The pseudopalindromic completion of regular languages

Szilárd Zsolt Fazekas

Akita University, Department of Computer Science and Engineering, 1-1 Tegata Gakuen-machi, Akita City, 010-8502, Japan

Florin Manea, Robert Mercaş*

Kiel University, Department of Computer Science, D-24098 Kiel, Germany

Kayoko Shikishima-Tsuji

Tenri University, 1050 Somanouchi Tenri, 632-8510, Japan

Abstract

Pseudopalindromes are words that are fixed points for some antimorphic involution. In this paper we discuss a newer word operation, that of pseudopalindromic completion, in which symbols are added to either side of the word such that the new obtained words are pseudopalindromes. This notion represents a particular type of hairpin completion, where the length of the hairpin is at most one. We give precise descriptions of regular languages that are closed under this operation and show that the regularity of the closure under the operation is decidable.

Keywords: Pseudopalindromes, Pseudopalindromic completion, Pseudopalindromic iterated completion, Regular languages, Algorithms, Decidability.
2000 MSC: 68Q45, 68Q42, 68Q15

---

*Corresponding author. Tel: +49 431 880-6531

Email addresses: `szilard.fazekas@gmail.com` (Szilárd Zsolt Fazekas), `flm@informatik.uni-kiel.de` (Florin Manea), `rgm@informatik.uni-kiel.de` (Robert Mercaş), `tsuji@sta.tenri-u.ac.jp` (Kayoko Shikishima-Tsuji)

1. Introduction

Palindromes are sequences which read the same starting from either end. Whenever the first half is the same as the complement of the second half read from right to left the sequence is called a pseudopalindrome. Besides their importance in combinatorial studies of strings, mirrored complementary sequences occur frequently in DNA and are often found at functionally interesting locations such as replication origins or operator sites. Already in the 1950's it was recognised that pseudopalindromic regions of DNA can exist in a cruciform structure with intrastrand base pairing of the self-complementary sequence, i. e., if a pseudopalindromic sequence occurs in a double strand, then pulling apart the two strands at the middle of the pseudopalindrome one can perform a "transfer-twist" in which each strand twists about itself, reducing the energy needed to separate the strands.

A similar phenomenon is when a single strand of DNA curls back on itself to become self-complementary, after which a polymerase chain reaction can extend the "shorter" end to generate a complete double strand, the result being a partial double helix with a bend in it. The structure is called a hairpin or stem-loop, and it is an important building block of many RNA secondary structures. Several operations on sequences were introduced which are either directly motivated by the biological phenomenon called stem-loop completion, or are very similar in nature to it. The mathematical hairpin concept introduced by Păun et al. in [1] refers to a word in which some suffix is the mirrored complement of a middle factor of the word. The hairpin completion operation, which extends such a word into a pseudopalindrome with a non-matching part in the middle was thoroughly investigated in [2, 3, 4, 5, 6, 7, 8, 9]. Most basic algorithmic questions about hairpin completion have been answered (see, e. g. Cheptea et al. [2] and Diekert et al. [3]) with a noteworthy exception: "given a word, can we decide whether the iterated application of the operation leads to a regular language?". There exist however particular cases of this concept, such as the bounded hairpin completion introduced by Ito et al. in [10], where also this latter problem was settled by Kopecki in [11].

Another operation related to our topic is the iterated palindromic closure which was first introduced by de Luca in the study of the Sturmian words in [12] and later generalised to pseudopalindromes by de Luca and De Luca in [13]. This operator allows one to construct words with infinitely many pseudopalindromic prefixes, called pseudostandard words. In this context the newly obtained words have length greater than the original ones, but are minimal among all pseudopalindromes that have the original word as prefix or suffix. On the same page, in [14] Mahalingam and Subramanian propose the study of some similar operation, that of pseudopalindromic completion of a word. The operation considers in this form all possible ways of extending the word into a pseudopalindrome, thus producing in one step of an application of the operation an infinite set from any starting word. It is worth noting that together with the notion of θ-inverse that refers to the shortest such completion and was introduced in the same paper, the latter concept represents a generalisation of the pseudopalindromic closure.

The operation studied here, is yet another type of pseudopalindromic completion. Although having the same name, it differs from the pseudopalindromic completion in [14] in that we require the word to have a pseudopalindromic prefix or suffix in order to be completed. In comparison with the (iterated) pseudopalindromic closure in [12] which considers the unique shortest word that completes the starting string into a pseudopalindrome, as we shall observe, here we take all possible extensions. However, this choice of restriction was not randomly picked, the subject of this work being closest in nature to the first presented operation. Since in the biological phenomenon serving as inspiration, the hairpin's length in the case of stable bindings is optimally limited (approximatively 4-8 base-pairs according to Slama-Schwok et al. in [15]) it is natural to consider completions with bounded middle part. As we will see our operation is in fact a rather restricted form of the hairpin completion (we do not allow for non-matching middles), and the questions asked are also a subset of problems considered for the initial operation.

After presenting the notions and results needed for our treatise, in Section 3 we state some simple one-step completion results. In Section 4 we gradually build the characterisation of regular languages which stay regular under the iterated application of pseudopalindromic completion. Section 5 is a collection of algorithmic results on this operation such as the membership problem for the iterated completion of a word and that of a language, the completion distance between two input words and decision methods regarding the preservation of regularity within iterated completion. We conclude our work with Section 6 where we present some further remarks regarding this operation.

2. Preliminaries

We assume the reader to be familiar with fundamental concepts from Formal Language Theory, such as the classes of the Chomsky hierarchy, finite automaton, regular expressions (e. g., see the textbook by Harrison [16]), as well as fundamental concepts from combinatorics on words (e. g., see the Lothaire textbook [17]).

Let Σ be a non-empty finite *alphabet* with *letters* as elements. A sequence of letters constitute a *word* $w \in \Sigma^*$ and we denote by ε the *empty word*.

The *length* of a finite word w is the number of not necessarily distinct symbols it consists of and is denoted by $|w|$. The ith symbol we write as $w[i]$ and use the notation $w[i..j]$ to refer to the part of a word starting at the ith and ending at the jth position.

Words together with the operation of concatenation form a free monoid, which is usually denoted by Σ^* for an alphabet Σ. Any subset of Σ^* is called a language. By Σ^+ we denote the set of non-empty words over Σ.

Repeated concatenation of a word w with itself is denoted by w^i for integers $i \geq 0$ with i representing a *power*. Furthermore, w is said to be *primitive* if there exists no non-empty word u such that $w = u^j$ for some integer $j > 1$. Otherwise, we call w a *repetition* and the smallest such u its *root* (note that in this case the word u is primitive).

A word u is a *factor* of w if there exist integers i, j with $1 \leq i, j \leq |w|$ such that $u = w[i..j]$. We say that u is a *prefix* of w whenever we can fix $i = 1$ and denote this by $u \leq_p w$. If $j < |w|$, then the prefix is called *proper*. *Suffixes* are the corresponding concept reading from the back of the word to the front. A word w has a positive integer k as a *period* if for all i, j such that $i \equiv j (\mathrm{mod} k)$ we have $w[i] = w[j]$, whenever both $w[i]$ and $w[j]$ are defined.

A central concept to this work is *palindromicity* in the general sense. For a word $w \in \Sigma^*$ we denote by w^R its *reversal*, that is $w[|w|]w[|w| - 1] \dots w[1]$. If $w = w^R$, the word is called a *palindrome*. Let $\mathcal{P}al(L)$ be the set of all palindromes of a language $L \subseteq \Sigma^*$ and $\mathcal{P}al_\Sigma$ be the language of all palindromes over Σ (when the alphabet is clear from the context we shall drop the Σ and denote this set by $\mathcal{P}al$).

We can generalise the previous definition by using an arbitrary antimorphic involution instead of the reversal. To this end, let θ be an *antimorphic involution*, i. e., $\theta : \Sigma^* \to \Sigma^*$ is a function such that $\theta(\theta(a)) = a$ for all $a \in \Sigma$ and $\theta(uv) = \theta(v)\theta(u)$ for all $u, v \in \Sigma^+$. Then, w is a $\theta-$*pseudopalindrome* if $w = \theta(w)$. To make notation simpler, we write $\overline{u}$ for $\theta(u)$ whenever θ

is understood from the context. As an example, for the antimorphism θ with $\theta(a) = b$ and $\theta(b) = a$, the word $aabb = \theta(aabb) = \theta(b)\theta(b)\theta(a)\theta(a)$ is a pseudopalindrome. The language of all $\theta-$pseudopalindromes, when the alphabet Σ and θ are fixed, is denoted by $\mathcal{PP}al$.

It is immediate to show that this is a linear context-free language, just like $\mathcal{P}al$. Moreover, the primitive root of every ($\theta-$pseudo)palindrome is a ($\theta-$pseudo)palindrome itself.

Note that all palindromes $p = aqa^R$ with q a (possibly empty) palindrome have the *trivial* palindromic prefixes (suffixes) ϵ, a and aqa^R. Hence, by a palindrome having a *non-trivial* palindromic prefix (suffix), we mean it has a proper prefix (suffix) of length at least two which is a palindrome. This notion can be canonically extended to θ-pseudopalindromes. Throughout this paper, unless explicitly stated otherwise, whenever we say *pseudopalindromic prefix* (*suffix*), we refer to a *non-trivial pseudopalindromic prefix* (*suffix*).

Definition 1. Let θ be an antimorphic involution over an alphabet Σ and $w \in \Sigma^*$. A word $w' \in \Sigma^*$ is in the *right* (respectively, *left*) $\theta-$*completion* of w if $w = uv$ where v (respectively, u) is a θ-palindrome with $1 < |v| < |w|$ (respectively, $1 < |u| < |w|$) and $w' = uv\theta(u)$ (respectively, $w' = \theta(v)uv$). A word w' is in the $\theta-$completion of w if it is either in the right or left $\theta-$completion of w; this is denoted by $w \ltimes w'$. The reflexive and transitive closure of $\ltimes$, denoted by $\ltimes^*$, is called the *iterated* $\theta-$*completion*; more precisely, for two words w and w' we write $w \ltimes^* w'$ if $w = w'$ or there exists an integer $n > 1$ and the words $v_1, v_2, \ldots, v_n$ with $v_1 = w$, $v_n = w'$ and $v_i \ltimes v_{i+1}$ for $1 \leq i \leq n-1$.

