Formal Language Theory of Hairpin Formations

Von der Fakultät Informatik, Elektrotechnik und
Informationstechnik der Universität Stuttgart zur Erlangung der
Würde eines Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte Abhandlung

Vorgelegt von

Steffen Kopecki

aus Stuttgart

Hauptberichter:	Prof. Dr. Volker Diekert
Mitberichter:	Prof. Dr. Victor Mitrana
	Priv.-Doz. Dr. Dirk Nowotka
Tag der mündlichen Prüfung:	9. Juni 2011

Institut für Formale Methoden der Informatik
Universität Stuttgart

2011

Acknowledgement

I owe my deepest gratitude to my adviser, Volker Diekert, for an excellent supervision and for many valuable discussions. Furthermore, I am grateful to my co-examiner, Victor Mitrana, who introduced me to this very interesting topic and also to Dirk Nowotka for examining my thesis. Finally, I would like to thank all members of the Institute for Formal Methods in Computer Science of the University of Stuttgart for a great working atmosphere and for various interesting and helpful discussions.

Contents

Abstract

The (bounded) hairpin completion, the hairpin lengthening, and their iterated versions are operations on formal languages which have been inspired by the hairpin formation in DNA biochemistry. In this paper we discuss the hairpin formations from a language theoretic point of view.

The hairpin completion of a formal language has been defined in 2006. In the first paper on this topic it has been shown that the hairpin completion of a regular language is not necessarily regular but always linear context-free. In the same paper the open problem was stated, if it is decidable, whether the hairpin completion of a regular language is regular. We solved this problem positively in 2009. Here, we prove that the problem is NL-complete and we present an algorithm whose time complexity is bounded by a polynomial of degree 8. As a by-product of our technique to prove the complexities, we obtain that the hairpin completion of a regular language is actually an unambiguous linear context-free language. In addition, we provide results concerning language classes within the regular languages. We show that if the hairpin completion of an aperiodic (i. e., star-free) language is regular, then the hairpin completion is aperiodic, too. The same is true for the language class induced by the variety **LDA**.

The hairpin lengthening is a variant of the hairpin completion which has been investigated first in 2010. The hairpin lengthening of a regular language seems to behave quite similar to the hairpin completion: It is not necessarily regular but linear context-free. We prove, however, that the hairpin lengthening of a regular language may be an inherent ambiguous linear language. We also consider the problem if it is decidable, whether the hairpin lengthening of a regular language is regular, but we are only able to prove decidability for the one-sided case of hairpin lengthening. (The one-sided case is closer to biochemistry, yet the two-sided case is more interesting from a theoretic point of view).

The bounded hairpin completion can be seen as a weaker variant of the hairpin completion. It is well-known that all language classes in the Chomsky hierarchy are closed under bounded hairpin completion and that context-free, context-sensitive, and recursively enumerable languages are closed under iterated bounded hairpin completion. In 2009 it was asked in literature, whether the regular languages are closed under iterated bounded hairpin completion as well. We solve this question by presenting a more general result. We give an effective representation for the iterated bounded hairpin completion which uses union, intersection with regular sets, and concatenation with regular. Thus, all language classes which are (effectively) closed under these basic operations are also (effectively) closed under iterated bounded hairpin completion. This applies to all classes in the Chomsky hierarchy and to all usual complexity classes. Furthermore, we give an exponential lower and up-

per bound for the size of non-deterministic finite automata accepting the iterated bounded hairpin completion of a regular language.

The iterated (unbounded) hairpin completion of a regular language is known to be context-sensitive and it may be not context-free. However, it is was not known whether the iterated hairpin completion of a singleton (or finite language) is always regular or context-free. This was stated as an open problem in 2008. In contrast to the previous questions this one has a negative answer. We give an example of a singleton whose iterated hairpin completion is not context-free.

Keywords: Hairpin completion, hairpin lengthening, formal languages, automata theory, computational complexity

Zusammenfassung

Die (begrenzete) Haarnadel Vervollständigung, die Haarnadel Verlängerung und ihre iterierten Varianten sind Operationen auf formalen Sprachen, welche durch die Haarnadelstruktur in der DNA Biochemie inspiriert sind. In dieser Arbeit behandeln wir die formalsprachliche Theorie der Haarnadelstrukturen.

In 2006 wurde die Haarnadel Vervollständigung von formalen Sprachen eingeführt. Aus der ersten Arbeit über Haarnadel Vervollständigungen ist bekannt, dass die Haarnadel Vervollständigung einer regulären Sprachen nicht regulär sein muss, aber immer linear kontextfrei ist. Ebenfalls in 2006 wurde das offene Problem gestellt, ob entscheidbar ist, ob die Haarnadel Vervollständigung einer regulären Sprache wieder regulär ist. Wir haben das Problem in 2009 gelöst. Hier zeigen wir, dass das Problem NL-vollständig ist, und wir werden einen Entscheidungsalgorithmus für das Problem angeben, dessen Zeitkomplexität durch ein Polynom achten Grades beschränkt ist. Aus dem Beweis der Komplexität können wir zudem folgern, dass die Haarnadel Vervollständigung einer regulären Sprache durch eine eindeutige linear kontextfreie Grammatik erzeugt wird. Darüber hinaus untersuchen wir die Haarnadel Vervollständigung von Sprachklassen innerhalb der regulären Sprachen. Wir zeigen, dass, wenn die Haarnadel Vervollständigung einer aperiodischen (d. h. sternfreien) Sprache regulär ist, dann ist sie wieder aperiodisch. Ein analoges Resultat zeigen wir für die Sprachklasse, welche durch die Varietät **LDA** induziert wird.

Die Haarnadel Verlängerung ist eine Variante der Haarnadel Vervollständigung, welche 2010 eingeführt wurde. Für die Haarnadel Verlängerung einer regulären Sprache gilt ebenfalls, dass sie nicht regulär sein muss, aber immer linear kontextfrei ist. Wir zeigen, dass die Haarnadel Verlängerung einer regulären Sprache, im Gegensatz zur Haarnadel Vervollständigung, eine inhärent mehrdeutige Sprache sein kann. Wir betrachten ebenfalls das Entscheidungsproblem, ob die Haarnadel Verlängerung einer regulären Sprache regulär ist. Allerdings können wir nur beweisen, dass das Problem für den einseitigen Spezialfall entscheidbar ist. (Der einseitige Fall beschreibt die biochemischen Prozesse besser, der beidseitige Fall ist allerdings aus einer theoretischen Sicht interessanter.)

Die begrenzte Haarnadel Vervollständigung kann als eine schwächere Variante der Haarnadel Vervollständigung gesehen werden. Es ist bekannt, dass alle Klassen in der Chomsky Hierarchie unter begrenzter Haarnadel Vervollständigung abgeschlossen sind und dass die kontextfreien, kontextsensitiven und rekursiv aufzählbaren Sprachen unter iterierter, begrenzter Haarnadel Vervollständigung abgeschlossen sind. 2009 wurde die Frage, ob reguläre Sprachen ebenfalls unter iterierter begrenzter Haarnadel Vervollständigung abgeschlossen sind, als offenes Problem gestellt. Wir lösen dieses Problem und zeigen ein allgemeineres

Resultat: Wir geben eine effektive Darstellung der iterierten begrenzten Haarnadel Vervollständigung einer Sprache an, die nur die Operationen Vereinigung, Durchschnitt mit regulären Sprachen und Konkatenation mit regulären Sprachen nutzt. Somit sind alle Sprachklassen, die unter diesen drei elementaren Operationen (effektiv) abgeschlossen sind, ebenfalls unter iterierter, begrenzter Haarnadel Vervollständigung (effektiv) abgeschlossen. Dies trifft auf alle Klassen in der Chomsky Hierarchie und auf alle üblichen Komplexitätsklassen zu. Des weiteren geben wir eine exponentielle untere und obere Schranke für die Größe nichtdeterministischer Automaten an, die die iterierte, begrenzte Haarnadel Vervollständigung einer regulären Sprache akzeptieren.

