#enc(I)\#\#x\#0^6 0^{10}\#\#, \qquad \alpha_2{:=}y00,$$

where x and y are distinct variables.

The 0s in α_2 in both cases ensure $\sigma(\alpha_2)$ having a length of at least $\kappa = 3$. This does not affect the proofs.

As explained in Section 4.5, all predicates of Section 4.2 can be converted into predicates for the nonerasing case. The reasoning does not change and the results of Section 4.2 can be transfused to nonerasing pattern languages. Again the only difference from the erasing case lies in the additional 0s in the definition of $\mathrm{VALC}_{\mathrm{NE}}(I)$, in parts of α_1 and in the predicates.

Thus, the inclusion problems for $\mathrm{nePAT}_{3,\Sigma}$ in $\mathrm{nePAT}_{2554,\Sigma}$ and for $\mathrm{nePAT}_{2,\Sigma}$ in $\mathrm{nePAT}_{2558,\Sigma}$ are undecidable. □

4.7. Proof of Theorem 3.11.

Proof. Let $N \geq 1$. As in the proof of Theorem 3.10, we construct patterns to decide whether $\mathrm{TRIV}_{\mathrm{NE}}(I) = \emptyset$. Let $\kappa{:=}2$, $\alpha_1{:=}\#0^6 0^N\#x\#0^6 0\#$ and $\alpha_2{:=}y0$, where x and y are distinct variables and $N \in \mathbb{N}_1$. The 0 in α_2 ensures $\sigma(\alpha_2)$ having a length of at least κ. This does not affect the proofs. Due to the results of Section 4.5, we know that if there is a nonerasing substitution σ with $\sigma(\alpha) \notin L_{\mathrm{NE},\Sigma}(\beta)$, then

$$\sigma(\alpha_1) \subseteq \#0^6 0^N\# \left(0^6 0^+\#\right)^+ 0^6 0\#.$$

Without the six additional 0s in every string of 0s, each word of this set would be the same word as in Section 4.3. The predicates π_4 to π_{10} of Section 4.3 can be converted into predicates π_{10} to π_{16} as seen in Section 4.5. The whole reasoning is the same as in the erasing case, apart from six additional 0s in the encoding.

Assume $\sigma(\alpha) \notin L_{\mathrm{NE},\Sigma}(\beta)$, thus, none of the predicates π_1 to π_{16} is satisfied by σ. Then $\sigma(\alpha_1)$ has to be the encoding of a sequence $n_0, \dots, n_l$ for some $l \geq 2$ with $n_i = \mathcal{C}^i(N)$ for all $i \in \{0, \dots, l\}$ and especially $n_l = 1$. This is possible only if N leads the Collatz function into the trivial cycle, and thus, $\mathrm{TRIV}_{\mathrm{NE}}(N) \neq \emptyset$. Now assume $\mathrm{TRIV}_{\mathrm{NE}}(N) \neq \emptyset$. This means that N leads the Collatz function into a trivial cycle, and thus, there is an $l \geq 2$ with $\mathcal{C}^l(N) = 1$. If we choose $\sigma(x){:=}0^{\mathcal{C}^1(N)}\#0^{\mathcal{C}^2(N)}\#\dots\#0^{\mathcal{C}^{l-1}(N)}$ and $\sigma(y){:=}0^m$, where m is big enough (for example, $m{:=}|\sigma(\alpha_1)|$), none of the predicates π_1 to π_{16} is satisfied by σ and thus, $\sigma(\alpha) \notin L_{\mathrm{NE},\Sigma}(\beta)$.

The pattern α contains only two variables. The number of predicates μ is 16. We can determine the number of different variables in β. As each of the predicates π_{10} to π_{16} needs two variables less than the corresponding erasing predicate, β contains $(2\mu + 2\kappa + 28) + 8 + 25 = 97$ different variables. □

4.8. Proof of Theorem 3.13

As in the proof of Theorem 3.12, we construct patterns to decide whether $\mathrm{NTCC}_{\mathrm{NE}} = \emptyset$.

For this theorem, we choose $\kappa{:=}2$, $\alpha_1{:=}\#x_1\#x_2\#x_3\#x_2\#$ and $\alpha_2{:=}x_2y$, where x_1, x_2, x_3 and y are pairwise distinct variables.

We use the same predicates π_{10} to π_{16} as in Section 4.7. The additional predicate π_{11} in Section 4.4 can be converted into the predicate π_{17} as seen in Section 4.5.

As in Section 4.7, the remaining reasoning is equal to the reasoning in the erasing case.

Thus, $\mathrm{NTCC}_{\mathrm{NE}} = \emptyset$ if and only if $L_{\mathrm{NE},\Sigma}(\alpha) \subseteq L_{\mathrm{NE},\Sigma}(\beta)$.

The pattern α contains four different variables. The additional predicate π_{17} uses three new variables and generates two additional variables outside of γ_{17} and δ_{17}, differing from Section 4.7. Thus, the number of different variables in β is 102.

5. Extensions of the Main Theorems

In this section, we extend the main theorems of the previous section to larger alphabets (Section 5.1), and show that all patterns from the second class can be replaced with terminal-free patterns (Section 5.2).

5.1. Larger Alphabets

As mentioned in Lemma 5 in [7], the construction for E-patterns can be adapted to all finite alphabets $|\Sigma|$ with $|\Sigma| \geq 3$. This modification is comparatively straightforward, but would require $2(|\Sigma| - 2)$ additional predicates, and increase the number of required variables in β by $|\Sigma| - 2$ for each predicate. With additional effort, both constructions can be adapted to arbitrarily large alphabets:

Theorem 5.1. *Let Σ be a finite alphabet with $|\Sigma| \geq 3$. The following problems are undecidable:*

1. *The inclusion problem of* $\mathrm{ePAT}_{2,\Sigma}$ *in* $\mathrm{ePAT}_{2882,\Sigma}$*,*
2. *the inclusion problem of* $\mathrm{nePAT}_{2,\Sigma}$ *in* $\mathrm{nePAT}_{2580,\Sigma}$*.*

The required modifications and the proof of their correctness for the E- and the NE-construction can be found in Section 5.1.1 and Section 5.1.2. Using the same modifications to the constructions, the remaining cases from Theorems 3.8 and 3.9 and Theorems 3.10 to 3.13 can also be adapted to ternary (or larger) alphabets, using only 22 additional variables.