In the following we assume Σ to be a fixed alphabet, such that all the words and languages that we work with are over Σ, and θ to be a fixed antimorphic involution over this alphabet. Therefore, we just say completion instead of θ-completion and pseudopalindrome instead of θ-pseudopalindrome.

Definition 2. For a language L, let $L^{\ltimes_0} = L$ and for $n > 0$ let $L^{\ltimes_n} = \{w \mid \text{there exists } u \in L^{\ltimes_{n-1}} \text{ such that } u \ltimes w\}$ be the pseudopalindromic completion of $L^{\ltimes_{n-1}}$. Moreover, we define $L^{\ltimes_*} = \bigcup_{n\geq 0} L^{\ltimes_n}$, the iterated pseudopalindromic completion of L.

To see an example, consider the antimorphic involution θ to be just the reverse function, e. g., $\theta(w) = w^R$, and the language $L = \{aabb, abb\}$. Then $aabb \ltimes \{aabbaa, bbaabb\}$ since $aabb$ has aa and bb as a palindromic prefix and suffix, respectively, while $abb \ltimes abba$.

For convenience, we define $L^{\ltimes_{\leq n}}$ to be the union of all completions up to the nth iteration of the language L, e. g., $L^{\ltimes_{\leq n}} = \bigcup_{i=0}^{n} L^{\ltimes_i}$. Moreover, for simplicity, let $w^{\ltimes_n}$ denote $\{w\}^{\ltimes_n}$ and $w^{\ltimes_{\leq n}}$ denote $\{w\}^{\ltimes_{\leq n}}$, i. e., the nth completion of the singleton language $\{w\}$ and the union of its ith completions for $0 \leq i \leq n$, respectively.

The following lemma is used frequently in our proofs:

Lemma 3. *For every regular language L there exists a positive integer k_L such that every word $w \in L$ longer than k_L has a factorisation $w = w_1 w_2 w_3$ with $w_2 \neq \epsilon$, $|w_1 w_2| \leq k_L$ and $w_1(w_2)^* w_3 \subseteq L$.*

Finally let us recall some well known combinatorial results:

Theorem 4 (Fine and Wilf)**.** *If two non-empty words u^i and v^j share a prefix of length $|u| + |v| - \gcd(|u|, |v|)$, then there exists a word r of length $\gcd(|u|, |v|)$ such that $u, v \in \{r\}^+$.*

Proposition 5. *If $uv = vw$ with $u, w \in \Sigma^+$ and $v \in \Sigma^*$, then $v = (xy)^k x$, $u = xy$, $w = yx$ for some words $x \in \Sigma^*$ and $y \in \Sigma^+$, and some integer $k \geq 0$.*

3. Pseudopalindromic Regular Languages

We begin our exposure with a simple observation. The pseudopalindromic completion of a word is always a finite set, as a word (of finite length) has only finitely many pseudopalindromic prefixes or suffixes.

In order to see that the class of regular languages is not closed under pseudopalindromic completion, consider the language $L = aa^+\overline{a}$. After one pseudopalindromic completion step we get $L^{\ltimes_1} = \{a^n \overline{a}^n \mid n \geq 2\}$, which is a non-regular context-free language. Moreover, no word in $L^{\ltimes_1}$ has a pseudopalindromic prefix or suffix, hence $L^{\ltimes_*} = L^{\ltimes_{\leq 1}}$.

We now look at the iterated pseudopalindromic completion. As an example consider the word $w = a\overline{a}ba\overline{a}\overline{b}$. The iterated completion of w is the infinite regular language $\{w\} \cup a\overline{a}(ba\overline{a}\overline{b}a\overline{a})^+ \cup ba\overline{a}\overline{b}(a\overline{a}ba\overline{a}\overline{b})^+$. In fact, the iterated completion of a word may be even more complex. For instance, if the starting word is $a^3ba^4ba^3$, the result of the iterated completion is not context-free (we will see further details regarding this result in the proof of Proposition 11).

Lemma 6. *For a word w, the language $w^{\ltimes_*}$ is infinite if and only if w has both pseudopalindromic prefixes and suffixes. Furthermore, in this case $w^{\ltimes_{\leq i}} \subsetneq w^{\ltimes_{\leq i+1}}$ for all integers $i \geq 1$.*

Proof. If w has both pseudopalindromic prefixes and suffixes, then after every completion, the resulting word will have them, too, therefore $w^{\ltimes_*}$ is infinite. To prove the other direction, suppose without loss of generality that no prefix of w is a pseudopalindrome, while one of its suffixes is a pseudopalindrome denoted by u with $|u| > 1$ (the other case is symmetric). Then, in one step the word $vu\overline{v}$ is reached via the operation, for the factorization $w = vu$. Since w had a pseudopalindrome only at one end, all of the prefixes and suffixes of length at most $|vu|$ of $vu\overline{v}$ are not pseudopalindromes. Now assume there is a factorisation $\overline{v} = xy$ with $1 \leq |x| < |v|$, such that vux is a pseudopalindrome (once more, the case $v = yx$ is symmetric).

If $|x| \leq \frac{|v|}{2}$, then not only is $\overline{x}ux$ a pseudopalindromic suffix of vux, but also $xu\overline{x}$ is a prefix of vu, contradicting our initial assumption, that w had no pseudopalindromic prefixes.

If $|x| > \frac{|v|}{2}$, then, because $\overline{x}ux$ is both a prefix and a suffix of vux, i. e., it is a border of vux which is longer than half of $|vux|$, it must be that vux has a border shorter than its half. Any border of a pseudopalindrome is a pseudopalindrome itself, hence vu starts with a pseudopalindrome, which is a contradiction. The second statement follows directly from the fact that the length increases with each iteration. □

Lemma 7. *For any pseudopalindrome w, the set of words obtained by right pseudopalindromic completion from w equals the set of words obtained by left pseudopalindromic completion from it.*

Proof. For a word w to have a right completion, it needs to have a decomposition $w = uvx\overline{v}$, where $x \in \Sigma \cup \{\epsilon\}$ with $x = \overline{x}$ and $v \neq \epsilon$. Therefore, $uvx\overline{v} \ltimes uvx\overline{v}\,\overline{u}$. Because the starting word w is a pseudopalindrome, we have $uvx\overline{v} = \overline{uvx\overline{v}} = v\overline{x}\,\overline{v}\,\overline{u}$, and a left completion gives us $uv\overline{x}\,\overline{v}\,\overline{u}$. Since $|x| \leq 1$ and according to our initial assumption $x = \overline{x}$, the conclusion follows. □

We remind the reader that a language is called pseudopalindromic if all of its elements represent pseudopalindromes. Hence, whenever considering several completion steps for some pseudopalindromic language L, it is enough to consider either the right or the left completions. We start by recalling the following result:

Lemma 8 (Kari and Mahalingam [18])**.** *For two words $u, v \in \Sigma^+$ such that uv is a pseudopalindrome, the following hold:*

- *$uv = \overline{v}u$ if and only if $u = \overline{u}$;*

- $v\overline{u} = uv$ *if and only if* $v = \overline{v}$.

Similar to the palindromic languages characterisation given by Horváth et al. in [19] we have the following characterisation:

Theorem 9. *A regular language* $L \subseteq \Sigma^*$ *is pseudopalindromic, if and only if it is a union of finitely many languages of the form*

$$L_p = \{p\} \text{ or } L_{r,s,q} = qr(sr)^*\overline{q}$$

where p*,* r *and* s *are pseudopalindromes, and* q *is an arbitrary word.*

Proof. Since L is a regular language, according to Lemma 3 there exists a constant k_L such that for any word $w \in L$ longer than k_L we have a factorisation $w = uvz$ with $0 < |uv| \leq k_L$ and $v \neq \epsilon$, such that $uv^iz \in L$ for any integer $i \geq 0$. Note that since k_L is fixed, the number of words that do not fulfil this property is finite, and given that they are all pseudopalindromes they will be in fact included in the set L_p of our statement.

The two cases being symmetric, we assume $|u| \leq |z|$. According to the same Lemma 3, since L is a pseudopalindromic language, it follows that for any integer $i \geq 0$ we have that $uv^iz \in L$ is a pseudopalindrome, thus, for a factorisation $z = x\overline{u}$ with $x \in \Sigma^*$, we have that v^ix is a pseudopalindrome. Since i can be arbitrarily large, there exists an integer j with $i > j \geq 0$ such that $x = v_1\overline{v}^j$, where $\overline{v} = v_2v_1$, and $uv^iz, uv^{i+1}z \in L$. However in this case we also get that v is a pseudopalindrome, and thus v_1 and v_2 are pseudopalindromes as well.

We conclude that our original word w can be written as $uv_1(v_2v_1)^{j+1}\overline{u}$. Following the description of Lemma 3 a similar decomposition exists for all words longer than k_L. Since all parts of the factorisation, u, v_1 and v_2 are shorter than k_L, the existence of finitely many such triplets, and therefore our results, are concluded. □

4. Iterated Pseudopalindromic Completion

Following our definition of pseudopalindromic completion, without loss of generality, we assume that all languages investigated in the case of iterated completion have only words longer than two (according to Definition 1 no word of length at most two can be extended by our operation, and their number is bounded by the square of the size of the alphabet).

First, the case of pseudopalindromic completion on unary alphabets is not difficult; even for arbitrary unary languages the iterated pseudopalindromic completion is regular:

Proposition 10. *The class of unary regular languages is closed under pseudopalindromic completion. Furthermore, the iterated pseudopalindromic completion of any unary language is regular.*

Proof. We know that all unary regular languages are expressed as a finite union of sets of the form $L = a^k(a^n)^*$ where k, n are some non-negative integers. Since for unary words to be pseudopalindromes we have $\overline{a} = a$, one step pseudopalindromic completion of a word a^r, for some integer $r > 1$, gives the language $\{a^\ell \mid r < \ell < 2r - 1\}$. Furthermore, when $n = 0$ we have that L is a singleton and thus its completion is regular.