Es ist bekannt, dass die iterierte (unbegrenzte) Haarnadel Vervollständigung einer regulären Sprache kontextsensitiv und nicht immer kontextfrei ist. In 2008 wurde das offene Problem gestellt, ob die iterierte Haarnadel Vervollständigung einer einenlementigen Sprache (oder einer endlichen Sprache) nicht regulär oder nicht kontextfrei sein kann. Wir lösen dieses Problem, indem wir eine einelementige Sprache angeben, deren iterierte Haarnadel Vervollständigung nicht kontextfrei ist.

Schlagworte: Haarnadel Vervollständigung, Haarnadel Verlängerung, Formale Sprachen, Automatentheorie, Komplexitätstheorie

1 Introduction

The hairpin completion is an operation on formal languages which has been inspired by biochemistry and DNA-computing. In this paper we discuss the hairpin completion from a purely language theoretic point of view. However, let us present the biochemical inspiration of the operation first.

1.1 Biochemical Background

A *single stranded DNA* (or simply a *strand*) is a long polymer that is composed of nucleotides which differ from each other by their bases A (adenine), C (cytosine), G (guanine), and T (thymine). For our purposes a strand can be seen as a finite sequence of bases or a word over the four letter alphabet $\{A, C, G, T\}$. Due to the chemical structure of the polymer, every strand has a so called $5'$ (*five prime*) and a $3'$ (*three prime*) end. This yields an orientation for the base sequences, they may either be denoted in the $5'$-to-$3'$ or $3'$-to-$5'$ orientation. By *Watson-Crick base pairing* the bases A and T (resp. C and G) can connect via *hydrogen bonds*; we say the bases A and T (resp. C and G) are complementary. Two strands with opposite orientation can bond to each other and form the well-known *double helix* if the strands are pairwise complementary. For a graphical example see Figure 1. Throughout the paper, we use the bar-notation for the *Watson-Crick complement* and its language theoretical pendant, we have $A = \overline{T}$ and $C = \overline{G}$. The notation is extended to base sequences in $5'$-to-$3'$ orientation by $\overline{a_1 \cdots a_n} = \overline{a_n} \cdots \overline{a_1}$. This suits the chemical model as $\overline{a_n} \cdots \overline{a_1}$ in $5'$-to-$3'$ orientation equals $\overline{a_1} \cdots \overline{a_n}$ in $3'$-to-$5'$ orientation. From a mathematical point of view this is like taking inverses in groups. Henceforth, all strands are denoted in $5'$-to-$3'$ orientation and the bar-notation implies that the base sequences α and $\overline{\alpha}$ may bond to each other.

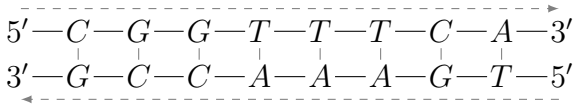


Figure 1: Bonding of two complementary strands

The *polymerase chain reaction* (PCR) is a technique which is often used in DNA-algorithms to amplify the quantity of particular strands within a set of strands. During the PCR the three chemical processes, described below, are repeated several times, see also Figure 2.

Annealing A short strand $\overline{\alpha}$, the *primer*, bonds to the suffix of a long strand $\sigma = \gamma\alpha$, the *template*. (A suffix of a strand is a base sequence preceding the $3'$ end.)

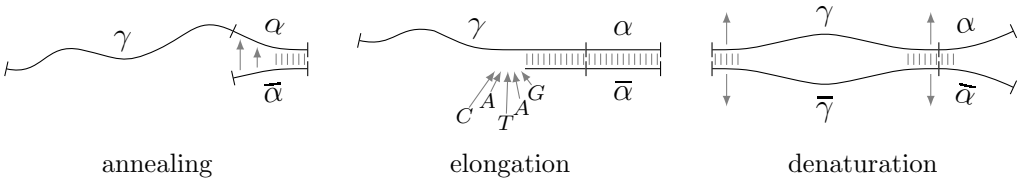


Figure 2: Polymerase chain reaction

Elongation Single nucleotides connect to the unbound γ-part of the template and create the complementary strand $\overline{\alpha}\overline{\gamma} = \overline{\sigma}$ which is bonded to the template σ.

Denaturation The template and its complement are unraveled and we obtain the single strands σ and $\overline{\sigma}$.

By adding primers that may bond to the suffix of the new template $\overline{\sigma}$, it is complemented during the second cycle of the PCR and we obtain a copy of the original strand σ. If we keep on repeating the three steps, the templates and their complements are amplified with exponential speed.

The hairpin completion of a strand can naturally develop during a cycle of the PCR. This is best explained by Figure 3. During annealing, a strand which has the form $\gamma\alpha\beta\overline{\alpha}$ can act as a primer to itself, where the suffix $\overline{\alpha}$ bonds to the infix α, and form an intramolecular base pairing known as *hairpin*. During elongation the γ-part is complemented and after the denaturation process we obtain the new single strand $\gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ which we call a *hairpin completion*.

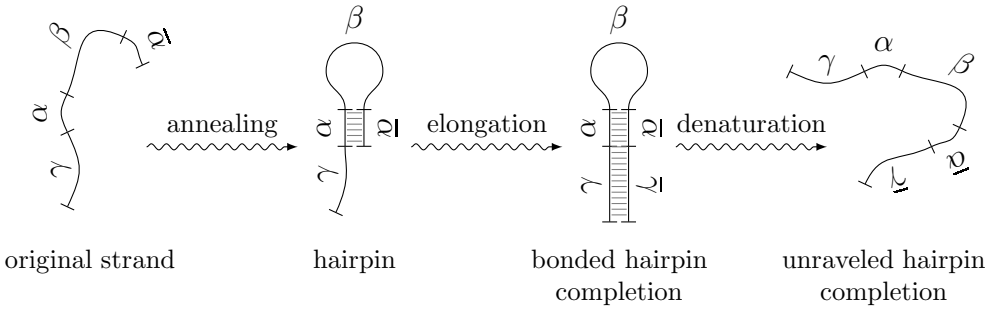


Figure 3: Hairpin completion of a strand

For many DNA algorithms the hairpin completions are by-products and cannot be used in the subsequent computation. Therefore, sets of strands which are unlikely to form hairpins (or lead to other *bad hybridizations*) have been analyzed in various papers, see [7,10,11,18,19] and the references within. However, some DNA algorithms heavily rely on the fact that strands can from hairpins. An example is an algorithm of Sakamoto et al. which was successfully used to solve an instance of the SATISFYABILITY problem [36]. One of

the main concepts of the algorithm is to eliminate all DNA molecules with a hairpin structure. Another example is a technique called *Whiplash PCR* where strands are designed to form a hairpin as in Figure 3 but where only a part of sequence γ is complemented. The length of the complemented part is controlled by so-called *stopper sequences*. This technique can be used to solve combinatorial problems including NP-complete ones like SATISFIABILITY and HAMILTONIAN-PATH, see [14, 37, 42].

1.2 Previous Work

On a more abstract level a strand can be seen as a word and a (possibly infinite) set of strands is a formal language. The hairpin completion has been first defined by Cheptea, Martín-Vide, and Mitrana in 2006 [6]. The hairpin completion and its iterated version have been investigated from language theoretical and algorithmic points of view in a series of papers [6, 25, 26, 28–31]. Besides the hairpin completion we also consider two related operations, the (iterated) bounded hairpin completion and the hairpin lengthening. For the bounded variant we impose a length bound for the γ-part of the hairpin completion. This operation has been defined and investigated in [16, 17]. For the hairpin lengthening we allow that the newly created suffix may only cover a part of the prefix, that means we obtain words $\gamma_1\alpha\beta\overline{\alpha}\overline{\gamma_2}$ where γ_2 is a suffix of γ_1, see [27]. Formal definitions of all operations are given in Section 2.2.

In [6] the closure properties of different language classes under the non-iterated and iterated hairpin completion have been analyzed. It follows that neither the regular nor the context-free languages are closed under hairpin completion whereas the family of context-sensitive languages is closed under this operation. More precisely, the hairpin completion of a regular language is always a linear context-free language. Actually, from [6] we can derive that the class DSPACE(f) (resp. the class NSPACE(f)) is closed under hairpin completion (resp. closed under iterated hairpin completion) for every function $f \in \Omega(\log)$. (By the class DSPACE(f) (resp. NSPACE(f)) we mean, as usual, the class of languages that can be accepted by a deterministic (resp. non-deterministic) Turing machine which uses $f(n)$ work space on input length n.) In particular, the class of context-sensitive languages is closed under iterated hairpin completion, too. Furthermore, if we apply the iterated hairpin completion to a regular (resp. context-free) language we stay inside NL(= NSPACE(log)) (resp. NSPACE($\log^2$), by Lewis, Stearns, and Hartmanis [23]) which is in terms of space complexity far below the class of deterministic context-sensitive languages.