5.1.1. E-Construction for Larger Alphabets

The patterns $\tilde{\alpha}$ and $\tilde{\beta}$ are defined by

$$\tilde{\alpha}{:=}\alpha\ \#^4\ w\ \#^4\ w$$

with

$$w{:=}(\mathtt{a}_1\dots\mathtt{a}_n)\,\#^4\,(0\,\mathtt{a}_2\dots\mathtt{a}_n)\dots(0\,\mathtt{a}_1\dots\mathtt{a}_{j-1}\,\mathtt{a}_{j+1}\dots\mathtt{a}_n)\dots(0\,\mathtt{a}_1\dots\mathtt{a}_{n-1})\,(0\,\mathtt{a}_1\dots\mathtt{a}_n)\,0\#^3 0$$

and

$$\tilde{\beta}{:=}\beta\ \#^4\ \beta_1''\ \#^4\ \beta_2''$$

with

$$\beta_1''{:=}\tilde{y}_{1,1}\ \tilde{x}_1\ \tilde{y}_{1,2}\ \#^4\ \tilde{z}_{1,1}\ 0\ \tilde{y}_{1,1}\ \tilde{y}_{1,2}\ 0\ \tilde{z}_{1,2}\ x_{\mu+1},$$
$$\beta_2''{:=}\tilde{y}_{2,1}\ \tilde{x}_2\ \tilde{y}_{2,2}\ \#^4\ \tilde{z}_{2,1}\ 0\ \tilde{y}_{2,1}\ \tilde{y}_{2,2}\ 0\ \tilde{z}_{2,2}\ x_{\mu+2},$$

where $\tilde{y}_{1,1}$ to $\tilde{y}_{2,2}$, $\tilde{z}_{1,1}$ to $\tilde{z}_{2,2}$, $\tilde{x}_1$ and $\tilde{x}_2$ are new pairwise distinct variables and $x_{\mu+1}$ and $x_{\mu+2}$ are the additional two new x_i-variables corresponding to the later defined two new predicates.

Lemma 4.1 still applies. But with the larger Σ, $\sigma(\alpha_1)$ and $\sigma(\alpha_2)$ can contain factors $\mathtt{a}_i$ with $i \in \{1,\dots,n\}$. The two new predicates shall be satisfied by those σ, where such a factor occurs. To be able to do this with only two new predicates, without caring about $|\Sigma|$, we need the already defined additional suffixes and some observations:

Both $\tilde{\alpha}$ and $\tilde{\beta}$ contain exactly six times the factor $\#^4$. As Lemma 4.1 is not affected by the changes, $\sigma(\tilde{\alpha})$ also contains exactly six times the factor $\#^4$, if σ is of κ-E-good form.

Hence for all substitutions $\tau : (\Sigma \cup \mathrm{var}(\tilde{\beta}))^* \to \Sigma^*$ with $\tau(\tilde{\beta}) = \sigma(\tilde{\alpha})$ and σ of κ-E-good form, $\tau(\tilde{\beta}) = \sigma(\tilde{\alpha})$ can be decomposed into a system of seven equations:

$$\tau((x_1)^2\dots(x_{\mu+2})^2) = 0\#^3 0\ 0\#^3 0, \tag{1}$$
$$\tau(\hat{\beta}_1\dots\hat{\beta}_{\mu+2}) = 0\#^3 0\ \sigma(\alpha_1)\ 0\#^3 0\ \sigma(\alpha_2)\ 0\#^3 0, \tag{2}$$
$$\tau(\ddot{\beta}_1\dots\ddot{\beta}_{\mu+2}) = 0\#^3 0\ 0\#\#0\ 0\#^3 0, \tag{3}$$
$$\tau(\tilde{y}_{1,1}\ \tilde{x}_1\ \tilde{y}_{1,2}) = \mathtt{a}_1\dots\mathtt{a}_n, \tag{4}$$
$$\tau(\tilde{z}_{1,1}0\tilde{y}_{1,1}\tilde{y}_{1,2}0\tilde{z}_{1,2}x_{\mu+1}) = (0\,\mathtt{a}_2\dots\mathtt{a}_n)\dots(0\,\mathtt{a}_1\dots\mathtt{a}_{n-1})\,(0\,\mathtt{a}_1\dots\mathtt{a}_n)\,0\#^3 0, \tag{5}$$
$$\tau(\tilde{y}_{2,1}\ \tilde{x}_2\ \tilde{y}_{2,2}) = \mathtt{a}_1\dots\mathtt{a}_n, \tag{6}$$
$$\tau(\tilde{z}_{2,1}0\tilde{y}_{2,1}\tilde{y}_{2,2}0\tilde{z}_{2,2}x_{\mu+2}) = (0\,\mathtt{a}_2\dots\mathtt{a}_n)\dots(0\,\mathtt{a}_1\dots\mathtt{a}_{n-1})\,(0\,\mathtt{a}_1\dots\mathtt{a}_n)\,0\#^3 0. \tag{7}$$

The equations (2) and (3) are of no further interest in this part, but note that x_1 to $x_{\mu+2}$, $\tilde{x}_1$ and $\tilde{x}_2$ are the only variables in the left sides of these equations that also occur in other equations. For exactly one $i \in \{1, \ldots, \mu+2\}$ we get $\tau(x_i) = 0\#^3 0$ and $\tau(x_j) = \lambda$ for all $j \neq i$ by equation (1), as already seen in the proof of Lemma 4.2.