Let us now consider the case when the size of L is infinite. Since in this case $n > 0$, for any non-negative integer $m > 2$, we have $2|a^k(a^n)^m| - 2 \geq |a^k(a^n)^{m+1}|$. Therefore, the one step completion of L consists of the union of a finite set of words together with the set of all words of the form a^r, with $r > k + 2n$, hence it is regular.

For the second statement the conclusion is trivial since the iterated completion of any unary language gives us the regular language described by the regular expression $a^j a^*$, where $j > 1$ denotes the length of the shortest word from the initial language. □

Next let us investigate what happens in the singleton languages case.

Proposition 11. *The class of iterated pseudopalindromic completion of singleton languages is incomparable with the classes of regular and context-free languages.*

Proof. To show that regular languages are obtained take the word $a\overline{a}a$. It is not difficult to check that the language obtained is $\{a\overline{a}a\} \cup \{(a\overline{a})^n, (\overline{a}a)^n \mid n \geq 2\}$. Since all the languages above are regular, so is their union.

To see that we may also get non-regular languages and, in fact, even non-context-free languages, consider the word $u = a^3ba^4ba^3$ and let θ be just the reverse function. Assume that $w' = xyx^R \in u^{\ltimes_i}$ such that $w \ltimes w'$ with $w = xy \in u^{\ltimes_{i-1}}$, for $i > 0$. Observe that in this case it must be that y is either the pseudopalindromic suffix aa or $y \in (a^3b(a^4b)^+)^*a^3$. If in all iteration steps up to the ith, either aa or $a^3b(a^4b)^ja^3$, with $j \geq 0$, is chosen, then $w' \in a^3b(a^4b)^+a^3$ and, conversely, the only way to get a word of that form is by choosing the previously mentioned suffixes. After the first completion step in which the suffix a^3 is chosen, the resulting word will be $a^3b(a^4b)^n a^3(ba^4)^n ba^3$, for some $n > 0$. Furthermore, it is not difficult to see that from here, the only way to get a word in the language $a^3b(a^4b)^+a^3b(a^4b)^+a^3b(a^4b)^+a^3$ after further iterations is to choose as pseudopalindromic suffix $a^3(ba^4)^j ba^3$, where $0 \leq j \leq n$. Summing up the above, the language $a^3b(a^4b)^+a^3b(a^4b)^+a^3b(a^4b)^+a^3$ intersected with $u^{\ltimes *}$ gives the language

$$\{a^3b(a^4b)^n a^3b(a^4b)^m a^3b(a^4b)^n a^3 \mid 0 < n \leq m \leq 2n\},$$

where the relation $n \leq m \leq 2n$ comes from the fact that the resulting word must be a pseudopalindrome with precisely two distinct factors ba^3b. This language is easily shown not to be context-free by applying the well known Bar-Hillel pumping lemma for context-free languages. As the class of context-free languages is closed under intersection with regular languages, we get that $u^{\ltimes *}$ is not context-free. □

The following result represents a generalisation of Proposition 5:

Proposition 12 (Kari and Mahalingam [18])**.** *Let $u, v \in \Sigma^+$ be words such that $uv = \overline{v}u$. Then $u = x(yx)^n$ and $v = (yx)^m$ for some pseudopalindromes x, y and integers $m \geq 1$ and $n \geq 0$.*

Lemma 13. *If v is a pseudopalindromic prefix or suffix of some other pseudopalindrome u, there always exist pseudopalindromes $x \neq \epsilon$ and y, such that $u, v \in x(yx)^*$. Moreover, for two pseudopalindromes $u = p_1(q_1p_1)^{i_1}$ and $v = p_2(q_2p_2)^{i_2}$, where $i_1, i_2 > 2$, $|p_1|, |p_2| > 1$ and $2|v| > |u|$, if v is a prefix of u, then there exist pseudopalindromes p, q, such that $p_j(q_jp_j)^+ \subseteq p(qp)^+$ with $j \in \{1, 2\}$.*

Proof. For the first part, as a direct consequence of Lemma 8 we have that $u = vx = \overline{x}v$, thus following Proposition 12 the result is available.

For the second statement, according to our assumptions $p_2(q_2p_2)^{i_2} > \frac{|p_1(q_1p_1)^{i_1}|}{2}$. If $|p_2q_2| \geq |p_1q_1|$, then we can apply Theorem 4 and get that p_1q_1 and p_2q_2 have the same primitive root r. If $|p_2q_2| < |p_1q_1|$, then we have two cases. If $|p_2(q_2p_2)^{i_2}| \geq |(p_1q_1)^2|$, then the Fine and Wilf Theorem applies directly giving that p_1q_1 and p_2q_2 have the same primitive root r. From

$$|(p_1q_1)^2| > |p_2(q_2p_2)^{i_2}| > |p_1q_1p_1| + \frac{|q_1|}{2},$$

either $|p_2q_2| \leq |p_1| + \frac{|q_1|}{2}$, or $|(q_2p_2)^2| > |p_1q_1p_1|$, and hence,

$$|p_2(q_2p_2)^{i_2}| \geq |(q_2p_2)^2| + |p_2q_2p_2| > |p_1q_1p_1| + |p_2q_2p_2| > |p_1q_1| + |p_2q_2|.$$

In both cases, we apply Theorem 4 to the same end.

Therefore, there exist pseudopalindromes p, q, such that pq is primitive, and $p_1 = p(qp)^{m_1}$ and $q_1 = q(pq)^{n_1}$ for some $m_1, n_1 \geq 0$. Thus $p_1(q_1p_1)^+ \in p(qp)^+$. Since $v \leq_p u$ and both words are pseudopalindromes, $v \leq_s u$ and u ends in $p((qp)^{m_1+n_1+1})^{i_1}$. But u also ends in $p_2(q_2p_2)^2$, so by the above argument, $p_2(q_2p_2)^+ \subseteq p(qp)^+$. □

Proposition 14. *For all words of the form $w = up(qp)^n\overline{u}$, where p and q are pseudopalindromes and u is a suffix of pq, there exist pseudopalindromes p', q' such that $w = p'(q'p')^m$, where $n \leq m \leq n+2$.*

Proof. We distinguish three distinct cases based on the lengths of u and q.

If $|u| \leq \frac{|q|}{2}$, then $q = \overline{u}xu$ for some (possibly empty) pseudopalindrome x. Thus, w can be written as $up(\overline{u}xup)^n\overline{u} = up\overline{u}(x.up\overline{u})^n$. We assign $p' = up\overline{u}$ and $q' = x$, and we can conclude.

When $\frac{|q|}{2} < |u| \leq |q|$, we have that the prefix u and the suffix $\overline{u}$ overlap in q, i. e., $q = xyxyx$ and $u = xyx$, for some pseudopalindromes x and y. Thus, $w = xyxp(xyxyxp)^nxyx = x(yxpxyx)^{n+1}$, and we find that $p' = x$ and $q' = yxpxy$ satisfy our requirements.

Finally, if $|q| < |u|$, then $u = xq$ for some suffix x of p. Thus, $w = xqp(qp)^nq\overline{x} = xq(pq)^{n+1}\overline{x}$ with x a suffix of p, which brings us back to one of the previous cases (if the latter, the exponent increases by one yet again). □

Proposition 15. *Let $u_ip_i(q_ip_i)^{k_i}\overline{u_i}$, where p_i, q_i are pseudopalindromes and $k_i > 0$, be a sequence of pseudopalindromes for all $1 \leq i \leq n$ such that*

$$u_1p_1(q_1p_1)^{k_1}\overline{u_1} \ltimes u_2p_2(q_2p_2)^{k_2}\overline{u_2} \ltimes \ldots \ltimes u_np_n(q_np_n)^{k_n}\overline{u_n},$$

where $u_1 = u_n$, $p_1 = p_n$ and $q_1 = q_n$. Then there exist pseudopalindromes p, q and positive integers t_i with $1 \leq i \leq n$, such that $u_ip_i(q_ip_i)^{k_i}\overline{u_i} = p(qp)^{t_i}$.

Proof. First, we show that there exist pseudopalindromes p', q' and a word u', such that for each $1 \leq i \leq n$, we have $u_ip_i(q_ip_i)^{k_i}\overline{u_i} = u'p'(q'p')^{l_i}\overline{u'}$, for some exponent l_i. All words in the sequence are prefixes of $u_np_n(q_np_n)^{k_n}\overline{u_n} = u_1p_1(q_1p_1)^{k_n}\overline{u_1}$, hence each can be written as $u_ip_i(q_ip_i)^{k_i}\overline{u_i} = u_1p_1(q_1p_1)^{\ell_i}x_i$, for an exponent $\ell_i \geq k_1$ and a proper prefix x_i of q_1p_1 or of u_1.

Now, if p_1q_1 is primitive and the length of x_i is smaller than that of u_1, since $u_1p_1(q_1p_1)^{\ell_i}x_i = \overline{x_i}p_1(q_1p_1)\overline{u_1}$, by synchronisation we get a contradiction ($p_1q_1p_1$ will overlap with p_1q_1 with no exact matching). Otherwise, set $u' = u_1$, $p' = p_1$ and $q' = q_1$ and we are done.

If p_1q_1 is not primitive, then there exist pseudopalindromes p', q' with $p'q'$ primitive, such that $p_1 \in p'(q'p')^*$ and $q_1 \in q'(p'q')^*$. Furthermore, $u_1 = u'(p'q')^k$, for some $k \geq 0$. However, by the previous argument, again all words in the sequence can be written as $u'p'(q'p')^{l_i}\overline{u'}$.