From [27] by Manea, Martín-Vide, and Mitrana we derive that the non-iterated variant of the hairpin lengthening behaves in a similar way. Neither the regular nor the context-free languages are closed under hairpin length-

ening whereas the context-sensitive languages are closed under the operation and by applying the hairpin lengthening to a regular language we obtain a linear context-free language. However, the situation changes if we consider the iterated variant. The context-free and the context-sensitive languages are closed under iterated hairpin lengthening, and it is open whether the regular languages are closed under iterated hairpin lengthening.

The bounded hairpin completion can be seen as a weaker variant of the hairpin completion. All classes in the Chomsky Hierarchy are closed under bounded hairpin completion and the classes of context-free, context-sensitive, and recursively enumerable languages are also closed under the iterated operation, see [16, 17]. But the status for regular languages remained unknown and was stated as an open problem in [17].

1.3 Outline

In this work we discuss some of the problems that have been left open by previous papers. In Section 2 we lay down the notation we use in this paper and give formal definitions of the hairpin formations.

In Section 3 we consider the hairpin completions of regular languages which are known to be linear context-free. As regularity of a linear context-free language is not decidable in general, the question arose whether we can decide regularity of the hairpin completion of a regular language. This question was stated as an open problem in [6]. We solve the question positively and present a detailed discussion on the space and time complexity of the decision algorithm. As a side-product of our technique to solve the problem we prove that the hairpin completion of a regular language is actually an unambiguous linear context-free language. The section covers the results that have been presented at the CIAA 2010 [8]. In addition we present new results concerning the hairpin completions language classes beneath regular languages. We show that there are aperiodic (i. e., star-free) languages whose hairpin completion are non-regular, but if the hairpin completion of an aperiodic language is regular, then it is aperiodic, too. The same is true for the language class induced by the variety **LDA**.

In Section 4 we discuss the problem if it is decidable whether the hairpin lengthening of a regular language is regular. We face a similar problem as we did for the hairpin completion since the hairpin lengthening of a regular language is also linear context-free. For this problem we are only able to provide partial results and we discuss a new difficulty that we encounter in comparison to the hairpin completion. We show that the hairpin lengthening of a regular language may lead to an inherent ambiguous context-free language.

Next we consider the iterated bounded hairpin completion in Section 5. We prove that every language class that is closed under very basic operations

(union, intersection with regular sets, and concatenation with regular sets) is also closed under iterated bounded hairpin completion. This applies to all classes in the Chomsky hierarchy and to all usual complexity classes. In particular, this solves the open problem, stated in [17], whether the regular languages are closed under iterated bounded hairpin completion. Furthermore, for a given non-deterministic finite automaton (NFA) accepting a language L, we give exponential lower and upper bounds for the size of an NFA accepting the iterated bounded hairpin completion of L. Thus, if we ignore constants, the NFA leads us to a linear time membership test for the iterated bounded hairpin completion of a fixed regular language.

Section 6 is devoted to the class of iterated hairpin completions of singletons (HCS). This class has been investigated in [29] and the corresponding journal version [31] by Manea, Mitrana, and Yokomori. HCS is included in the class of context-sensitive languages. However, the questions if HCS contains non-regular or non-context-free languages has been unsolved and has been stated as an open problem in [29]. We answer this question by stating a singleton whose iterated hairpin completion is not context-free.

The results of Section 5 and 6 have been presented as a poster at the DLT 2010 [20] and can also be found in the corresponding journal version [21].

2 Notation and Definitions

We assume the reader to be familiar with the fundamental concepts of automata, language, and complexity theory, see [15, 34]. In this section we fix the notation we use within the rest of the paper and recall some definitions that may be non-standard. In Section 2.2 we state the definitions of the hairpin formations.

2.1 Notation

A finite set of of *terminals* or *letters* is called an *alphabet*, a finite sequence of letters a *word*, and a set of words is a *language*. In this paper we always denote the alphabet by Σ. We denote by Σ^* all *words* over the alphabet Σ, as usual, by 1 we denote the empty word, and we let $\Sigma^+ = \Sigma^* \setminus \{1\}$. Furthermore, we assume that Σ is equipped with an *involution* $\overline{\ }$, i. e., we have $\overline{\overline{a}} = a$ for all $a \in \Sigma$. (In biochemistry $\Sigma = \{A, C, G, T\}$ with $\overline{A} = T$ and $\overline{C} = G$.) We extend the involution to words by $\overline{a_1 \cdots a_n} = \overline{a_n} \cdots \overline{a_1}$. For a language L we let $\overline{L} = \{\overline{w} \mid w \in L\}$. Do not confuse this notation with the set-theoretic complement which we denote by $L^c = \Sigma^* \setminus L$.

Let $w \in \Sigma^*$ be a word. The length of $w = a_1 \cdots a_n$ is denoted by $|w| = n$ (for letters $a_1, \ldots, a_n \in \Sigma$), the number of occurrences of a letter $b \in \Sigma$ in w is denoted by $|w|_b = |\{i \mid a_i = b\}|$, the i-th letter is denoted by $w[i] = a_i$, and

by $w[i,j]$ we mean the sequence $a_i \cdots a_j$ ($w[i,j] = 1$ if $j < i$). If we can write $w = xyz$ for some $x, y, z \in \Sigma^*$, then x, y, and z are called *prefix*, *factor*, and *suffix* of w, respectively. By a *proper prefix* (resp. *proper suffix*) of w we mean a prefix (resp. suffix) x of w such that $x \neq w$ (but we allow $x = 1$). For the prefix relation we also use the notation $x \leq w$ if x is a prefix of w and $x < w$ if x is a proper prefix of w. Note that if z is a suffix of w, then $\overline{z}$ is a prefix of $\overline{w}$ or $\overline{z} \leq \overline{w}$.

A word r is called *primitive*, if there is no word $u \in \Sigma^+$ and $i > 1$ such that $r = u^i$. For every word w there exists a unique primitive word r such that $w = r^i$ for some $i \geq 1$; we say r is the *primitive root* of w.

For a length bound $m \in \mathbb{N}$ we let $\Sigma^{\leq m}$ denote all words whose length is at most m, i. e., $\Sigma^{\leq m} = \bigcup_{i \leq m} \Sigma^i$. Analogously, we define $\Sigma^{<m} = \bigcup_{i<m} \Sigma^i$, $\Sigma^{\geq m} = \bigcup_{i \geq m} \Sigma^i$, and $\Sigma^{>m} = \bigcup_{i>m} \Sigma^i$.

2.1.1 Finite Automata

A common way to describe regular languages are *non-deterministic finite automata* (NFAs). An NFA is a tuple $\mathcal{A} = (\mathcal{Q}, \Sigma, E, \mathcal{I}, \mathcal{F})$ where $\mathcal{Q}$ is the finite set of *states*, $E \subseteq \mathcal{Q} \times \Sigma \times \mathcal{Q}$ is the set of *labeled transitions* or *arcs*, $\mathcal{I} \subseteq \mathcal{Q}$ is the set of *initial states*, and $\mathcal{F} \subseteq \mathcal{Q}$ is the set of final states. If there is a path from a state p to a state q which is labeled by a word w, we write $p \xrightarrow{w} q$. The language that is accepted by $\mathcal{A}$ contains all words that label a path from an initial state to a final state:

$$L(\mathcal{A}) = \left\{ w \in \Sigma^* \;\middle|\; \exists p \in \mathcal{I}\ \exists q \in \mathcal{F} \colon p \xrightarrow{w} q \right\}.$$

In Section 3.2 and 4.1 it will be crucial to use paths which avoid final states. Therefore, we introduce the notation $p \overset{w}{\Longrightarrow} q$. Formally we let $E' = \{(p,a,q) \in E \mid q \notin \mathcal{F}\}$ and we write $p \overset{w}{\Longrightarrow} q$ if there is a path which uses transitions from E' only. Note that we allow $p \in \mathcal{F}$ but on the path we never meet a final state again.