Because of equation (4), $\tau(\tilde{y}_{1,1}\tilde{y}_{1,2})$ does not contain 0 or # as a factor. Together with equation (5) we obtain $|\tau(\tilde{y}_{1,1}\tilde{y}_{1,2})| \in \{n-1, n\}$. If $\tau(x_{\mu+1}) = 0\#^3 0$, then $|\tau(\tilde{y}_{1,1}\tilde{y}_{1,2})| = n-1$, and thus, $\tau(\tilde{x}_1) = \mathtt{a}_i$ with $i \in \{1, \ldots, n\}$. If $\tau(x_{\mu+1}) = \lambda$, $\tau(\tilde{y}_{1,1}\tilde{y}_{1,2})$ could also be $\mathtt{a}_1 \ldots \mathtt{a}_n$, and thus, $\tau(\tilde{x}_1) \in \{\lambda, \mathtt{a}_1, \ldots, \mathtt{a}_n\}$.

There are no other restrictions for $\tau(\tilde{x}_1)$, as for every choosen i the equations (4) and (5) can be satisfied without additional restrictions for other equations. As the variables $\tilde{y}_{1,1}$, $\tilde{y}_{1,2}$, $\tilde{z}_{1,1}$ and $\tilde{z}_{1,2}$ do not occur outside these equations, their images do not affect the other proofs.

By equations (6) and (7) we obtain the same results for $\tau(\tilde{x}_2)$ with the two other possible choices of $\tau(x_{\mu+2})$.

Now we can define the two additional predicates $\pi_{\mu+1}$ and $\pi_{\mu+2}$:

$$\gamma_{\mu+1} := y_{\mu+1,1}\ \tilde{x}_1\ y_{\mu+1,2}, \qquad \gamma_{\mu+2} := y_{\mu+2},$$
$$\delta_{\mu+1} := \hat{y}_{\mu+1}, \qquad \delta_{\mu+2} := \hat{y}_{\mu+2,1}\ \tilde{x}_2\ \hat{y}_{\mu+2,2},$$

where $y_{\mu+1,1}$, $y_{\mu+1,2}$, $y_{\mu+2}$, $\hat{y}_{\mu+1}$, $\hat{y}_{\mu+2,1}$ and $\hat{y}_{\mu+2,2}$ are new pairwise distinct variables.

If $\sigma \in H$ of κ-E-good form satisfies $\pi_{\mu+1}$ or $\pi_{\mu+2}$, then $\sigma(\alpha_1)$ or $\sigma(\alpha_2)$ contains a factor $\mathtt{a}_i$ with $i \in \{1, \ldots, n\}$, respectively. If σ of κ-E-good form does not satisfy $\pi_{\mu+1}$ and/or $\pi_{\mu+2}$, we have to choose $\tau(\tilde{x}_1) = \lambda$ and/or $\tau(\tilde{x}_2) = \lambda$, else $\sigma(\tilde{\alpha}) \neq \tau(\tilde{\beta})$. We can not choose $\tau(\tilde{x}_1) = \lambda$, if $\pi_{\mu+1}$ is the only of the $\mu+2$ predicates satisfied by σ. Because then we would have to choose $\tau(x_{\mu+1}) = 0\#^3 0$, and thus, $\tau(\tilde{x}_2) \neq \lambda$. The corresponding follows if only $\pi_{\mu+2}$ is satisfied by σ. The predicates $\pi_{\mu+1}$ and $\pi_{\mu+2}$ break the rule that none of the elements of $\mathrm{var}(\gamma_i\delta_i\eta_i)$ occurs outside these three factors, as $\tilde{x}_1$ and $\tilde{x}_2$ do occur in the new suffix. But in this special case it leads to no problem in the proof of Lemma 4.2, as $\tau(\tilde{x}_1)$ and $\tau(\tilde{x}_2)$ are adequate delimited by the suffix and can only be λ if the corresponding patterns are not mapped to $\sigma(\alpha_1)$ and $\sigma(\alpha_2)$.

The functionality of $\pi_{\mu+1}$ and $\pi_{\mu+2}$ is the same as of π_1 and π_2 and can be shown in the same way as Lemma 4.1.

With these two additional predicates, Lemma 4.2 can again be proved without greater changes. In the whole proof μ is changed to $\mu+2$. In the first half of the proof the definition of τ' is longer, as the images of the variables, which only occur in the suffix, also have to be defined.

The additional parts of τ' with $k \in \{1, 2\}$, and π_i is the predicate that is to be satisfied, are defined by

1. $\tau'(\tilde{y}_{k,1}) := \begin{cases} \mathtt{a}_1 \ldots \mathtt{a}_{j-1} & \text{if } i = \mu+k \text{ and } \tau(\tilde{x}_k) = \mathtt{a}_j \\ \mathtt{a}_1 \ldots \mathtt{a}_n & \text{else ,} \end{cases}$
2. $\tau'(\tilde{y}_{k,2}) := \begin{cases} \mathtt{a}_{j+1} \ldots \mathtt{a}_n & \text{if } i = \mu+k \text{ and } \tau(\tilde{x}_k) = \mathtt{a}_j \\ \lambda & \text{else ,} \end{cases}$
3. $\tau'(\tilde{z}_{k,1}) := \begin{cases} \tilde{v}_j & \text{if } i = \mu+k \text{ and } \tau(\tilde{x}_k) = \mathtt{a}_j \\ (0\,\mathtt{a}_2 \ldots \mathtt{a}_n) \ldots (0\,\mathtt{a}_1 \ldots \mathtt{a}_{n-1}) & \text{else ,} \end{cases}$