The second step is to apply for the words in the sequence the same decompositions as in the proof of Proposition 14, i. e.,

- if $|u'| \leq \frac{|q'|}{2}$, then $q' = \overline{u'}xu'$ for some (possibly empty) pseudopalindrome x, and we set $p = u'p'\overline{u'}$ and $q = x$;
- if $\frac{|q'|}{2} < |u'| \leq |q'|$, we have that the prefix u' and the suffix $\overline{u'}$ overlap in q', i. e., $q' = xyxyx$ for some pseudopalindromes x and y, where $u' = xyx$, and we set $p = x$ and $q = yxp'xy$;
- if $|q| < |u|$ it follows that $u = xq$ for some suffix x of p', and this brings us back to one of the previous cases.

The obtained words p' and q' satisfy the requirements of our statement. □

Proposition 16. *Take the words $p, q, u \in \Sigma^*$ with p, q pseudopalindromes. If all pseudopalindromic prefixes of $upqp\overline{u}$ are trivial, then for any integer $i \geq 0$ so are those of $up(qp)^i\overline{u}$.*

Proof. First, assume that v is the shortest pseudopalindromic prefix of some word $up(qp)^k\overline{u}$, where $k \geq 0$. Since v is not a prefix of $upqp\overline{u}$, the length of up is less than the length of v, hence, we have $v = up(qp)^jx$, for some integer j with $0 \leq j \leq k$ and a word x which is either a prefix of qp or of $\overline{u}$.

If x is a prefix of $\overline{u}$, then $\overline{x}p(qp)^kx$ is a pseudopalindromic suffix (thus prefix) shorter than v itself. Therefore, for the remaining cases we can always consider that we have $1 < j < k$.

If x is a prefix of q, then $\overline{x}$ is a suffix of q. Hence, the last occurrence of p in v is preceded by $\overline{x}$ and followed by x, and we have that $\overline{x}px$ is a pseudopalindromic prefix of v, and, therefore, v is not the shortest.

If x is a prefix of qp but not of q, then $x = qx'$ and $\overline{x'}(qp)^iqx'$ is a pseudopalindromic suffix, hence, prefix of v, contradicting our assumption. This concludes our result. □

Since any pseudopalindromic prefix of $up(qp)^i\overline{u}$ would either have length at most $upqpu$ or will have this pseudopalindrome as a prefix or as a suffix, we have the following remark:

Remark 1. *If $up(qp)^i\overline{u}$, for some integer $i > 0$, does not have any pseudopalindromic prefixes, then $upqp\overline{u}$ does not have any, either.*

Theorem 17. *The iterated pseudopalindromic completion of a word w is regular if and only if w has either only pseudopalindromic prefixes or only pseudopalindromic suffixes, or for all words $w' \in w^{\ltimes_1}$ there exist the unique pseudopalindromes p and q with $|p| \geq 2$, such that:*

- *$w' \in p(qp)^+$*
- *w' has no pseudopalindromic prefixes except for the words in $p(qp)^*$.*

Proof. Due to Lemma 7, for $w^{\ltimes_*}$ we need only consider the finite union of all one sided iterated pseudopalindromic completion of words $w' \in w^{\ltimes_1}$.

For the first direction the result is easily obtained, since at each completion step from some word of form $p(qp)^n$ with $n \geq 1$ we get all words $p(qp)^{n+1}, p(qp)^{n+2}, \ldots, p(qp)^{2n}$, for $n \geq 1$. Thus, the final result is a finite union of regular languages, and the result follows.

Now assume that the iterated pseudopalindromic completion $w^{\ltimes_*}$, of some word w, is regular. The case when w has either only pseudopalindromic prefixes or only pseudopalindromic suffixes is trivial since in this case only one step of palindromic completion is possible. For the second case, following Theorem 9, $w^{\ltimes_*}$ is the union of some finite language $\{p \mid p \text{ pseudopalindrome}\}$ and some finite union of languages $\{qr(sr)^*\overline{q} \mid r, s \in \Sigma^* \text{ pseudopalindromes}\}$.

First we note that if $w \ltimes^* w'$ such that w' has no pseudopalindromic prefixes or suffixes, then according to Lemma 6, $w' \in w^{\ltimes_{\leq 1}}$ and therefore $w^{\ltimes_*} = w^{\ltimes_{\leq 1}}$ is itself finite. Hence, we neglect the case of the finite language $\{p \mid p \text{ pseudopalindrome}\}$ and consider the case when $w^{\ltimes_*}$ is the finite union of languages of form $\{qr(sr)^*\overline{q} \mid q, r, s \in \Sigma^* \text{ and } r, s \text{ pseudopalindromes}\}$.

With the help of the pigeon hole principle for a variable q, big enough integer k_1 and positive integer i_1, we have that $qr(sr)^{k_1}\overline{q} \ltimes^* qr(sr)^{k_1+i_1}\overline{q}$. We apply Proposition 15 and get that there exist some pseudopalindromes u, v, such that $qr(sr)^*\overline{q} \subset u(vu)^*$. Moreover, from Proposition 15, also all words obtained in the intermediate pseudopalindromic completion steps are in the language $qr(sr)^*\overline{q}$, hence, in $u(vu)^+$.

Therefore, we now know that there exist at most finitely many pairs of pseudopalindromes u, v, such that $w' \in u(vu)^+$. Assume there exist n pairs of pseudopalindromes (u_i, v_i) such that $w' \in u_i(v_iu_i)^+$ with $u_i \neq u_j$ and $|u_i| \geq 2$ for $1 \leq i, j \leq n$, where $i \neq j$. If $|u_1v_1| = |u_2v_2|$, then $|u_1| = |u_2|$ and since they are suffixes of the same word, $u_1 = u_2$ and, hence, $v_1 = v_2$, which is a contradiction. Hence, without loss of generality, we may assume $|u_1v_1| > |u_2v_2|$. In this case, $u_2v_2u_2$ is a pseudopalindromic prefix of $u_1v_1u_1$, and according to Lemma 13 we have $u_1v_1u_1, u_2v_2u_2 \in x_1(y_1x_1)^+$ for some pseudopalindromes x_1 and y_1. Repeating the argument for all the pairs (x_i, y_i) and (u_{i+2}, v_{i+2}), we conclude the proof. □

Next we consider what happens in the case of regular languages. We already know that the one step pseudopalindromic completion of a regular language is not necessarily regular. We have also seen that the iterated pseudopalindromic completion of a word may result in a non-context-free language. The following result is immediate:

Corollary 18. *The iterated pseudopalindromic completion of a regular language is not necessarily context-free.*

By [12, Lemma 3] the following is straightforward:

Lemma 19. *A pseudopalindrome w has period $p < |w|$ if and only if it has a pseudopalindromic prefix of length $|w| - p$.*

Proof. If u is a pseudopalindromic prefix of w, then it must be that u is also a suffix of w, and, as a result, $|w| - |u|$ is a period of the whole word.

In the same manner, whenever p is a period of w, we can write $w = u^ku'$ for some integer $k \geq 1$, where $|u| = p$ and u' is a prefix of u. However, since w is a pseudopalindrome, it follows that also $\overline{u'}$ is a prefix of u, thus u' is a pseudopalindrome. But, following the same argument, since w is a pseudopalindrome we get that u'' is also a pseudopalindrome, where $u = u'u''$. Therefore $(u'u'')^{k-1}u'$ with length $|w| - p$ is a pseudopalindromic prefix of w, and we conclude. □

First we make the following observation:

Lemma 20. *Let $L \subseteq up(qp)^*q'$ be a regular language, for some word u, pseudopalindromes p, q with $qp \neq \epsilon$ and word q' that is a prefix of either $\overline{u}$ or of $qp\overline{u}$ such that each word in L may only have pseudopalindromic prefixes of the form $p(qp)^i$, for some integer $i \geq 0$. Then $L^{\ltimes *}$ is regular if $L^{\ltimes *} \setminus L \subseteq up(qp)^*\overline{u} \cup V$ for some finite language of pseudopalindromes V.*

Proof. The language V contains only a finite number of pseudopalindromes that are obtained from sparse words in L, and thus it is regular. Let us assume that $L'' = L^{\ltimes_*} \setminus L \subseteq up(qp)^*\overline{u}$.

We first look at the right completion of the elements of L, which are all in the set L''. We can get the length of any pseudopalindromic suffix v, by which we can complete a word from L, from the equation $|v| = 2|u| + (k+1)|p| + k|q| - 2|u| - 2|x|$, where $xv = p(qp)^\ell q'$, for some integers $k, \ell \geq 0$ and word $x \in \Sigma^*$. Therefore,

$$|x| + |v| = (\ell+1)|p| + \ell|q| + |q'|, \tag{1}$$

and from the two equations we get that

$$|x| + (k+1)|p| + k|q| - 2|x| = (\ell+1)|p| + \ell|q| + |q'|.$$

It follows that

$$|x| + |q'| = (k-l)(|p| + |q|). \tag{2}$$

Now (1) and (2) give us that $|v| = (2\ell - k + 1)|p| + (2\ell - k)|q| + 2|q'|$ and this means $v = \overline{q'}p(qp)^{2\ell-k}q'$. However, from here we know that any suffix of the form $\overline{q'}p(qp)^*q'$ is a pseudopalindrome, hence

$$\{up(qp)^n q'\}^{\ltimes_1} = \{up(qp)^m\overline{u} \mid n \leq m \leq 2n\},$$

if $up(qp)^n q'$ does not have pseudopalindromic prefixes. Since for any regular language K there is a constant N_K, such that the difference between the lengths of two consecutive words is smaller than N_K, we get that there is a constant N' such that $L^{\ltimes_*} = R \cup \{up(qp)^n\overline{u} \mid n \geq N'\}$, where R is a finite language, so $L^{\ltimes_*}$ is regular.