The automaton $\mathcal{A}$ is called *deterministic finite automaton* (DFA) if $|\mathcal{I}| = 1$ and for every $p \in \mathcal{Q}$ and $a \in \Sigma$ there is exactly one arc $(p,a,q) \in E$. In particular, in this paper DFAs are always *complete*. Thus, we can read every word to its end. We also write $p \cdot w = q$ if $p \xrightarrow{w} q$. This yields a totally defined function $\mathcal{Q} \times \Sigma^* \to \mathcal{Q}$. (It defines an action of Σ^* on $\mathcal{Q}$ on the right.)

2.1.2 Grammars and Ambiguity

A grammar is a quadruple $G = (V, \Sigma, P, S)$ where V is a finite set of *non-terminals*, Σ is the finite set of *terminals* (or the alphabet), P is the finite set of *production rules*, and $S \in V$ is the *axiom*. A grammar is called *context-free* if every production rule has the form $A \Longrightarrow \mu$ where $A \in V$ and $\mu \in (V \cup \Sigma)^*$; it is

called *linear (context-free)* if, in addition, μ contains at most one non-terminal. For strings $\mu, \nu \in (V \cup \Sigma)^*$ we write $\mu \Longrightarrow \nu$ if ν can be derived by applying one production rule to a non-terminal in μ. If $\mu = \mu_0 \Longrightarrow \mu_1 \Longrightarrow \cdots \Longrightarrow \mu_m = \nu$ for some $m \geq 0$, we write $\mu \overset{*}{\Longrightarrow} \nu$ and call this a *derivation*. The language generated by the grammar is the set of (terminal) words such that a derivation from the axiom S to the word exists

$$L(G) = \left\{ w \in \Sigma^* \;\middle|\; S \overset{*}{\Longrightarrow} w \right\}.$$

A derivation $\mu_0 \Longrightarrow \mu_1 \Longrightarrow \cdots \Longrightarrow \mu_m$ is called a *left-most derivation* if in every step a production rule is applied to the left-most non-terminal. Every derivation can easily be transformed into a left-most derivation and two derivations are called different if their corresponding left-most derivations differ.

For a word w the *degree of ambiguity* $d_G(w)$ is the number of different derivations $S \overset{*}{\Longrightarrow} w$ in G. (Another way to define the generated language is $L(G) = \{w \in \Sigma^* \mid d_G(w) \geq 1\}$.) The degree of ambiguity of a grammar G is defined as the supremum

$$d(G) = \sup \{ d_G(w) \mid w \in \Sigma^* \}.$$

If $d(G) = \infty$ (i. e., for all $m \in \mathbb{N}$ exists $w \in \Sigma^*$ such that $d_G(w) \geq m$), we say the grammar G has an unbounded degree of ambiguity. If $d(G) \leq 1$ the grammar is called *unambiguous*, otherwise G is called *ambiguous*. A language is *unambiguous* if it can be generated by an unambiguous grammar, and it is *inherent ambiguous* if it can only be generated by ambiguous grammars.

2.1.3 Varieties

In Section 3.3 we investigate the hairpin completions of varieties of formal languages. Here, we give a short introduction on the theory of varieties and introduce the concepts we use later in this paper. For a profound presentation of this rich theory we refer to [35].

Let L be a formal language, M be a monoid, and $h \colon \Sigma^* \to M$ be a morphism. We say h *recognizes* L if there is a subset $N \subseteq M$ such that $L = h^{-1}(N)$. By extension, we also say M recognizes L.

Let L be a formal language. The language L induces an *syntactic congruence* $\sim_L$ over words such that $u \sim_L v$ if and only if for all words x, y we have $xvy \in L \iff xwy \in L$. The *syntactic monoid* of L is the quotient monoid $M(L) = \Sigma^* / \sim_L$. It is obvious that $M(L)$ recognizes L and it is also well-known that $M(L)$ divides every monoid M which recognizes L. Another well-known fact is that L is regular if and only if its syntactic monoid is finite. Let M be a finite monoid. An element e is called an *idempotent* if $e = e^2$.

Every element $s \in M$ generates a unique idempotent $s^\omega = s^{2\omega}$ for some $\omega \geq 1$. By the power s^ω we always denote the idempotent generated by s.

A class $\mathcal{V}$ of finite monoids is called a *variety* if it is closed under division and direct product. Every variety $\mathcal{V}$ induces a *variety of languages* which consists of all languages whose syntactic monoid belongs to $\mathcal{V}$. (Equivalently, a language belongs to the variety induced by $\mathcal{V}$ if it is recognized by some monoid in $\mathcal{V}$.) By definition, all varieties of languages are closed under intersection, union, and (set-theoretic) complement. Some varieties of languages have many different (algebraic, automata-theoretical, combinatorial, and logical) characterizations. We are especially interested in the two varieties **A** and **LDA** which are defined below.

A finite monoid is called *aperiodic* if it satisfies the equation $s^\omega = s^{\omega+1}$ for all $s \in M$. The set of aperiodic monoids forms a variety which is denoted by **A**. We also call a language aperiodic if its syntactic monoid is aperiodic. The variety of languages induced by **A** has a huge number of different characterizations, some of them are star-free regular expressions, counter-free automata, and first-order logic, see e. g., [32, 35, 39]. The star-free regular expressions are regular expressions that do not use the Kleene star, but taking the complement is allowed. Hence, infinite languages like $\Sigma^* = \emptyset^c$ are definable by star-free regular expressions. It follows that aperiodic languages are closed under concatenation.

For a homomorphism $h : \Sigma^* \to M$ the monoid M belongs to **LDA** if it satisfies the equation

$$(esete)^\omega ese(esete)^\omega = (esete)^\omega$$

for all elements $s, t \in M$ and all idempotents $e \in h(\Sigma^+)$. Note that, by choosing $t = 1$ and $e = s^\omega$, we can derive $s^\omega = s^{\omega+1}$ and hence **LDA** is a subclass of **A**. It is known that a language is recognizable by a monoid in **LDA** if and only if it is definable by a sentence in *first-order logic with two variables and successor predicate* ($\mathrm{FO}^2[<, +1]$), see [1, 24] and the references within for a detailed discussion and further characterizations of **LDA**. In this paper we prefer the representation by $\mathrm{FO}^2[<, +1]$-sentences to the representation by monoid equations. A definition of $\mathrm{FO}^2[<, +1]$ is given below.

2.1.4 First-Order Logic over Words

The *atomic formulae* in first-order logic are $\top$ (*true*) and the predicates $\lambda_a(x)$ (position x of a word is labeled by a), and $x < y$ (with the usual meaning) for variables (or positions) x, y and a letter $a \in \Sigma$. A *formula* in first-order logic is a quantified boolean combination of atomic formulae. We use the abbreviations $\bot = \neg\top$ and $(x \leq y) = \neg(y < x)$. The set of formulae in first-order logic is denoted by $\mathrm{FO}[<]$. A formula is a *sentence* if it does not have free variables.

Let $\varphi(x_1, \ldots, x_m)$ be a formula with m free variables, let w be a word, and let $1 \leq \ell_1, \ldots, \ell_m \leq |w|$ denote positions in w. If w satisfies the formula φ with $x_i = \ell_i$ for $1 \leq i \leq m$, we say $w, \ell_1, \ldots, \ell_m$ is a *model* for φ and denote this by $w, \ell_1, \ldots, \ell_m \models \varphi$. Every sentence φ defines a language $L(\varphi) = \{w \in \Sigma^* \mid w \models \varphi\}$.

Two formulae φ and ψ with m free variables each are *equivalent* if for all words w and positions $\ell_1, \ldots, \ell_m$ we have

$$w, \ell_1, \ldots, \ell_m \models \varphi \quad \iff \quad w, \ell_1, \ldots, \ell_m \models \psi.$$

In this case we also write $\varphi \equiv \psi$.

By $\mathrm{FO}^2[<, +1]$ we denote the set of first-order formulae which only use (and reuse) two variables, yet we allow the additional *successor predicate* $x = y + 1$ for variables x, y. It is well-known that three variables already give the expressional power of $\mathrm{FO}[<]$ and that, in this case, the successor predicate is redundant.