4. $\tau'(\tilde{z}_{k,2}) := \begin{cases} \tilde{w}_j & \text{if } i = \mu + k \text{ and } \tau(\tilde{x}_k) = \mathtt{a}_j \\ \#^3 0 & \text{else ,} \end{cases}$

where

$$\tilde{v}_j := (0\,\mathtt{a}_2 \dots \mathtt{a}_n) \dots (0\,\mathtt{a}_1 \dots \mathtt{a}_{j-2}\,\mathtt{a}_j \dots \mathtt{a}_n),$$
$$\tilde{w}_j := (\mathtt{a}_1 \dots \mathtt{a}_j\,\mathtt{a}_{j+2} \dots \mathtt{a}_n) \dots (0\,\mathtt{a}_1 \dots \mathtt{a}_{n-1})(0\,\mathtt{a}_1 \dots \mathtt{a}_n)$$

are factors of w depending on j.

In the second half we can presume that $\sigma(\alpha_1)$ and $\sigma(\alpha_2)$ do not contain $\mathtt{a}_i$ with $i \in \{1, \dots, n\}$ as a factor, as else $\pi_{\mu+1}$ or $\pi_{\mu+2}$ would be satisfied. Also we get the above seven equations instead of only three, but the four additional equations do not affect the proof.

Nothing after Lemma 4.2 has to be changed in the E-construction to transfer the results to Σ with $|\Sigma| \geq 3$.

But, because of the additional suffix and the two new predicates, the pattern $\tilde{\beta}$ has 22 variables more than β in the E-construction.

5.1.2. NE-Construction for Larger Alphabets

In the nonerasing case the additional suffixes are simpler than in the erasing case. The pattern $\tilde{\alpha}$ and $\tilde{\beta}$ are defined by

$$\tilde{\alpha} := \alpha\ \#^5\ 0\ \mathtt{a}_1 \dots \mathtt{a}_n\ 0\ \#^5\ 0\ \mathtt{a}_1 \dots \mathtt{a}_n\ 0,$$
$$\tilde{\beta}; = \beta\ \#^5\ \tilde{y}_1\tilde{x}_1\tilde{z}_1\ \#^5\ \tilde{y}_2\tilde{x}_2\tilde{z}_2,$$

where $\tilde{x}_1$, $\tilde{x}_2$, $\tilde{y}_1$, $\tilde{y}_2$, $\tilde{z}_1$ and $\tilde{z}_2$ are new pairwise distinct variables.

In addition to this, we have to change the definition of the nonerasing substitution $\psi : \left(var(\hat{\beta}_1 \dots \hat{\beta}_{\mu+2}) \cup \Sigma\right)^* \to \Sigma^*$, to $\psi(\tilde{x}_1) = \psi(\tilde{x}_2) = \mathtt{a}_1 \dots \mathtt{a}_n$ and $\psi(x) = 0$ for $x \in \mathrm{var}(\hat{\beta}_1 \dots \hat{\beta}_{\mu+2}) \backslash \{\tilde{x}_1, \tilde{x}_2\}$. This affects the terminal-strings t in α, as $t := \psi(\hat{\beta}_1 \dots \hat{\beta}_{\mu+2})$.

Lemmas 4.4 and 4.5 are not affected by the changes. Thus, for a $\sigma \in H^+$ of κ-NE-good form, $\sigma(\tilde{\alpha})$ contains exactly three times the factor $\#^5$. As $\tilde{\beta}$ already contains the factor $\#^5$ three times, the equation $\tau(\tilde{\beta}) = \sigma(\tilde{\alpha})$, with a nonerasing substitution $\tau : (\mathrm{var}(\tilde{\beta}) \cup \Sigma)^* \to \Sigma^*$, can be decomposed into a system of four equations. To define the two additional predicates we need observations from the third and fourth equation:

$$\tau(\tilde{y}_1\tilde{x}_1\tilde{z}_1) = 0\ \mathtt{a}_1 \dots \mathtt{a}_n\ 0,$$
$$\tau(\tilde{y}_2\tilde{x}_2\tilde{z}_2) = 0\ \mathtt{a}_1 \dots \mathtt{a}_n\ 0.$$

As $\tilde{y}_1$, $\tilde{y}_2$, $\tilde{z}_1$ and $\tilde{z}_2$ do not occur outside of these equations and τ is a nonerasing substitution, we get $\tau(\tilde{x}_1), \tau(\tilde{x}_2) = \mathtt{a}_i \dots \mathtt{a}_j$ with $1 \leq i \leq j \leq n$. The images $\tau(\tilde{x}_1)$ and $\tau(\tilde{x}_2)$ are nonempty factors of $\mathtt{a}_1 \dots \mathtt{a}_n$. This includes factors $\mathtt{a}_i$ with $i \in \{1, \dots, n\}$. The longer factors are not needed for the proofs, and do not increase the set of σ satisfying the new predicates.

Now we can define the two additional predicates $\pi_{\mu+1}$ and $\pi_{\mu+2}$:

$$\gamma_{\mu+1} := y_{\mu+1,1}\ \tilde{x}_1\ y_{\mu+1,2}, \qquad \gamma_{\mu+2} := 0\ y_{\mu+2}\ 0,$$
$$\delta_{\mu+1} := 0\ \hat{y}_{\mu+1}\ 0, \qquad \delta_{\mu+2} := \hat{y}_{\mu+2,1}\ \tilde{x}_2\ \hat{y}_{\mu+2,2},$$

where $y_{\mu+1,1}$, $y_{\mu+1,2}$, $y_{\mu+2}$, $\hat{y}_{\mu+1}$, $\hat{y}_{\mu+2,1}$ and $\hat{y}_{\mu+2,2}$ are new pairwise distinct variables.