Now suppose the word $up(qp)^n q' \in L$ had some pseudopalindromic prefix v, so it is of the form vx, for some word x. We know that $\overline{x}vx$ has to be in $up(qp)^*\overline{u}$, therefore vx is of the form $\overline{q'}(pq)^n p\overline{u}$, and we can employ the same length argument as above. Moreover, we have that

$$up(qp)^n q' = \overline{q'}(pq)^n p\overline{u},$$

is itself a pseudopalindrome. In this case, however, since the word can be completed in both directions and all its pseudopalindromic prefixes and suffixes are of the form $p(qp)^i$ with $i \geq 0$, the iterated completion of each word w will be an infinite regular language $\{p(qp)^m \mid m \geq N_w\}$, where N_w is a number which depends on the length of the word. Clearly then, there exists some number $N' = \min_{w \in L} N_w$, such that $L^{\ltimes_*} = \{p(qp)^m \mid m \geq N'\}$. $\square$

We are now ready to state one of the main results of this work.:

Theorem 21. *For a regular language L, its iterated pseudopalindromic completion $L^{\ltimes_*}$ is regular if and only if L can be written as the union of disjoint regular languages L', L'', and L''', where*

- $L'' = \{w \in L \mid w^{\ltimes_{\leq 1}} = w^{\ltimes_*}\}$ *and the completion of every word in L'' is a subset of a finite union of languages of the form $up(qp)^*\overline{u}$, where $upqp\overline{u}$ has no pseudopalindromic prefixes and p, q are pseudopalindromes;*

- $L''' = \{w \in L \mid w^{\ltimes_*} \setminus (w^{\ltimes_1}) \neq \emptyset\}$ *and, for an integer $m \geq 0$ depending on L and pseudopalindromes p_i, q_i such that p_iq_i have only one pseudopalindromic prefix and only one pseudopalindromic suffix, the completion of every word in L''' is a subset of $\bigcup_{i=1}^{m} p_i(q_ip_i)^+$;*

- $L' = L'^{\ltimes_{\leq 1}} = L \setminus (L'' \cup L''')$.

Proof. We start by first making the observation that the words in L'' and L''' are prefixes (suffixes) of length at least $|up| + \lceil\frac{|q|}{2}\rceil + 1$ and $|p_i| + \lceil\frac{|q_i|}{2}\rceil + 1$ long, respectively, because the shorter ones do not have a completion of the desired form or do not extend beyond one step completion when pq (and p_iq_i, respectively) is primitive. This does not make a difference for the characterisation, but only in the decision process.

For the first direction, we consider $L^{\ltimes_*}$ to be regular. Clearly, any language $L \subset \Sigma^*$ can be written as a union of three disjoint languages where: one of them (L'') has all the words which have either only pseudopalindromic prefixes or only suffixes and whose completion are subsets of a finite union of languages of the form $up(qp)^*\overline{u}$, where $upqp\overline{u}$ has no pseudopalindromic prefixes and p, q are pseudopalindromes; another one (L''') contains all the words which can be extended in both directions by pseudopalindromic completion such that all of their completions (one step completion is in fact enough according to Proposition 16 and Remark 1) are, for an integer $m \geq 0$ depending on L and pseudopalindromes p_i, q_i such that p_iq_i have only one pseudopalindromic prefix and only one pseudopalindromic suffix, a subset of $\bigcup_{i=1}^{m} p_i(q_ip_i)^+$; and the third one ($L'$) contains the words which have neither pseudopalindromic prefixes nor suffixes, or their completion is already in L, and they are not part of any of the two aforementioned languages.

Since $L^{\ltimes_*}$ is regular, $L^{\ltimes_*} \setminus L$ is a regular language as well. Moreover, $L^{\ltimes_*} \setminus L$ is a pseudopalindromic language, since all of its words are the result of pseudopalindromic completion. From Theorem 9 it follows that there exists a finite set of words x_i, r_i, s_i, where $i \in \{1, 2, \ldots, n\}$ and r_i, s_i are pseudopalindromes, such that the words in $L^{\ltimes_*} \backslash L$ are elements of $x_i r_i (s_i r_i)^* \overline{x_i}$ with $1 \leq i \leq n$.

First we identify the language L'''. For each j, using once more the pigeon hole principle, there exist big enough integers k_1 and k_2 with $k_1 < k_2$ and

$$x_j r_j (s_j r_j)^{k_1} \overline{x_j} \ltimes^* x_j r_j (s_j r_j)^{k_2} \overline{x_j},$$

or, for some $i \neq j$ and k_j we have

$$x_j r_j (s_j r_j)^{k_j} \overline{x_j} \ltimes^* x_i r_i (s_i r_i)^{k_1} \overline{x_i} \ltimes^* x_i r_i (s_i r_i)^{k_2} \overline{x_i}.$$

In the first case we apply Proposition 15 and get that there exist pseudopalindromes $p \neq \epsilon$ and q such that $x_j r_j (s_j r_j)^{k_i} \overline{x_j} \in p(qp)^+$ for $i \in \{1, 2\}$, and all intermediary words $x_j r_j (s_j r_j)^{k_j} \overline{x_j}$ are also in $p(qp)^+$. In the second case we can apply Proposition 15 to the second relation. Then by Lemma 13 and Proposition 15 we get that all three words are in $p(qp)^+$ for some suitable p, q. Furthermore, pq has no pseudopalindromic prefixes except for p and no pseudopalindromic suffixes except for q, since otherwise by Theorem 17 its iterated completion leads to non-regular languages. After finding these finitely many (say, m) pairs p_k, q_k, the language of all prefixes of $\bigcup_{k=1}^{m} p_k (q_k p_k)^+$ is a regular language, hence, its intersection with L is also regular.

At this point we have that $L^{\ltimes_*} \setminus L'''^{\ltimes_*} = L'^{\ltimes_*} \cup L''^{\ltimes_*}$ is regular, and, therefore $L_{\mathbf{diff}} = (L'^{\ltimes_*} \cup L''^{\ltimes_*}) \setminus L \subset L''^{\ltimes_*}$ is also a pseudopalindromic regular language. Again, from Theorem 9 we know that $L_{\mathbf{diff}}$ can be written as a finite union of languages of the form $up(qp)^* \overline{u}$. Clearly then, all words in L'' are prefixes of some word in $up(qp)^* \overline{u}$. Since by definition $L''^{\ltimes_{\leq 1}} = L''^{\ltimes_*}$, we know that the words in $up(qp)^* \overline{u} \cap L_{\mathbf{diff}}$ have no pseudopalindromic prefixes. Thus, following Remark 1 and using the fact that we can choose long enough words, we obtain that $upqp\overline{u}$ does not have any pseudopalindromic prefixes either. Let L'' be the finite union of the languages $\mathbf{Pref}(up(qp)^+) \cap L$, where $\mathbf{Pref}(A)$ is the language of all prefixes of A. This way, L'' is regular and since from it we obtain $L_{\mathbf{diff}}$ by pseudopalindromic completion, it meets the requirements. All that is left is to assign $L' = (L \setminus L''') \setminus L''$, which is regular and all its words either have only trivial pseudopalindromic prefixes or suffixes, or their pseudopalindromic completion is already in L.

Finally, let us consider that each of the three languages is regular, and prove that the iterated closure of the language L is also regular. First observe that $L'^* = L'$ is regular according to the hypothesis. Similarly, using the result of Lemma 20, we get that both $L''^{\ltimes_*}$ and $L'''^{\ltimes_*}$ are also regular.

We conclude that $L^{\ltimes_*}$ is regular, as it is a union of regular languages. $\square$

To see that the conditions imposed on the one step completion of the words in the languages L'' and L''' are justified, consider for example the word $w = abaacaa$. Then $w \ltimes \{abaacaaba, abaacaacaaba, aacaabaacaa\}$. As it can be seen the first of these words has a decomposition in terms of the pseudopalindromes $p = aba$ and $q = aca$, while the second has a decomposition in terms of $p = aba$ and $q = acaaca$. Furthermore, the last of these words can be decomposed in terms of $p = aacaa$ and $q = b$ with pq having both aa and p as pseudopalindromic prefixes. Thus, according to Theorem 17, its iterated completion is not a regular language. Since this can take place for every word in the infinite regular language $(abaaca)^*a$, our statement justifies the conditions imposed on L'''. Similarly, for L'' one could consider, for some $u \in \Sigma^+$, the word $uabaacaa$ and reach the same conclusion.

Furthermore, the definition of L' is also necessary in this form, as if a word w has its completion of the form $up(qp)^*\overline{u}$ but is in L, according to our definitions, then this word should be included in L''.

As a consequence of Theorems 9 and 21, the following result is obtained:

Corollary 22. *If for some regular language L we have that $L^{\ltimes_*}$ is regular, then for any integer $n \geq 1$ we have that $L^{\ltimes_n}$ are also regular. Moreover, the languages $L^{\ltimes_{\leq n}}$ are regular as well.*

In other words, when from a regular language we get a regular iterated pseudopalindromic completion language, all intermediate pseudopalindromic completion languages are regular.

5. Decidability questions

We conclude this paper with some algorithmic results which build on the previously obtained language theoretic characterisations. We begin with a series of preliminary remarks and results.

First, we provide some basic information on the computational model we use: the unit-cost RAM (Random Access Machine) with logarithmic word size. In this model (which is generally used in the analysis of algorithms) we assume that each memory cell can store $\mathcal{O}(\log n)$ bits, or, in other words, that *the machine word size* is $\mathcal{O}(\log n)$. The instructions are executed one after another, with no concurrent operations. The model contains common instructions: arithmetic (add, subtract, multiply, divide, remainder, shifts and bitwise operations, etc.), data movement (indirect addressing, load the content of a memory cell, store a number in a memory cell, copy the content of a memory cell to another), and control (conditional and unconditional branch, subroutine call and return). Each such instruction takes a constant amount of time. We point out that testing the equality of two numbers is also assumed to take a constant amount of time. Basically, this model allows us to measure the number of instructions executed in an algorithm, making abstraction of the time spent to execute each of the basic instructions.

In the upcoming algorithmic problems, we assume that the words we process are sequences of integers (called letters, for simplicity), each integer fitting in a single memory-word. Therefore, testing the equality of two letters is done in constant time, according to the definition of our computational model, given above. This is a common assumption in algorithmics on words (see, e. g., the discussion of Kärkkäinen et al in [20]).