For a positions x and a word v we use the abbreviation $\vec{\lambda}_v(x)$ if x and the $|v| - 1$ successive positions are labeled by the letters of v, respectively. Formally, we let

$$\vec{\lambda}_v(x) \;=\; \lambda_{v[1]}(x) \wedge \exists y\big(y = x+1 \wedge \lambda_{v[2]}(y) \wedge \exists x\big(x = y+1 \wedge \lambda_{v[3]}(x) \wedge \exists y(\cdots)\big)\big).$$

Therefore, we may use the abbreviations in $\mathrm{FO}^2[<, +1]$-formulae. Analogously, we use $\overleftarrow{\lambda}_v(x)$ if x and the $|v| - 1$ preceding positions are labeled by the letters of v, respectively.

2.2 The Hairpin Completion and its Variants

If a word w has a factorization $w = \gamma\alpha\beta\overline{\alpha}$, the suffix $\overline{\alpha}$ can bind to the factor α, it can form a hairpin, and we obtain the hairpin completion $\pi = \gamma\alpha\beta\overline{\alpha}\overline{\gamma}$, again see Figure 3. We call π a *right hairpin completion* of w. In biochemistry a hairpin structure is stable only if the factor α is long enough, so we fix a small constant k and ask $|\alpha| = k$. (Note that the definition does not change if we asked $|\alpha| \geq k$.) In this paper by k we always mean this constant. Symmetrically, for $w = \alpha\beta\overline{\alpha}\overline{\gamma}$ the word $\pi = \gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ is a *left hairpin completion* of w.

2.2.1 The Hairpin Completion

Most literature distinguishes three cases of the hairpin completion of a formal language: the *right-sided*, *left-sided*, and *two-sided hairpin completion*. We also refer to the first two cases as the *one-sided hairpin completion*. Here we use a

slightly more general definition from [9] which allows us to treat all cases at once. Let L_1 and L_2 be formal languages, we define the *hairpin completion* as

$$\mathcal{H}_k(L_1, L_2) = \{\gamma\alpha\beta\overline{\alpha}\overline{\gamma} \mid |\alpha| = k \wedge (\gamma\alpha\beta\overline{\alpha} \in L_1 \vee \alpha\beta\overline{\alpha}\overline{\gamma} \in L_2)\},$$

i. e., we join all right hairpin completions of words from L_1 with all left hairpin completions of words from L_2.

For a formal language L, the right-sided (resp. left-sided) hairpin completion is given by $\mathcal{H}_k(L, \emptyset)$ (resp. $\mathcal{H}_k(\emptyset, L)$) and the two-sided hairpin completion is $\mathcal{H}_k(L) = \mathcal{H}_k(L, L)$.

For biochemical applications the right-sided hairpin completion seems to be the most natural case as the elongation (the chemical process where $\overline{\gamma}$ is created) works only in one direction. In some biochemical applications, especially those using PCR, a strand σ and its complement $\overline{\sigma}$ always co-occur, so it is also natural to assume $L = \overline{L}$ and to consider the two-sided hairpin completion $\mathcal{H}_k(L)$. (Note that $\mathcal{H}_k(L) = \mathcal{H}_k(L, \emptyset) \cup \overline{\mathcal{H}_k(L, \emptyset)}$ if $L = \overline{L}$.) However, from a theoretic point of view the two-sided case is interesting, too, as some of the language-theoretic and computational problems become much more challenging, see e. g., Theorem 3.3 or Proposition 3.8.

2.2.2 The (Iterated) Parameterized Hairpin Completion

In order to define the iterated hairpin completion and the (iterated) bounded hairpin completion we will introduce the *(iterated) parameterized hairpin completion* which covers these operations as special cases. We have introduced this variant of the hairpin completion in [21]. Here we only consider one language L and words from L may form left or right hairpin completions, yet we introduce two length bounds $\ell, r \in \mathbb{N} \cup \{\infty\}$. The parameterized hairpin completion covers all words $\gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ that can be derived by a left (resp. right) hairpin completion from a word in L and where the length of the factor γ (chosen as in the definition above) is bounded ℓ (resp. r). We will define the parameterized hairpin completion in a constructive way. For a word $\alpha \in \Sigma^k$ we define

$$\mathcal{H}_\alpha(L, \ell, 0) = \bigcup_{\gamma \in \Sigma^{\leq \ell}} \gamma(L \cap \alpha\Sigma^*\overline{\alpha}\overline{\gamma})$$
$$\mathcal{H}_\alpha(L, 0, r) = \bigcup_{\gamma \in \Sigma^{\leq \ell}} (L \cap \gamma\alpha\Sigma^*\overline{\alpha})\overline{\gamma}$$
$$\mathcal{H}_\alpha(L, \ell, r) = \mathcal{H}_\alpha(L, \ell, 0) \cup \mathcal{H}_\alpha(L, 0, r)$$

and for the constant k we define

$$\mathcal{H}_k(L, \ell, r) = \bigcup_{\alpha \in \Sigma^k} \mathcal{H}_\alpha(L, \ell, r).$$

Note that the one-sided hairpin completions, as defined above, are given by $\mathcal{H}_k(L,\emptyset) = \mathcal{H}_k(L,0,\infty)$ and $\mathcal{H}_k(\emptyset,L) = \mathcal{H}_k(L,\infty,0)$; the two-sided hairpin completion is given by $\mathcal{H}_k(L) = \mathcal{H}_k(L,\infty,\infty)$. The bounded hairpin completion, as it was defined in [16], arises if we choose $\ell = r \in \mathbb{N}$.

By definition it is plain, that if L is regular and the bounds are finite ($\ell, r \in \mathbb{N}$), then the parameterized hairpin completion $\mathcal{H}_k(L,\ell,r)$ is regular. Therefore, the bounded hairpin completion of a regular language is regular, too. This does not necessarily apply if $\ell = \infty$ or $r = \infty$ as we obtain an infinite union.

Now, let us define the iterated variant. For a language L and $\ell, r \in \mathbb{N} \cup \{\infty\}$ we let

$$\mathcal{H}^0_\alpha(L,\ell,r) = L, \qquad \mathcal{H}^i_\alpha(L,\ell,r) = \mathcal{H}_\alpha(\mathcal{H}^{i-1}_\alpha(L,\ell,r),\ell,r),$$
$$\mathcal{H}^0_k(L,\ell,r) = L, \qquad \mathcal{H}^i_k(L,\ell,r) = \mathcal{H}_k(\mathcal{H}^{i-1}_k(L,\ell,r),\ell,r)$$

for $i \geq 1$. The *iterated parameterized hairpin completion* is defined as

$$\mathcal{H}^*_\alpha(L,\ell,r) = \bigcup_{i\geq 0} \mathcal{H}^i_\alpha(L,\ell,r), \qquad \mathcal{H}^*_k(L,\ell,r) = \bigcup_{i\geq 0} \mathcal{H}^i_k(L,\ell,r).$$

In other words, $\mathcal{H}^*_k(L,\ell,r)$ contains all words which belong to a sequence $w_0,\ldots,w_m$ where $w_0 \in L$ and for $1 \leq i \leq m$ either the word w_i is a left hairpin completion of w_{i-1} and $|w_{i-1}| + \ell \geq |w_i|$ or it is a right hairpin completion of w_{i-1} and $|w_{i-1}| + r \geq |w_i|$. In that case we call w_m an (m-)iterated hairpin completion of w_0.

For the iterated (unbounded, two-sided) hairpin completion we also use the notation $\mathcal{H}^*_k(L) = \mathcal{H}^*_k(L,\infty,\infty)$.

Example 2.1. Figure 4 shows a 3-iterated hairpin completion of $\alpha u\overline{\alpha}v\alpha$ where $|\alpha| = k$. In each step the dashed part is the newly created prefix or suffix.