As $\tau(\tilde{x}_1)$ and $\tau(\tilde{x}_2)$ can be any $\mathtt{a}_i$ with $i \in \{1, \dots, n\}$, all $\sigma \in H^+$ of κ-NE-good form, where $\sigma(\alpha_1)$ or $\sigma(\alpha_2)$ contains a factor $\mathtt{a}_i$ with $i \in \{1, \dots, n\}$, satisfies $\pi_{\mu+1}$ and/or $\pi_{\mu+2}$. This can be proved in the same way as Lemma 4.5.

The changes in the proof of Lemma 4.6 are similar to the changes in the erasing case. The factors $\mathtt{a}_1 \dots \mathtt{a}_n$ in the terminal-string t of α do not affect the proof, as the variables $\tilde{x}_1$ and $\tilde{x}_2$ can be mapped to the whole factor $\mathtt{a}_1 \dots \mathtt{a}_n$, if the corresponding $\gamma_{\mu+1}$, $\delta_{\mu+1}$ and $\gamma_{\mu+2}$, $\delta_{\mu+2}$ are not mapped to $\sigma(0\alpha_1 0)$, $\sigma(0\alpha_2 0)$.

The additional parts in the definition of τ' with $k \in \{1, 2\}$ in the first half of the proof of Lemma 4.6, where π_i is the satisfied predicate, would be

1. $\tau'(\tilde{x}_k) := \mathtt{a}_1 \dots \mathtt{a}_n$ if $i \neq \mu + k$,
2. $\tau'(\tilde{y}_k) := \begin{cases} 0\,\mathtt{a}_1 \dots \mathtt{a}_{j-1} & \text{if } i = \mu + k \text{ and } \tau(\tilde{x}_k) = \mathtt{a}_j \\ 0 & \text{else ,} \end{cases}$
3. $\tau'(\tilde{z}_k) := \begin{cases} \mathtt{a}_{j+1} \dots \mathtt{a}_n\,0 & \text{if } i = \mu + k \text{ and } \tau(\tilde{x}_k) = \mathtt{a}_j \\ 0 & \text{else .} \end{cases}$

In the second half we can presume that $\sigma(0\alpha_1 0)$ and $\sigma(0\alpha_2 0)$ do not contain $\mathtt{a}_i$ with $i \in \{1, \dots, n\}$ as a factor, as else $\pi_{\mu+1}$ or $\pi_{\mu+2}$ would be satisfied. Also we get four equations instead of only two, but the two additional equations do not affect the proof.

Nothing after Lemma 4.5 has to be changed in Section 4.5 to transfer the results to Σ with $|\Sigma| \geq 3$.

Because of the additional suffix and the two new predicates, the pattern $\tilde{\beta}$ has 22 variables more than the pattern β from the original NE-construction.

5.2. *Inclusion in* $\mathrm{ePAT}_{\mathrm{tf},\Sigma}$ *or* $\mathrm{nePAT}_{\mathrm{tf},\Sigma}$

Both constructions can also be adapted to use terminal-free patterns β:

Theorem 5.2. *Let $|\Sigma| = 2$. The following problems are undecidable:*

1. *The inclusion problem of* $\mathrm{ePAT}_{2,\Sigma}$ *in* $\mathrm{ePAT}_{\mathrm{tf},\Sigma}$,
2. *the inclusion problem of* $\mathrm{nePAT}_{2,\Sigma}$ *in* $\mathrm{nePAT}_{\mathrm{tf},\Sigma}$.

We explain these modifications in Sections 5.2.2 and 5.2.1.

Note that the number of different variables in the patterns from $\mathrm{Pat}_{\mathrm{tf}}$ remains bounded. Although one might expect that this result could be modified to show that the open inclusion problem for $\mathrm{nePAT}_{\mathrm{tf},\Sigma}$ is undecidable, we consider this doubtful, as the modified NE-construction relies heavily on the terminal symbols in α. Furthermore, although it is considerably easier to modify the NE-construction, the fact that the inclusion problem for $\mathrm{ePAT}_{\mathrm{tf},\Sigma}$ is decidable casts further doubt on that expectation. As in Section 5.1, all other results that are based on one of the two constructions can be adapted as well.

5.2.1. Construction for Inclusion in $\mathrm{ePAT}_{\mathrm{tf},\Sigma}$

As in the nonerasing case, all terminals # and 0 are changed into the new variables c and d. We extend each of the patterns α and β with an additional prefix, which has to be extremely long compared to the rest of the pattern.

As now the new variables could be mapped to the empty word and $\beta \in \mathrm{Pat}_{\mathrm{tf}}$, $L_{\mathrm{E},\Sigma}(\beta)$ would be Σ^*, if any variable in β occurs only once. To avoid this problem all patterns $\hat{\beta}_i$ with $i \in \{1, \dots, \mu\}$ shall be redefined, as the only variables, which occurred only once, where in these patterns. Analogously, the middle section of α shall be redefined.

The new pattern α is defined by

$$\alpha := w\ v\ v\ \#^4\ v\ \alpha_1\ v\ \alpha_1\ v\ \alpha_2\ v\ \alpha_2\ v\ \#^4\ v\ u\ v,$$

with $u{:=}0\#^2 0$ and $v{:=}0\#^3 0$ (as before), and the new additional prefix

$$w := \#^\nu 0 \#^{\nu-1} 0 \#^{\nu-2} 0 \dots \#^3 0 \#^2 0 \# 0\ 0,$$

where $\nu \in \mathbb{N}_1$ shall be specified after the definition of β. Note that in the middle part of α, $\alpha_1 v$ and $\alpha_2 v$ were doubled.