We also recall basic facts about the data structures we use. For a word w, with $|w| = n$, over $\Sigma \subseteq \{1, 2, \ldots, n\}$ we can build in linear time a suffix array structure as well as data structures allowing us to retrieve in constant time the length of the longest common prefix of any two suffixes $w[i..n]$ and $w[j..n]$ of w, denoted $LCP_w(i, j)$. These structures are called LCP data structures in the following. For details, see, e. g., [20, 21].

Remark 2. *1. A number p is a period of w if and only if $LCP_w(1, p+1) = |w| - p$. There exists a factorisation $w = x^k$ with x a prefix of w if and only if $|x|$ is a period of w and $|w| = k|x|$. Thus, once LCP data structures are constructed for w, testing whether a number p is a period of w, as well as testing whether w is a power of a prefix $w[1..i]$, can be done in constant time.*
2. For a prefix x of w, one can compute in constant time the maximal value $k \geq 1$ such that x^k is a prefix of w. Indeed, $k = 1 + \left\lfloor \frac{LCP_w(1,1+|x|)}{|x|} \right\rfloor$.

5.1. Membership problems

We start this part of our investigation with the observation that while in the classical hairpin completion case the extension of a word is both to the right and the left of the word, here, by Lemma 7, the two extensions are identical for pseudopalindromes, hence, for all words starting with the second step of the derivation, therefore making the problem simpler.

The membership problem for the one step pseudopalindromic completion of a word is trivial as one has to check for the shorter word if it is a prefix while its θ image is a suffix of the longer one, or vice-versa, and these two occurrences overlap. Obviously, the time needed for this is linear. A more interesting problem is the membership problem for the iterated pseudopalindromic completion. While the problem is obviously decidable, the naive approach leads to a quadratic-time algorithm. We show that in this setting it is still solvable in linear time.

Lemma 23. *If u is a pseudopalindromic prefix of another pseudopalindrome v such that $|u| > \lceil |v|/2 \rceil$, then $u \ltimes v$.*

Proof. For words v with length at most 7, the result is easy to check. Thus assume that $|v| > 7$. Hence, it must be that $|u| > 5$. Since u is a pseudopalindromic prefix and suffix of v, and $|u| > \lceil |v|/2 \rceil$ there exists a factorisation of it $u = xy\overline{x}$ for some $x, y \in \Sigma^*$ such that $v = xy\overline{x}\,\overline{y}x$ and $|x| > 1$. But v is a pseudopalindrome itself, thus $x = \overline{x}$ and $y = \overline{y}$. We conclude that $u = xyx$ and $v = xyxyx$ and the result follows, since both x and y are pseudopalindromes. □

With the following result we show that for two pseudopalindromes u and v, we say that v can be obtained from u if and only if u is a prefix of v and all pseudopalindromic prefixes of v greater than u have as prefix some other pseudopalindrome of length greater than half theirs.

Proposition 24. *For two pseudopalindromes u, v, we have $u \ltimes^* v$ if and only if u is a prefix of v with $|u| > 2$ and every prefix w of v with length greater than $2(|u|-1)$ has a pseudopalindromic prefix of length greater than $\lceil |w|/2 \rceil$.*

Proof. Note that the condition that $|u| > 2$ is necessary such that the pseudopalindromic completion operation can be applied on the word u.

The case when $u \ltimes^* v$ is quite obvious. Since we start with the pseudopalindrome u, we always have after a completion step u as both prefix and suffix. Moreover, after each step the pseudopalindrome on which we do the completion is both a prefix and a suffix of the new word, and the two occurrences overlap.

Now let us consider the other direction. In order for v to be part of the iterated pseudopalindromic completion of a word it must be that the second of the properties holds. Taking into account that v starts with u and that the second property holds, with the help of Lemma 23 we get that v is in the language given by the iterated pseudopalindromic completion of u. □

Lemma 25. *For a word v, one can identify in time $\mathcal{O}(|v|)$ all its pseudopalindromic prefixes.*

Proof. We first construct in linear time *LCP* data structures for the word $w = v\overline{v}$. Further, we compare for each $i \leq |v|$ the words $w[1..2|v|]$ and $w[i + |v|..2|v|]$, and establish that $v[1..|v| - i + 1]$ is a pseudopalindrome if and only if the length of the longest common prefix of the two compared words is at least $|v| - i + 1$. $\square$

The previous three technical results lead to the aforementioned linear algorithm.

Theorem 26. *One can decide in linear time if for two words u and v, where v is a pseudopalindrome of length n greater than $|u|$, we have $u \ltimes^* v$.*

Proof. By Proposition 24, it suffices to check two things: if u or $\overline{u}$ is a prefix of v, while its counterpart a suffix, which is done in linear time, and then whether all pseudopalindromic prefixes of v of length greater than $|u|$, have as prefix a pseudopalindrome of length more than half of theirs plus one.

By Lemma 25, we identify in time $\mathcal{O}(|v|)$ the set of pseudopalindromic prefixes of v. Next, looking at the lengths of all elements in this set, we need to check that the difference between no two consecutive increasing ones is greater than double the smallest of them; again linear time is enough to do this, and the conclusion follows. $\square$

Using a strategy similar to the previous proof together with the result of Lemma 25, one can efficiently compute the pseudopalindromic completion distance between two given words u and v. We start with the longest element of $u^{\ltimes_1}$, and in each step choose v's longest pseudopalindromic prefix which is shorter than twice the length of the current one minus one. The greedy technique can be proved to be optimal by Proposition 24, while Lemma 23 proves the correctness of each step. Therefore, we proved:

Theorem 27. *Given a word u and a pseudopalindrome v of length n greater than $|u|$, one can compute in linear time the minimum number of pseudopalindromic completion iterations needed in order to get from u to v, when possible.*

The following is a more general result related to Theorem 26. In what follows, a deterministic finite automaton (DFA) is defined by a quintuple $A = \langle Q, \Sigma, q_0, \sigma, F\rangle$, where Q is the set of states out of which q_0 is the initial one, Σ the input alphabet, $\sigma : Q \times \Sigma \to Q$ the transition function, and F the set of final states. To define the computation of a DFA A on a word w the function σ is extended to work on arguments from $Q \times \Sigma^*$ as follows: $\sigma(q, \epsilon) = q$ and $\sigma(q, wa) = \sigma(\sigma(q, w), a)$ for $q \in Q$, $w \in \Sigma^*$ and $a \in \Sigma$. A word w is accepted by A if and only if $\sigma(q_0, w) \in F$. Recall that the class of languages accepted by DFAs is exactly the class of regular languages. For details on finite automata and closure properties, see the book of Harrison [16].

Theorem 28. *Given a word w of length n and a regular language L accepted by a DFA with q states, one can decide in $\mathcal{O}(max(n, q))$ if $w \in L^{\ltimes_*}$.*

Proof. Obviously, if $w \in L$, then we answer YES, otherwise we move on. Next we check if w is a pseudopalindrome. If the answer is NO, then we answer negatively to the question.

Now we know that in time $\mathcal{O}(n)$ we can find all pseudopalindromic prefixes of w. We construct a vector ok of length n with $ok[i] = 1$ whenever $w[1..i]$ is pseudopalindrome and $ok[i] = 0$ otherwise, for all $0 < i \leq n$.

Next, we feed the word to the automaton and for each integer i with $w[1..i] \in L$ we make $ok[i] = 2$. At this point our vector ok has on position i a 2 if $w[1..i]$ is a prefix of w accepted by the automaton and a 1 if $w[1..i]$ is a pseudopalindrome.

Finally, we set $j = n$ and start traversing the vector ok downwards. If there exists $i > j/2 - 1$ and $ok[i] > 1$, then we answer YES as a direct consequence of Lemma 23 and Proposition 24. Otherwise, if there exists $i > j/2 - 1$ and $ok[i] = 1$, then we update $j = i$ and continue with the next i. The answer is negative when the entire vector has been traversed without providing YES as an answer. □

5.2. Regularity problems

Further, we focus on problems in which we are required to decide whether the iterated pseudopalindromic completion of a certain set is regular or not.

Theorem 29. *For some word w of length n, one can decide in $\mathcal{O}(n)$ whether its iterated pseudopalindromic completion $w^{\ltimes_*}$ is regular.*

Proof. According to Theorem 17, the iterated pseudopalindromic completion of w is regular if and only if w has either only pseudopalindromic prefixes or only pseudopalindromic suffixes, or there exist the pseudopalindromes p and

q such that any word w' obtained within a one completion step from w is in the language $p(qp)^+$ and all the pseudopalindromic prefixes of w' are also in the language $p(qp)^*$.

Following Lemma 25 we can compute in linear time all the pseudopalindromic prefixes of w. Applying the same result for the mirror image of w, we can compute, also in linear time, all the pseudopalindromic suffixes of w. Note that in this process we computed LCP data structures for the word w.

In the case when w has only pseudopalindromic prefixes or only pseudopalindromic suffixes we can already conclude that the iterated pseudopalindromic completion of w is regular. Therefore, let us assume in the following that w has at least one pseudopalindromic prefix, as well as at least one pseudopalindromic suffix.

We will only present the case when there exist the pseudopalindromes p and q such that any word w' that can be obtained in one right completion step from w is in the language $p(qp)^+$ and all the pseudopalindromic prefixes of w' are also in $p(qp)^*$. Exactly the same ideas (in a sense, the arguments applied are mirrored) can be used to test whether any word w' that can be obtained in one left completion step from w is in the language $p(qp)^+$ and all the pseudopalindromic prefixes of w' are also in $p(qp)^*$.

Next, we show how these properties can be checked in linear time.

A very simple observation is that, in fact, if the iterated completion of w is regular, then the shortest pseudopalindromic prefix of w is the word p, whose existence is stated in Theorem 17, while the second shortest pseudopalindromic prefix of w is the word pqp. Nevertheless, w should have the form $w = p(qp)^k x$, where x is a prefix of qp. Using LCP_w queries we can check in constant time whether p and q can be correctly computed; that is, once p found, we know that the second largest pseudopalindrome ends with p, so we just check whether its length is at least $2|p|$, and identify its middle part as q. Moreover, we check whether w has the desired form, which is equivalent to checking (in constant time, by Remark 2), whether $w[1+|p|..n]$ has the period $|qp|$. If one of these checks fails, the completion of w is not regular. Let us assume that they were both true.