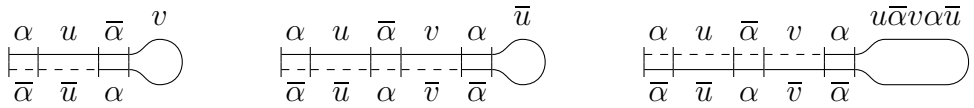


Figure 4: Example for the iterated hairpin completion

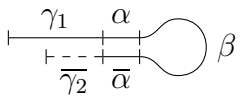
2.2.3 The Hairpin Lengthening

The *hairpin lengthening* is an operation which we suggested as a natural variant of the hairpin completion in [9], back than we called it partial hairpin completion. The operation has been investigated in [27]. The idea is that the

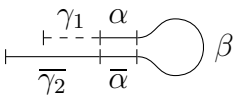
elongation process during the PCR may be interrupted before the whole unbound prefix (or suffix) is complemented, see Figure 5. We define the hairpin lengthening of two languages as L_1 and L_2 as

$$\mathcal{HL}_k(L_1, L_2) = \{\gamma_1\alpha\beta\overline{\alpha}\overline{\gamma_2} \mid |\alpha| = k \wedge ((\gamma_1\alpha\beta\overline{\alpha} \in L_1 \wedge \overline{\gamma_2} \leq \overline{\gamma_1}) \vee (\alpha\beta\overline{\alpha}\overline{\gamma_2} \in L_2 \wedge \overline{\gamma_1} \leq \overline{\gamma_2}))\}.$$

Again, if $L_1 = \emptyset$ or $L_2 = \emptyset$, we call $\mathcal{HL}_k(L_1, L_2)$ a one-sided hairpin lengthening.



right hairpin lengthening



left hairpin lengthening

Figure 5: Hairpin lengthening

3 The Hairpin Completion of Regular Languages

In this section L_1 and L_2 are always regular languages. We investigate the hairpin completion $\mathcal{H}_k(L_1, L_2)$. It is known that $\mathcal{H}_k(L_1, L_2)$ is not necessarily regular but always linear context-free, see [6] or Corollary 3.7.

Example 3.1. Consider $\Sigma = \{a, \overline{a}\}$ and $L_1 = a^*\overline{a}^k$. The (one-sided) hairpin completion

$$\mathcal{H}_k(L_1, \emptyset) = \left\{a^i\overline{a}^j \;\middle|\; i \geq j \geq k\right\}$$

is linear context-free. However, if we choose $L_2 = \overline{L_1} = a^k\overline{a}^*$, the hairpin completion

$$\mathcal{H}_k(L_1, L_2) = \left\{a^i\overline{a}^j \;\middle|\; i, j \geq k\right\}$$

is regular.

As the regularity of (linear) context-free languages is not decidable in general [3, 13], the question arose whether the regularity of the hairpin completion of regular languages is decidable; this was stated as open problem in 2006 [6]. In 2009 we solved this question positively [9] and we provided a polynomial time result (by putting some restrictions on the input). In a second approach [8] we improved our previous results and we deduced that $\mathcal{H}_k(L_1, L_2)$ is actually an unambiguous linear context-free language. The unambiguous context-free languages form a class strictly in between the context-free languages, where the regularity problem is not decidable, and the deterministic context-free languages, where the regularity problem is decidable

by Stearns [38]; yet the best known algorithm (by Valiant [41]) solving the problem runs in double exponential time. To the best of our knowledge the regularity problem for unambiguous context-free languages is open.

In this section we present the results of our CIAA 2010 paper [8]. Let us assume that an DFA that accepts L_1 and an DFA that accepts $\overline{L_2}$ are given. By n we mean the size (or number of states) of the larger DFA.

Theorem 3.2. *The problem whether the hairpin completion $\mathcal{H}_k(L_1, L_2)$ is regular is* NL*-complete.*

Theorem 3.3. *The problem whether the hairpin completion $\mathcal{H}_k(L_1, L_2)$ is regular is decidable in (deterministic) time*

i.) $\mathcal{O}(n^2)$ *if* $L_1 = \emptyset$ *or* $L_2 = \emptyset$.

ii.) $\mathcal{O}(n^6)$ *if* $L_1 = \overline{L_2}$.

iii.) $\mathcal{O}(n^8)$ *in general.*

The proof of both theorems can be found in Section 3.2. Within the proof we also derive that $\mathcal{H}_k(L_1, L_2)$ is an unambiguous linear context-free language, see Corollary 3.7. This allows to compute the growth of the hairpin completion and compare it with the underlying languages L_1 and L_2, see Section 3.2.2.

In Section 3.3 we present new results which concern language classes within the regular languages. We investigate the closure of varieties of languages under the hairpin completion. There are examples of quite *simple* languages that lead to a non-regular hairpin completion, e. g., the languages $L_1 = a^*\overline{a}^k$ and $L_2 = \emptyset$ in Example 3.1. The syntactic monoids of these languages belong to the variety $\mathbf{LDA} \subseteq \mathbf{A}$ (more precisely, we are in the variety $\mathbf{DA}$ as L_1 and L_2 are monomials). However, if we investigate the case where the $M(L_1)$ and $M(L_2)$ belong to $\mathbf{A}$ (resp. $\mathbf{LDA}$) and $\mathcal{H}_k(L_1, L_2)$ is regular, then we deduce that the syntactic monoid of $\mathcal{H}_k(L_1, L_2)$ is in $\mathbf{A}$ (resp. $\mathbf{LDA}$) as well.

We start in Section 3.1 by proving the decidability of the problem whether $\mathcal{H}_k(L_1, L_2)$ is regular without paying attention to efficiency. This first proof is independent from the proof of Theorem 3.2 and 3.3 in Section 3.2. However, it might help to understand the main ideas which are used in both proofs as we do not get too technical. Also, at the end of Section 3.1 we derive a regular representation for $\mathcal{H}_k(L_1, L_2)$ (if it is regular) which is reused in Section 3.3.

3.1 Decidability

In this section we prove:

Proposition 3.4. *It is decidable whether the hairpin completion $\mathcal{H}_k(L_1, L_2)$ is regular.*

We start by providing a linear representation for the hairpin completion $\mathcal{H}_k(L_1, L_2)$. As input we consider a finite monoid M and a morphism $h\colon \Sigma^* \to M$ which recognizes L_1 and L_2. For words u, v we write $u \sim v$ if $h(u) = h(v)$.

Let π be a word in the hairpin completion $\mathcal{H}_k(L_1, L_2)$. By definition there is at least one factorization $\pi = \gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ with $|\alpha| = k$ and $\gamma\alpha\beta\overline{\alpha} \in L_1$ or $\alpha\beta\overline{\alpha}\overline{\gamma} \in L_2$ (or both). We are interested in the unique factorization satisfying in addition:

1. If a prefix of $\gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ belongs to L_1, it is a prefix of $\gamma\alpha\beta\overline{\alpha}$.

2. If a suffix of $\gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ belongs to L_2, it is a suffix of $\alpha\beta\overline{\alpha}\overline{\gamma}$.

In other words, $|\gamma|$ is chosen minimal. For convenience and by symmetry we assume $\gamma\alpha\beta\overline{\alpha} \in L_1$.

We will now define the linear language $L_{\alpha,B,C} \subseteq \mathcal{H}_k(L_1, L_2)$, within a finite set of languages, which includes π. Moreover, for each word $\pi' \in L_{\alpha,B,C}$ the representation yields a factorization as above. The idea is to define two regular languages B and C which consist of words that can replace β and γ, respectively, while preserving all properties of the factorization. Let $B = h^{-1}(h(\beta))$ and C such that $\gamma' \in C$ if and only if

1. $\gamma'\alpha\beta\overline{\alpha}$ is the longest prefix of $\gamma'\alpha\beta\overline{\alpha}\overline{\gamma'}$ which belongs to L_1,

2. if a suffix of $\gamma'\alpha\beta\overline{\alpha}\overline{\gamma'}$ belongs to L_2 it is a suffix of $\alpha\beta\overline{\alpha}\overline{\gamma'}$,

3. $\gamma'\alpha \sim \gamma\alpha$, and $\overline{\alpha}\overline{\gamma'} \sim \overline{\alpha}\overline{\gamma}$.

Condition 1 and 2 ensure that for all $\beta \in B$ and $\gamma \in C$ the word $\gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ has the desired factorization. The purpose of condition 3 will become clear later, but note here that it implies, for $\gamma_1, \gamma_2 \in C$ and $\beta \in B$ the longest prefix of $\gamma_1\alpha\beta\overline{\alpha}\overline{\gamma_2}$ which belongs to L_1 is $\gamma_1\alpha\beta\overline{\alpha}$ and if there is a suffix of $\gamma_1\alpha\beta\overline{\alpha}\overline{\gamma_2}$ which belongs to L_2 it is a suffix of $\alpha\beta\overline{\alpha}\overline{\gamma_2}$.