Now we define the new pattern β by

$$\beta := \beta'\ (x_1)^2 \dots (x_\mu)^2\ c^4\ \hat{\beta}_1 \dots \hat{\beta}_\mu\ c^4\ \ddot{\beta}_1 \dots \ddot{\beta}_\mu,$$

where for all $i \in \{1, \dots, \mu\}$

$$\begin{aligned} \hat{\beta}_i &:= x_i\ \gamma_i\ x_i\ \gamma_i\ x_i\ \delta_i\ x_i\ \delta_i\ x_i, \\ \ddot{\beta}_i &:= x_i\ \eta_i\ x_i \end{aligned}$$

and the new additional prefix

$$\beta' := c^\nu d c^{\nu-1} d c^{\nu-2} d \dots c^3 d c^2 d c d\ d.$$

The number ν has to be at least $|\beta| - |\beta'| + 6$. We do not exactly define this number, as it is sufficient if ν is large enough.

This is used to restrict the images of the new variables. As ν affects $|\beta|$ and $|\beta'|$ by the same amount, $|\beta| - |\beta'| + 6$ is independent of ν.

As in the nonerasing case, the new prefixes are equal if c is mapped to # and d is mapped to 0. In the definition of the original E-construction, only variables from the subpatterns γ_i and δ_i with $i \in \{1, \dots, \mu\}$ can occur only once in β. Now all variables are occur at least twice in β, as all γ_i and δ_i were doubled in $\hat{\beta}_i$. Note that β' and the two factors c^4 in the definition of β are all occurrences of the new variables, as all other parts of β were terminal-free in the original E-construction.

Similar to the nonerasing case we can formulate a lemma, which shows that σ has to be of κ-E-bad-form, if the new variables are not mapped to the corresponding terminals. But to prove this, we need that every element of $\mathrm{var}(\beta)$ occurs at least twice in β.

Lemma 5.3. *Let $\sigma \in H$, and let $\tau : (\mathrm{var}(\beta))^* \to \Sigma^*$ be a substitution with $\tau(\beta) = \sigma(\alpha)$. If $\tau(c) \neq \#$ or $\tau(d) \neq 0$, then $\sigma(\alpha_1)$ or $\sigma(\alpha_2)$ contains the factor $\#^3$.*

Proof. As α begins with a long sequence of terminals, the prefix of $\sigma(\alpha)$ is equal to the prefix of α in this proof.

Remember the prefixes w and β' defined above and that $\tau(\beta) = \sigma(\alpha)$ holds.
Case 1: $\tau(c) \notin \{\lambda, \#\}$. Now $\tau(c^\nu)$ is not a prefix of $\sigma(\alpha)$, as α begins with $\#^\nu 0$ and this factor occurs only once in $\sigma(\alpha)$, if $\sigma(\alpha_1)$ and $\sigma(\alpha_2)$ do not contain $\#^3$ as a factor. Thus, $\tau(c) \in \{\lambda, \#\}$
Case 2: $\tau(c) = \lambda$. Then

$$\begin{aligned}\tau(\beta') &= \tau(c^\nu d c^{\nu-1} d c^{\nu-2} d \ldots c^3 d c^2 d c d d)\\ &= \tau(d^{\nu+1}),\end{aligned}$$

and thus, $\tau(d) = \lambda$, as α has the prefix $\#^\nu 0$ and this factor occurs only once in $\sigma(\alpha)$, if $\sigma(\alpha_1)$ and $\sigma(\alpha_2)$ do not contain $\#^3$ as a factor.

If ν is larger than $|\beta| - |\beta'| + 5$, then there has to exist a variable $x \in var(\beta)$ with

$$\tau(x) = w_1 \ 0 \ \#^j \ 0 \ w_2,$$

$w_1, w_2 \in \Sigma^*$ and $j \in \{5, \ldots, \nu - 1\}$. But a variable with such an image cannot exist, if $\sigma(\alpha_1)$ and $\sigma(\alpha_2)$ do not contain $\#^3$ as a factor, as each factor $0\#^j 0$ with $j \in \{5, \ldots, \nu - 1\}$ occurs exactly once in $\sigma(\alpha)$, but each variable occurs at least twice in β.

Thus, $\tau(c) \neq \lambda$, if ν is big enough.
Case 3: $\tau(c) = \#$. Then $\tau(d) = 0$, as else $\tau(c^\nu d c^{\nu-1})$ would not be prefix of $\sigma(\alpha)$ if $\sigma(\alpha_1)$ and $\sigma(\alpha_2)$ did not contain $\#^3$ as a factor.

If ν is large enough, then only Case 3 is possible, and thus, $\tau(c) = \#$ and $\tau(d) = 0$. □

With Lemma 5.3, we can formulate a new version of Lemma 4.1.

Lemma 5.4. *Let $\tau : (\mathrm{var}(\beta))^* \to \Sigma^*$ be a substitution. A substitution $\sigma \in H$ is of κ-E-bad form if and only if $\tau(c) \neq \#$, $\tau(d) \neq 0$ or σ satisfies π_1 or π_2.*

Proof. The proof is identical to the proof of Lemma 5.6, mutatis mutandis. □

The doubled parts in α and β barely affect the proof of Lemma 4.2. In the first half of the proof we extend the definition of the morphism τ' by $\tau'(c){:=}\#$ and $\tau'(d){:=}0$. This change leads to an almost identical proof. In the second half, as this direction is already shown for all $\sigma \in H$ of κ-E-bad form, $\tau(c) = \#$ and $\tau(d) = 0$ follow. The equation in the proof can again be decomposed, as now the prefixes of $\tau(\beta)$ and $\sigma(\alpha)$ are equal. The doubled parts do not affect the reasoning. As Lemma 4.2 can also be proved for the special case $\beta \in \mathrm{Pat}_{\mathrm{tf}}$, all results can be transferred.