Thus, we located an occurrence of p and an occurrence of q in the word w in constant time, once the pseudopalindromic prefixes of w were computed. Now we can easily check whether all the pseudopalindromic prefixes of w are of the form $p(qp)^*$, which can be done by just checking whether the length difference between two consecutive such pseudopalindromes is $|p|+|q|$. Assume, again, that this holds, so the only thing left to do is to check under which conditions for any pseudopalindromic suffix v of w which allows for a completion, the resulting word w' is in $p(qp)^+$. There are three cases to consider depending on the length of x with respect to $|qp|$.

If $|x| = |qp|$ or $x = \epsilon$, then we are done since the pseudopalindromic suffixes will be the same as the prefixes, hence all belong to $p(qp)^+$.

In the case $|x| < |q|$, note that for the resulting word to be of length $|p| + i|qp|$, for some integer i with $2k > i \geq k$, we need to have $2(|p| + k|qp| + |x|) = |v| + |p| + i|qp|$. From here, we get $|v| = 2|x| + |p| + (2k - i)|qp|$. Since, we have the list of pseudopalindromic suffixes of w, for each v from this list we can check in constant time whether $(|v| - 2|x| - |p|) = 0 \mod |qp|$. If this is true for all pseudopalindromic suffixes, then the iterated completion of w is regular, otherwise it is not.

Finally, if $|q| \leq |x| < |qp|$, then $x = qx'$, such that $0 \leq |x'| < |p|$. Just as above, the pseudopalindromic completion will be in $p(qp)^+$ if and only if $|v| = 2|x'| + |q| + \ell|qp|$, for some positive integer ℓ, a condition which can be again checked in constant time for each pseudopalindromic suffix.

This concludes our proof. □

In the following results we are no longer concerned with computational complexity issues, switching our focus to showing that several problems are decidable. We assume in all the following statements that the given language L is presented to us as a DFA $< Q, \Sigma, q_0, \sigma, F >$ with $|Q| = n$ states.

Theorem 30. *Given a regular language L, one can decide whether $L = L^{\ltimes_*}$.*

Proof. In order to prove that $L \neq L^{\ltimes_*}$, we need to show that there exists some non-empty word u and pseudopalindrome p of length at least two, such that $up \in L$ and $up\overline{u} \notin L$. Let us suppose that u is the shortest such word. We show that, should u exist we can find it after finitely many steps. Let L_{ul} denote the language $\{w \mid \sigma(q_0, w) = \sigma(q_0, u)\}$.

Define F_u as the set of final states which we can reach by a pseudopalindrome after first reading u, that is,

$$F_u = \{q \in F \mid \text{ exists a pseudopalindrome } w \text{ with } \sigma(q_0, uw) = q\}$$

and the language accepted starting from one of these states

$$L_{ur} = \{v \mid \text{ exist } p \in F_u \text{ and } q \in F \text{ with } \sigma(p, v) = q\}.$$

Then, u is the shortest word in $L_{ul} \setminus L_{ur}^{\theta} = L_{ul} \cap (\Sigma^* \setminus L_{ur}^{\theta})$, where $L^{\theta} = \{\overline{v} | v \in L\}$ is the θ image of L. Note that the languages L_{ul} and L_{ur}

depend only on the state to which our supposed u takes the automaton, hence all possibilities can be accounted for by only considering all the states of the automaton. The sets F_u and L_{ur} are used to obtain a length bound on the candidate for u, therefore they do not need to be effectively constructed, although such a construction is possible. The number of the states of the automaton $L_{ul} \setminus L_{ur}^{\theta}$ is unfortunately quite high, and, hence so is the length up to which we have to check all words whether they might be equal to u:

- the automaton accepting L_{ul} has at most n states;
- for L_{ur} we have an NFA of at most n states;
- reversal and determinisation of the automaton accepting L_{ur}^{θ} takes it up to at most 2^{n+1} states (see the work of Holzer and Kutrib [22]);
- $L_{ul} \cap (\Sigma^* \setminus L_{ur}^{\theta})$ results in an automaton with at most $n2^{n+1}$ states, giving us that the shortest word accepted by it is at most as long as the number of states.

Therefore, for all words u with length $|u| \leq n2^{n+1}$, we have to check whether $((u \cdot \mathcal{PP}al) \cap L)\overline{u} \setminus L = \emptyset$. If for at least one word the set is not empty, we answer NO, otherwise YES.

The decision procedure for each word u involves effective constructions of finite and pushdown automata, and emptiness check of a context-free language. Since $u \cdot \mathcal{PP}al$ is a linear language (hence context-free), a pushdown automaton accepting its intersection with the regular language L can be constructed. Concatenation with $\overline{u}$ and intersection with the complement of L still results in a context-free language, hence can be effectively tested for emptiness. For details on these well established methods we refer the reader to any textbook on formal languages and automata (see, e. g., [16]). □

The following theorem states the main result of this section.

Theorem 31. *Given a regular language L, one can decide whether $L^{\ltimes *}$ is regular. If the answer is YES, provided one has an automaton for L one can construct an automaton accepting $L^{\ltimes *}$.*

Proof. The outline of the decision procedure is based on the description of $L^{\ltimes *}$ given in Theorem 21. Thus, first we identify the words p_i, q_i forming L''', if any exist. Then we construct a DFA which accepts $L' \cup L'' = L \setminus L'''$. In the resulting automaton we check for the words u_k, p_k and q_k, which form

L'', if any exist, and construct the automaton for $L' = (L \setminus L''') \setminus L''$. Last, we check whether $L' = L'^{\ltimes_*}$, that is $L' = L'^{\ltimes_{\leq 1}}$, with the help of Theorem 30. If yes, then $L^{\ltimes_*}$ is regular, otherwise it is not.

The automata for the intermediary steps are computable using well-known algorithms (see, e. g., [16]). What we have to show, is that the words u_k, p_k, q_k can be found, given an automaton. First, we check every cycle of length at most N_L in the automaton, where N_L is a constant computable from the representation of L (for the argument on N_L see the last part of the proof). This can be done by a depth-first search. If the label of the cycle can be written as pq for some pseudopalindromes $p \neq \epsilon$ and q, then we check all paths w of length at most N_L, which lead to the cycle from the initial state and all paths v of length at most N_L, going from the cycle to a final state. If there exist pseudopalindromes $x \neq \epsilon$ and y such that xy is a cyclic shift of pq and $wpqv$ is a prefix or suffix of a word in $x(yx)^+$, then we identified a pair p_i, q_i for L'''. If there exist pseudopalindromes $x \neq \epsilon$ and y, and some word u, such that xy is a cyclic shift of pq and $wpqv = ux(yx)^i$ for some $i \geq 1$, then we identified a triple u_k, p_k, q_k for L''.

Observe here, that to check that both hypotheses for L'' and L''' are fulfilled by these pairs and triples it is enough to look at their pseudopalindromic prefixes and suffixes and check if they have the length equal to $|q| + 2|q'|$ or $(m+1)|p| + m|q| + 2|q'|$, for some prefix q' of qp. The latter of these two can, in fact, be checked just by looking at the shortest form of a word (that is $up(qp)^i q'$ and $q'p(qp)^i u$ for i minimal among all indices of words of this form). Since both checks can be done within a finite number of steps, the conditions can be verified.

After finding all pairs p, q for L''', we construct for each of them the automaton accepting $L \setminus L_{pq}$, where L_{pq} is the set of prefixes of $p(qp)^+$ longer than $|p| + \lceil \frac{|q|}{2} \rceil + 1$. The intersection of these languages (the $L \setminus L_{pq}$ languages) is $L' \cup L''$. Afterwards we subtract, for each triple u, p, q forming L'', the language of prefixes of $up(qp)^+\overline{u}$ which are longer that $|up| + \lceil \frac{|q|}{2} \rceil + 1$ (since shorter prefixes cannot be completed into a word from $up(qp)^+\overline{u}$). The resulting language is our candidate for L'.

As mentioned above, if $L' = L'^{\ltimes_{\leq 1}}$, output YES, otherwise NO.

Apart from the automata construction techniques mentioned in the proof of the previous theorem, we only use a method which builds the automaton accepting the language of prefixes of $p(qp)^+$, for given words p, q. This can be done by constructing the automata for the languages $(pq)^*p'$, with p' one of the finitely many prefixes of p, and then the automaton accepting the union of these languages.

We end the proof by showing that N_L is computable from the presentation of L, as it is the number of states of a newly constructed automaton.

If $L^{\ltimes_*}$ is regular, then so is $L^{\ltimes_1}$, by Corollary 22. If $L^{\ltimes_1}$ is regular, then Theorem 9 applies to $L^{\ltimes_*} \setminus L$ and gives us that it can be written as the finite union of languages of the form $xr(sr)^*\overline{x}$, with r, s pseudopalindromes.