Also note that B and C are determined by α, $h(\beta)$, $h(\gamma)$, and $h(\overline{\gamma})$, hence there is only a finite set of triples (α, B, C) which satisfies all conditions stated above. Now, let us define

$$L_{\alpha,B,C} = \{\gamma\alpha\beta\overline{\alpha}\overline{\gamma} \mid \beta \in B \wedge \gamma \in C\}\,.$$

In order to prove Theorem 3.4 we proceed as follows: We state two decidable properties for the triple (α, B, C) which are necessary for the regularity of $\mathcal{H}_k(L_1, L_2)$ and we will show that if both properties are satisfied, there exists a regular language R such that $L_{\alpha,B,C} \subseteq R \subseteq \mathcal{H}_k(L_1, L_2)$. Therefore, if all those triples have both properties, then the hairpin completion is a finite union of regular languages and it is regular itself.

3.1.1 Property 1

Note first that if the language C is finite, than $L_{\alpha,B,C}$ is regular. Hence, we will focus on *long words* in C, only. Let n be a fixed constant which is at least k and at least the size of the syntactic monoid of C.

By a pumping argument, for each word $\gamma \in C$ which has at least a length of n there is a factorization $\gamma\alpha = uvw$ such that $|uv| \leq n$, $v \neq 1$, and $uv^i w \in C\alpha$ for all $i \geq 0$.

Let $\gamma\alpha = uvw$ as above, $\beta \in B$, and assume $\mathcal{H}_k(L_1, L_2)$ is regular. There is $s \geq 1$ such that the power v^s is idempotent in the syntactic monoid of $\mathcal{H}_k(L_1, L_2)$. Since the word $uv^s w\beta\overline{w}\overline{v}^s\overline{u}$ belongs to $\mathcal{H}_k(L_1, L_2)$, so does

$$\pi = uv^{st}w\beta\overline{w}\overline{v}^s\overline{u}$$

where $t > |\beta\overline{w}|$. Every suffix of π which belongs to L_2 is a suffix of $\alpha\beta\overline{w}\overline{v}^s\overline{u}$ which is too short to build the hairpin. (A suffix which builds the hairpin has to cover at least half of π.) The longest prefix of π which belongs to L_1 is $uv^{st}w\beta\overline{\alpha}$ and it has to build the hairpin, see Figure 6. The suffix $\overline{w}\overline{v}^s\overline{u}$ is complementary to a prefix of uv^{st} whence $\overline{\overline{w}\overline{v}^s\overline{u}} = uv^s w \leq uv^{st}$, in particular the factor w is a prefix of a power of v.

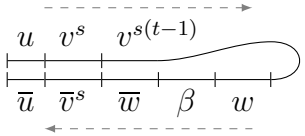


Figure 6: The hairpin of π (Read the upper part from left to right and the lower part from right to left.)

We may rewrite $\gamma\alpha = uvw$ as follows: Let $v = v_1v_2$ such that $vw = v^j v_1$ for some $j \geq 1$ and let $u' = uv_1$ and $v' = v_2v_1$. We have $\gamma\alpha = u'v'^j$ and $u'v'^i \in C\alpha$ for all $i \geq j$. Note that for some $1 \leq m \leq n$ the power v'^m is idempotent in the syntactic monoid of C and, therefore, we have $u'v'^n v'^* \subseteq C\alpha$.

We conclude, if $\mathcal{H}_k(L_1, L_2)$ is regular, the following property is fulfilled:

Property 1: Let V be the (finite and computable) set containing all pairs of words (u, v) which satisfy $|u| \leq n$, $1 \leq |v| \leq n$, and $uv^n v^* \subseteq C\alpha$. The following inclusion holds:

$$(C \cap \Sigma^{\geq n})\alpha \subseteq \bigcup_{(u,v)\in V} uv^+$$

For a regular language C we can decide if it has Property 1, by a simple inclusion test of regular languages.

3.1.2 Property 2

Let $(u, v) \in V$ and $\beta \in B$. We now consider the words

$$\pi_{i,j} = uv^i\beta\overline{v}^j\overline{u}.$$

For $i \geq j \geq n$ the word $\pi_{i,j}$ is included in the hairpin completion $\mathcal{H}_k(L_1, L_2)$ since $uv^i\beta\overline{\alpha}$ belongs to L_1 and α is a suffix v^k. Again, we assume $\mathcal{H}_k(L_1, L_2)$ is regular and we choose $s \geq n$ such that v^s is idempotent in the syntactic monoid of $\mathcal{H}_k(L_1, L_2)$. By pumping, we obtain that $\pi_{i,j} \in \mathcal{H}_k(L_1, L_2)$ for $j > i \geq s$ as well.

Let us factorize $\beta = v^t\beta'$ such that v is no prefix of β' and consider

$$\pi_{s,s+t+1} = uv^{s+t}\beta'\overline{v}^{s+t+1}\overline{u} \in \mathcal{H}_k(L_1, L_2).$$

The longest prefix of $\pi_{s,s+t+1}$ which belongs to L_1 is $uv^{s+t}\beta'\overline{\alpha}$ and it cannot build the hairpin (since v is no prefix of β'). Hence there is a suffix of $\alpha v^t\beta'\overline{v}^{s+t+1}\overline{u}$ which belongs to L_2 and which can build the hairpin. Let us denote this suffix by $\delta\nu\overline{x}\overline{v}^s\overline{u}$ such that $|\delta| = k$ and $v^t\beta'\overline{v}^{t+1} = x\nu\overline{x}$. (Hence δ is the suffix of αx of length k.) As v is no prefix of β', we see that x is a proper prefix of v^{t+1} and we may write $x = v^\ell w$ with $w < v$ and $\ell \leq t$. Moreover, we have $\beta\overline{v} = v^t\beta'\overline{v} = v^\ell w\mu\overline{w}$ for a fitting prefix μ of ν. Note here that $\delta\mu\overline{w}\overline{v}^*\overline{v}^n\overline{u} \subseteq L_2$ as $\overline{v}^i\overline{u} \sim \overline{v}^n\overline{u}$ for $i \geq n$.

We conclude, the following property is satisfied if $\mathcal{H}_k(L_1, L_2)$ is regular:

Property 2: For $w \in \Sigma^*$ let δ_w be the suffix of αw of length k. For every $(u, v) \in V$ and $\beta \in B$ there is a factorization $\beta\overline{v} = v^\ell w\mu\overline{w}$ such that $w < v$, $v^\ell \leq \beta$, and $\delta_w\mu\overline{w}\overline{v}^n\overline{u} \in L_2$.

The correctness of Property 2 follows by the observations above, but decidability is not completely obvious. We may run a tests for each $(u, v) \in V$, but we have to treat all $\beta \in B$ at once. We will reduce Property 2 to a series of inclusion tests of regular languages.

Let $(u, v) \in V$ and define $W = \{w \mid w < v\}$ as the set of proper prefixes of v. Provided that $\beta\overline{v} = v^\ell w\mu\overline{w}$ for some $\beta \in B$ and $w \in W$, then

$$\begin{aligned} v^\ell \leq \beta \quad &\iff \quad \left|v^\ell\right| \leq |\beta| \\ &\iff \quad \left|v^{\ell+1}\right| \leq |\beta\overline{v}| = \left|v^\ell w\mu\overline{w}\right| \\ &\iff \quad |v| - 2\,|w| \leq |\mu|\,. \end{aligned}$$

Therefore, we may define the regular set of valid words μ as

$$M_w = \{\mu \mid \delta_w\mu\overline{w}\overline{v}^n\overline{u} \in L_2 \wedge |\mu| \geq |v| - 2\,|w|\}\,.$$

Property 2 is fulfilled if and only if for each $(u, v) \in V$

$$B\overline{v} \subseteq \bigcup_{w \in W} v^*wM_w\overline{w}.$$

3.1.3 The regular set R

Assume the triple (α, B, C) satisfies Property 1 and 2. We show, there is a regular language R such that $L_{\alpha,B,C} \subseteq R \subseteq \mathcal{H}_k(L_1, L_2)$. Let $\gamma \in C \cap \Sigma^{\geq n^2}$ and $\beta \in B$. By Property 1 there is $(u, v) \in V$ such that $\gamma\alpha = uv^i$ with $i \geq n$. Therefore, $\gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ is included in

$$R_{u,v} = uv^n v^* B \overline{v}^* \overline{v}^n \overline{u}$$

and we claim $R_{u,v} \subseteq \mathcal{H}_k(L_1, L_2)$. Indeed, let $i, j \geq n$ and consider $uv^i\beta\overline{v}^j\overline{u}$. By Property 2 we may factorize $\beta\overline{v} = v^\ell w \mu \overline{w}$ where $w < v$, $\beta = v^\ell\beta'$, and $\delta_w \mu \overline{w}\overline{v}^*\overline{v}^n\overline{u} \subseteq L_2$. If $i + \ell \geq j$, we can use the prefix $uv^{i+\ell}\beta\alpha \in L_1$ to build the hairpin. Otherwise, we can use the suffix $\delta_w\mu\overline{w}\overline{v}^{j-1}\overline{u} \in L_2$ to build the hairpin.