Note that these changes increase $|\mathrm{var}(\beta)|$ by 2. These modifications can be combined with the results of Section 5.1. This adds another 22 variables to $\mathrm{var}(\beta)$, and the additional suffix used in Section 5.1.1 has to be doubled as well.

5.2.2. Construction for Inclusion in $\text{nePAT}_{\text{tf},\Sigma}$

With the additional prefixes we shall get for all $\sigma \in H^+$ of κ-NE-good form and nonerasing substitutions $\tau : (\text{var}(\beta))^* \to \Sigma^*$ the limitations $\tau(c) = \#$ and $\tau(d) = 0$, if $\sigma(\alpha) = \tau(\beta)$. As under this condition the image of the new prefix of β shall be equal to the new prefix of α, all results of the prior sections shall be transfered to the case $\beta \in \text{Pat}_{\text{tf}}$.

The new pattern β is defined by

$$\beta{:=}(c^3\ d)^2\ a\ b\ c^5\ a\ x_1 \dots x_\mu\ b\ c^5\ r_1\ \hat{\beta}_1 \dots r_\mu\ \hat{\beta}_\mu\ r_{\mu+1},$$

where $(c^3 d)^2$ is the new additional prefix. Note that the $\#^5$ in the middle part of β has changed into c^5. The patterns $\hat{\beta}_1$ to $\hat{\beta}_\mu$ and β' are defined as in Section 4.5, but the terminals are changed into the corresponding new variables c and d. Note, these are also the only changes in the patterns γ_i and δ_i with $i \in \{1, \dots, \mu\}$.

The pattern α is defined by

$$\alpha{:=}(\#^3\ 0)^2\ 0^{\mu+1}\ \#^5\ 0^\mu \# 0^\mu\ \#^5\ t\ v\ 0\alpha_1 0\ v\ 0\alpha_2 0\ v\ t,$$

where $(\#^3 0)^2$ is the new additional prefix and the rest of the pattern did not change in contrast to Section 4.5. Even the terminal-string t did not change, as the nonerasing substitution $\psi : (\text{var}(\hat{\beta}_1 \dots \hat{\beta}_\mu))^* \to \Sigma^*$ now is defined by $\psi(c) = \#$ and $\psi(x) = 0$ for all $x \in \text{var}(\hat{\beta}_1 \dots \hat{\beta}_\mu) \backslash c$. Thus, Lemma 4.4 still holds.

Now we can show how, by the additional prefixes, $\sigma(0\alpha_1 0)$ and $\sigma(0\alpha_2 0)$ are restricted, if the new variables are not mapped to the corresponding terminals.

Lemma 5.5. *Let $\sigma \in H^+$ and $\tau : (\text{var}(\beta))^* \to \Sigma^*$ be a nonerasing substitution with $\tau(\beta) = \sigma(\alpha)$. If $\tau(c) \neq \#$ or $\tau(d) \neq 0$, then $\sigma(0\alpha_1 0)$ or $\sigma(0\alpha_2 0)$ contains the factor $\#^3$.*

Proof. The pattern α contains the factor $\#^3$ less than 17 times. (Proof: All parts, where the factor $\#^3$ occurs or can occur: $\#^3$ occurs two times in the new prefix; the factor $\#^5$ occurs twice in α; each of the three factors v contains once the factor $\#^4$; each of the two factors t contains once the factor $\#^\kappa$.) Outside of α_1 and α_2 no variable occurs in α. The variables c and d both occur more than 17 times in β. (Proof: The variable c occurs six times in the new prefix $(\#^3 0)^2$, ten times in the two factors c^5 and κ times in γ_1. The variable d occurs 21 times in γ_4 to γ_9.) Thus, $\sigma(0\alpha_1 0)$ or $\sigma(0\alpha_2 0)$ has to contain the factor $\#^3$ if $\tau(c)$ or $\tau(d)$ contains the factor $\#^3$.

As α begins with a long sequence of terminals, the prefix of $\sigma(\alpha)$ is equal to the prefix of α in this proof.

Remember, $\tau(\beta) = \sigma(\alpha)$ holds and the substitutions are nonerasing.

At first we examine $\tau(c)$. As $\#^3$ is prefix of α and c is prefix of β, $\tau(c) \in \{\#, \#^2\}$, if $\tau(c)$ does not contain $\#^3$ as a factor. But as $\#^3 0$ is prefix of α and c^2 is prefix of β, $\tau(c) \neq \#^2$. Thus, $\tau(c) = \#$ if $\sigma(0\alpha_1 0)$ and $\sigma(0\alpha_2 0)$ do not contain the factor $\#^3$.

Note that c^3dc^3 is a prefix of β. As $\tau(c) = \#$, we get $\tau(c^3dc^3) = \#^3\tau(d)\#^3$. Furthermore, $\#^30\#^3$ is a prefix of α, and thus, $\tau(d) \in \{0, 0\#, 0\#^2\}$ if $\tau(d)$ does not contain $\#^3$ as a factor. But $\tau(d) \notin \{0\#, 0\#^2\}$, as $\#^30\#^30$ is prefix of α and if $\tau(d) \in \{0\#, 0\#^2\}$ then $\tau(c^3dc^3) \in \{\#^30\#^4, \#^30\#^5\}$. Thus, $\tau(d) = 0$ if $\sigma(0\alpha_10)$ and $\sigma(0\alpha_20)$ do not contain the factor $\#^3$. □

If $\tau(c) = \#$ and $\tau(d) = 0$, then the only difference of $\tau(\beta)$ in contrast to Section 4.5 is the additional prefix, which then is equal to the additional prefix of α, and thus, do not affect the rest of the patterns.

Now we can formulate a new version of Lemma 4.5, which includes the results of Lemma 5.5.