For every state $p \in Q$, let us define the languages $\mathrm{LEFT}_p = \{u \mid \sigma(q_0, u) = p\}$ and $\mathrm{RIGHT}_p = \{u \mid \exists q \in F : \sigma(p, u) = q\}$. For every pair of states $p \in Q$, $q \in F$, let L_{pq} denote the language $\mathrm{LEFT}_p \setminus \theta(\mathrm{RIGHT}_q)$, when $\sigma(p, w) = q$ for some pseudopalindrome $w \notin \Sigma \cup \{\epsilon\}$, and $L_{pq} = \emptyset$, otherwise. Now,

$$L_c = \bigcup_{p,q \in Q} L_{pq}$$

is a regular language, as it is a finite union of regular languages. Also, every word in L_c is the prefix of a word in one of the finitely many languages $xr(sr)^*\overline{x}$ mentioned above. If L_c is infinite, then by Lemma 3 and Theorem 4 we get that the label of every cycle in the automaton accepting L_c is of the form w^k, where w is a cyclic shift of pq and $k \geq 1$. Hence, the same holds for cycles of length at most m, where m is the number of states of the automaton accepting L_c. On the other hand, suppose there is a pair r_1, s_1, such that all cycles which are cyclic shifts of $(r_1 s_1)^k$ for some $k \geq 1$ are longer than m. Then, again by Lemma 3 and the pigeon hole principle, we get that $r_1 s_1$ is the cyclic shift of some other pair r_2, s_2, where $|r_2 s_2| \leq m$. Hence, we conclude that by checking all cycles of length at most m of the automaton accepting L_c we discover the pairs r, s from the characterisation in Theorem 9. The automaton accepting L_c can be constructed, given L, and m is computed by counting the states, hence take N_L to be m. □

6. Further remarks

We see that in the context of pseudopalindromes some of the decidability questions raised in the context of hairpin completion are decidable. This is mainly due to the fact that in this context the right and the left completions are equal. Moreover, to this contributes also the fact that the length of the hairpin-loop is at most one in our case. Note, that in the biological phenomenon serving as inspiration for the model, the length of the hairpin in the case of stable bindings is limited (optimal loop length is of about 4-8 base-pairs) not only making the model more easily approachable, but also keeping close to the original motivation.

Therefore, we suggest as future work the study of k-pseudopalindromes, structures that bind together perfectly but for some middle part of length at most k, for some fixed integer $k \geq 0$. Formally, a word which can be written as $uv\overline{u}$, with $|v| \leq k$, is a k-pseudopalindrome. In line with the above, the 0-pseudopalindromes correspond to the even length pseudopalindromes studied here. That is, $\mathcal{PP}al_0 \subset \mathcal{PP}al$. However, when the odd length pseudopalindromes are added, the gain is not that big, thus $\mathcal{PP}al \subset \mathcal{PP}al_1$. To see this, please note that beside our odd length pseudopalindromes, the class $\mathcal{PP}al_1$ also includes words of the form $wa\theta(w)$, such that $a \neq \theta(a)$. Furthermore, it follows from the definition that for any integers i and j with $0 \leq i < j$, we have $\mathcal{PP}al_i \subset \mathcal{PP}al_j$. As a first observation, let us point out that Theorem 9 can be stated and proved for k-pseudopalindromes much the same way as it was given earlier for pseudopalindromes.

Theorem 32. *For a fixed integer $k \geq 0$, a regular language $L \subseteq \Sigma^*$ is k-pseudopalindromic, if and only if it is a union of finitely many languages of the form $L_p = \{p\}$ or $L_{r,s,q} = qr(sr)^*q^R$ where p is a k-pseudopalindrome, r and s are (0-)pseudopalindromes, and q is an arbitrary word.*

Proof. It is straightforward that if L follows the description in the statement, then L is both regular and k-pseudopalindromic.

Let us now consider the other direction. Since L is regular, according to Lemma 3 there exists a constant k_L such that for any word $w \in L$ longer than k_L we have a factorisation $w = uvz$ with $0 < |uv| \leq k_L$ and $v \neq \epsilon$, such that $uv^iz \in L$ for any $i \geq 0$. Note that since k_L is fixed, the number of words that do not fulfil this property is finite, and since they are all k-pseudopalindromes they will be in fact included in the set L_p of our statement.

The two cases being symmetric, we assume $|u| \leq |z|$. According to the same Lemma 3, since L is a k-pseudopalindromic language, it follows that for any integer $i \geq 0$ we have that $uv^iz \in L$ is a k-pseudopalindrome, thus, for a factorisation $z = x\overline{u}$ with $x \in \Sigma^*$, we have that v^ix is a k-pseudopalindrome. This gives us that there exists an integer $j \geq 0$ such that $x = v_1\overline{v}^j$, where $\overline{v} = v_2v_1$. Since i can be arbitrarily large and k is fixed, there exists some value of i with $i|v| > (2j+1)|v| + k$, such that $uv^iz, uv^{i+1}z \in L$. For a k-pseudopalindrome, apart from at most k mismatches in the middle, the prefixes have to be θ-images of the suffixes, therefore we get $\overline{\overline{v_2v_1}^{j+1}} =$

$\overline{v_2}v_1(v_2v_1)^j$. Moreover, we have that v_2 is a pseudopalindrome. Comparing the matching ends further gives us $\overline{\overline{v_2}\overline{v_1}^{j+1}\overline{v_1}} = \overline{v_1}\,\overline{v_2}v_1(v_2v_1)^j$, and hence v_1 is a pseudopalindrome, too.

We conclude that our original word w can be written as $uv_1(v_2v_1)^{j+1}\overline{u}$, with v_1 and v_2 pseudopalindromes. According to Lemma 3 a similar decomposition exists for all words longer than k_L. Since all parts of the factorisation, u, v_1 and v_2 are shorter than k_L, the existence of finitely many such triplets, and therefore our results, are concluded. □

Given that the characterisation of k-pseudopalindromic languages can be so readily obtained from the characterisation of pseudopalindromic languages, it is justified to ask why we did not state our results in this more general setting. The answer is that later results, in particular Lemma 19 and Theorem 21, rely heavily on pseudopalindromic completion being independent of which side of a pseudopalindrome we apply it to (see Lemma 7). This, however, is not the case with k-pseudopalindromes, hence a different approach is needed to characterise regular languages whose iterated k-pseudopalindromic closure is regular.

7. Acknowledgement

A shorter version of this paper (see the reference [23]) was presented at the 16th International Conference on Developments in Language Theory in 2012. The work of Florin Manea was supported by the *DFG* grant 596676. The work of Robert Mercaş was supported by the *Alexander von Humboldt Foundation* and *DFG* grant 582014.

The authors thank the anonymous reviewers for there valuable comments that improved in a great measure the presentation of this paper.

References

[1] G. Păun, G. Rozenberg, T. Yokomori, Hairpin languages, International Journal of Foundations of Computer Science (2001) 837–847.

[2] D. Cheptea, C. Martín-Vide, V. Mitrana, A new operation on words suggested by DNA biochemistry: Hairpin completion, Transgressive Computing (2006) 216–228.

[3] V. Diekert, S. Kopecki, V. Mitrana, On the hairpin completion of regular languages, in: M. Leucker, C. Morgan (Eds.), ICTAC, Vol. 5684 of Lecture Notes in Computer Science, Springer, 2009, pp. 170–184.

[4] L. Kari, S. Kopecki, S. Seki, Iterated hairpin completions of non-crossing words, in: M. Bieliková, G. Friedrich, G. Gottlob, S. Katzenbeisser, G. Turán (Eds.), SOFSEM, Vol. 7147 of Lecture Notes in Computer Science, Springer, 2012, pp. 337–348.

[5] F. Manea, C. Martín-Vide, V. Mitrana, On some algorithmic problems regarding the hairpin completion, Discrete Applied Mathematics 157 (9) (2009) 2143–2152.

[6] F. Manea, C. Martín-Vide, V. Mitrana, Hairpin lengthening, in: F. Ferreira, B. Löwe, E. Mayordomo, L. M. Gomes (Eds.), CiE, Vol. 6158 of Lecture Notes in Computer Science, Springer, 2010, pp. 296–306.

[7] F. Manea, V. Mitrana, Hairpin completion versus hairpin reduction, in: S. B. Cooper, B. Löwe, A. Sorbi (Eds.), CiE, Vol. 4497 of Lecture Notes in Computer Science, Springer, 2007, pp. 532–541.

[8] F. Manea, V. Mitrana, T. Yokomori, Two complementary operations inspired by the DNA hairpin formation: Completion and reduction, Theoretical Computer Science 410 (4-5) (2009) 417–425.

[9] F. Manea, V. Mitrana, T. Yokomori, Some remarks on the hairpin completion, International Journal on Foundations of Computer Science 21 (5) (2010) 859–872.

[10] M. Ito, P. Leupold, F. Manea, V. Mitrana, Bounded hairpin completion, Information and Computation 209 (3) (2011) 471–485.

[11] S. Kopecki, On iterated hairpin completion, Theoretical Computer Science 412 (29) (2011) 3629–3638.

[12] A. de Luca, Sturmian words: Structure, combinatorics, and their arithmetics, Theoretical Computer Science 183 (1) (1997) 45–82.

[13] A. de Luca, A. D. Luca, Pseudopalindrome closure operators in free monoids, Theoretical Computer Science 362 (1-3) (2006) 282–300.

[14] K. Mahalingam, K. G. Subramanian, Palindromic completion of a word, in: BIC-TA, IEEE, 2010, pp. 1459–1465.

[15] A. Slama-Schwok, E. Brossalina, Y. Demchenko, M. Best-Belpomme, V. Vlassov, Structural flexibility of a DNA hairpin located in the long terminal repeat of the drosophila 1731 retrotransposon, Nucleic Acids Research 26 (22) (1998) 5142–5151.

[16] M. A. Harrison, Introduction to Formal Language Theory, Addison-Wesley, Reading, Massachusetts, 1978.

[17] M. Lothaire, Combinatorics on Words, Cambridge University Press, 1997.

[18] L. Kari, K. Mahalingam, Watson–Crick palindromes in DNA computing, Natural Computing 9 (2) (2010) 297–316.

[19] S. Horváth, J. Karhumäki, J. Kleijn, Results concerning palindromicity, Journal of Information Processing and Cybernetics 23 (1987) 441–451.

[20] J. Kärkkäinen, P. Sanders, S. Burkhardt, Linear work suffix array construction, Journal of the ACM 53 (2006) 918–936.

[21] D. Gusfield, Algorithms on strings, trees, and sequences: computer science and computational biology, Cambridge University Press, New York, NY, USA, 1997.

[22] M. Holzer, M. Kutrib, Nondeterministic descriptional complexity of regular languages, International Journal of Foundations of Computer Science 14 (6) (2003) 1087–1102.

[23] S. Z. Fazekas, R. Mercaş, K. Shikishima-Tsuji, Hairpin completion with bounded stem-loop, in: O. H. Ibarra, H.-C. Yen (Eds.), DLT, Vol. 7410 of Lecture Notes in Computer Science, Springer, 2012, pp. 428–439.