Lemma 5.6. *Be $\tau : (\mathrm{var}(\beta))^* \to \Sigma^*$ a nonerasing substitution. A nonerasing substitution $\sigma \in H^+$ is of κ-NE-bad form if and only if $\tau(c) \neq \#$, $\tau(d) \neq 0$ or σ satisfies π_1 or π_2.*

Proof. If $\tau(c) \neq \#$ or $\tau(d) \neq 0$, then $\sigma(0\alpha_10)$ or $\sigma(0\alpha_20)$ contains the factor $\#^3$, because of Lemma 5.5, and thus, σ is of κ-NE-bad form.

If $\tau(c) = \#$ and $\tau(d) = 0$, then the predicates π_1 and π_2 are equal to the predicates π_1 and π_2 used in Section 4.5, and thus, Lemma 4.5 holds. □

Now all results can be transfered to the special case $\beta \in \mathrm{Pat}_{\mathrm{tf}}$ in the nonerasing case, as – by Lemmas 5.5 and 5.6 – Lemma 4.6 can be transfered to the special case $\beta \in \mathrm{Pat}_{\mathrm{tf}}$. Roughly speaking, the replacement of # and 0 by c and d enlarges the language $L_{\mathrm{NE},\Sigma}(\beta)$, but none of the additional elements in $L_{\mathrm{NE},\Sigma}(\beta)$ is also in $L_{\mathrm{NE},\Sigma}(\alpha)$.

The needed changes to transfer Lemma 4.6 are marginal. In the first half of the proof in the definition of τ' the images of c and d have to be defined separately by $\tau'(c) = \#$ and $\tau'(d) = 0$. In the second half of the proof follows immediately $\tau(c) = \#$ and $\tau(d) = 0$, as in this direction, $\sigma \in H$ has to be of κ-NE-good form.

The changes enlarged $|\mathrm{var}(\beta)|$ only by two. The results of Section 5.1 can also be transfered to the special case $\beta \in \mathrm{Pat}_{\mathrm{tf}}$, which would increase $|\mathrm{var}(\beta)|$ in addition by 22.

References

[1] D. Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Sciences*, 21:46–62, 1980.

[2] J. Bremer and D.D. Freydenberger. Inclusion problems for patterns with a bounded number of variables. In *Proc. DLT 2010*, volume 6224 of *LNCS*, pages 100–111, 2010.

[3] C. Câmpeanu, K. Salomaa, and S. Yu. A formal study of practical regular expressions. *Int. J. Found. Comput. Sci.*, 14:1007–1018, 2003.

[4] C. Choffrut and J. Karhumäki. Combinatorics of words. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 1, chapter 6, pages 329–438. Springer, 1997.

[5] V.G. Durnev. Undecidability of the positive $\forall\exists^3$-theory of a free semigroup. *Siberian Math. J.*, 36(5):917–929, 1995.

[6] A. Ehrenfeucht and G. Rozenberg. Finding a homomorphism between two words is NP-complete. *Inform. Process. Lett.*, 9:86–88, 1979.

[7] D.D. Freydenberger and D. Reidenbach. Bad news on decision problems for patterns. *Inform. and Comput.*, 208(1):83–96, 2010.

[8] O.H. Ibarra, T.C. Pong, and S.M. Sohn. A note on parsing pattern languages. *Pattern Recognit. Lett.*, 16(2):179–182, 1995.

[9] T. Jiang, E. Kinber, A. Salomaa, K. Salomaa, and S. Yu. Pattern languages with and without erasing. *Int. J. Comput. Math.*, 50:147–163, 1994.

[10] T. Jiang, A. Salomaa, K. Salomaa, and S. Yu. Decision problems for patterns. *J. Comput. Syst. Sci.*, 50:53–63, 1995.

[11] J.C. Lagarias. The 3x+1 problem: An annotated bibliography (1963–1999), Aug 2009. `http://arxiv.org/abs/math/0309224`.

[12] J.C. Lagarias. The 3x+1 problem: An annotated bibliography, II (2000–2009), Aug 2009. `http://arxiv.org/abs/math/0608208`.

[13] M. Margenstern. Frontier between decidability and undecidability: a survey. *Theor. Comput. Sci.*, 231(2):217–251, 2000.

[14] V. Mitrana. Patterns and languages: An overview. *Grammars*, 2(2):149–173, 1999.

[15] T. Nagell, editor. *Selected mathematical papers of Axel Thue.* Universitetsforlaget, Oslo, 1977.

[16] T. Neary and D. Woods. Four small universal Turing machines. *Fundam. Inform.*, 91(1):123–144, 2009.

[17] Y.K. Ng and T. Shinohara. Developments from enquiries into the learnability of the pattern languages from positive data. *Theor. Comput. Sci.*, 397:150–165, 2008.

[18] E. Ohlebusch and E. Ukkonen. On the equivalence problem for E-pattern languages. *Theor. Comput. Sci.*, 186:231–248, 1997.

[19] D. Reidenbach. *The Ambiguity of Morphisms in Free Monoids and its Impact on Algorithmic Properties of Pattern Languages.* PhD thesis, Fachbereich Informatik, Technische Universität Kaiserslautern, 2006. Logos Verlag, Berlin.

[20] K. Salomaa. Patterns. In C. Martin-Vide, V. Mitrana, and G. Păun, editors, *Formal Languages and Applications*, number 148 in Studies in Fuzziness and Soft Computing, pages 367–379. Springer, 2004.

[21] T. Shinohara. Polynomial time inference of extended regular pattern languages. In *Proc. RIMS Symposia on Software Science and Engineering, Kyoto*, volume 147 of *Lecture Notes in Computer Science*, pages 115–127, 1982.

[22] A. Thue. Über unendliche Zeichenreihen. *Kra. Vidensk. Selsk. Skrifter. I Mat. Nat. Kl.*, 7, 1906. In German. Reprinted in [15].