Therefore, we let R be the regular language

$$R = \bigcup_{(u,v)\in V} R_{u,v} \cup \bigcup_{\gamma\in C\cap\Sigma^{<n^2}} \gamma\alpha B\overline{\alpha}\overline{\gamma}$$

and conclude $L_{\alpha,B,C} \subseteq R \subseteq \mathcal{H}_k(L_1, L_2)$ as desired.

3.2 A Decision Algorithm in NL and P

In this section we prove Theorem 3.2 and 3.3. We start with the construction of an NFA $\mathcal{A}$. Most of the further work will be done by investing this automaton. In Section 3.2.2 we deduce a natural representation for the hairpin lengthening and show that it yields an unambiguous linear grammar. Next, in Section 3.2.3, we discuss the characteristics of the one-sided case and why it is easier to solve. In the main part of the proof, Section 3.2.4 and 3.2.5, we investigate the automaton $\mathcal{A}$ and state two properties which are necessary if $\mathcal{H}_k(L_1, L_2)$ is regular. Furthermore, both properties together are sufficient for the regularity of $\mathcal{H}_k(L_1, L_2)$. The properties yield three tests and in the final Sections 3.2.6 and 3.2.7 we will prove the NL and time performance of these tests.

3.2.1 The Automaton $\mathcal{A}$

Let $\mathcal{A}_i = (\mathcal{Q}_i, \Sigma, E_i, \{q_{0i}\}, \mathcal{F}_i)$ for $i = 1, 2$ be two DFAs accepting the languages L_1 and $\overline{L_2}$, respectively, and let $n_i = |\mathcal{Q}_i|$. As input size we consider $n = \max\{n_1, n_2\}$. We also need the set of states of the usual product automaton

$$\mathcal{Q}_{12} = \{(p_1, p_2) \in \mathcal{Q} \mid \exists w \in \Sigma^* \colon q_{01} \cdot w = p_1 \wedge q_{02} \cdot w = p_2\}$$

together with the operation $(p_1, p_2) \cdot w = (p_1 \cdot w, p_2 \cdot w)$ for $(p_1, p_2) \in \mathcal{Q}_{12}$ and $w \in \Sigma^*$. Furthermore, we let $n_{12} = |\mathcal{Q}_{12}|$. Note that if $L_2 = \emptyset$ or $L_1 = \overline{L_2}$, we have $n_{12} = n_1 = n$. Let us recall from Theorem 3.3 that we are able to

provide better time complexities for these two special cases. In general we have $n \leq n_{12} \leq n^2$.

For every quadruple $(p_1, p_2, q_1, q_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \mathcal{Q}_1 \times \mathcal{Q}_2$ we define a regular language

$$B(p_1, p_2, q_1, q_2) = \{w \in \Sigma^* \mid p_1 \cdot w = q_1 \wedge p_2 \cdot \overline{w} = q_2\}.$$

We say that (p_1, p_2, q_1, q_2) is a *basic bridge* if $B(p_1, p_2, q_1, q_2) \neq \emptyset$. The idea behind of this notation is that $B(p_1, p_2, q_1, q_2)$ closes a gap between pairs (p_1, p_2) and (q_1, q_2). Later we choose the β-factors of the hairpin completions from these languages. For a letter $a \in \Sigma$ we call (p_1, p_2, q_1, q_2) an *a-bridge* if $B(p_1, p_2, q_1, q_2) \cap a\Sigma^* \neq \emptyset$.

We also need *levels* for $0 \leq \ell \leq k$, hence there are $k+1$ levels. By $[k]$ we denote in this paper the set $\{0, \ldots, k\}$. Define

$$\{((p_1, p_2), q_1, q_2, \ell) \in \mathcal{Q}_{12} \times \mathcal{Q}_1 \times \mathcal{Q}_2 \times [k] \mid (p_1, p_2, q_1, q_2) \text{ is a basic bridge}\}$$

as the state space of an NFA called $\mathcal{A}$. For $N = n_{12}n_1n_2 \leq n^4$ the size of $\mathcal{A}$ is bounded by $N \cdot (k+1) \in \mathcal{O}(N) \subseteq \mathcal{O}(n^4)$. We have $N \leq n^2$ for $L_1 = \emptyset$ or $L_2 = \emptyset$, and $N \leq n^3$ for $L_2 = \overline{L_1}$.

By a (slight) abuse of languages we call a state $((p_1, p_2), q_1, q_2, \ell)$ a *bridge*, and we keep in mind that there exists a word w such that $p_1 \cdot w = q_1$ and $p_2 \cdot \overline{w} = q_2$. Bridges are frequently denoted by (P, q_1, q_2, ℓ) with $P = (p_1, p_2) \in \mathcal{Q}_{12}$, $q_1 \in \mathcal{Q}_1$, $q_2 \in \mathcal{Q}_2$, and $\ell \in [k]$. Bridges are a central concept in the following.

The a-transitions in the NFA for $a \in \Sigma$ are given by the following arcs:

$$\begin{aligned}
(P,\ q_1 \cdot \overline{a},\ q_2 \cdot \overline{a}, 0) &\xrightarrow{a} (P \cdot a,\ q_1, q_2, 0) && \text{for } q_i \cdot \overline{a} \notin \mathcal{F}_i,\ i = 1, 2,\\
(P,\ q_1 \cdot \overline{a},\ q_2 \cdot \overline{a}, 0) &\xrightarrow{a} (P \cdot a,\ q_1, q_2, 1) && \text{for } q_1 \cdot \overline{a} \in \mathcal{F}_1 \text{ or } q_2 \cdot \overline{a} \in \mathcal{F}_2,\\
(P,\ q_1 \cdot \overline{a},\ q_2 \cdot \overline{a}, \ell) &\xrightarrow{a} (P \cdot a,\ q_1, q_2, \ell + 1) && \text{for } 1 \leq \ell < k.
\end{aligned}$$

Observe that no state of the form $(P, q_1, q_2, 0)$ with $q_1 \in \mathcal{F}_1$ or $q_2 \in \mathcal{F}_2$ has an outgoing arc to level zero; we must switch to level one. There are no outgoing arcs on level k, and for each $(a, P, q_1, q_2, \ell) \in \Sigma \times \mathcal{Q}_{12} \times \mathcal{Q}_1 \times \mathcal{Q}_2 \times [k-1]$ there exists at most one arc $(P, q_1', q_2', \ell) \xrightarrow{a} (P \cdot a, q_1, q_2, \ell')$. Indeed, the triple (q_1', q_2', ℓ') is determined by (q_1, q_2, ℓ) and the letter a. Not all arcs exist because (P, q_1', q_2', ℓ) can be a bridge whereas $(P \cdot a, q_1, q_2, \ell')$ is not. Thus, there are at most $|\Sigma| \cdot N \cdot k \in \mathcal{O}(N)$ arcs in the NFA.

The set of initial states $\mathcal{I}$ contains all bridges of the form $(Q_0, q_1', q_2', 0)$ where $Q_0 = (q_{01}, q_{02})$. The set of final states $\mathcal{F}$ is given by all bridges (P, q_1, q_2, k) on level k.

For an example and a graphical presentation of the NFA, see Figure 7.

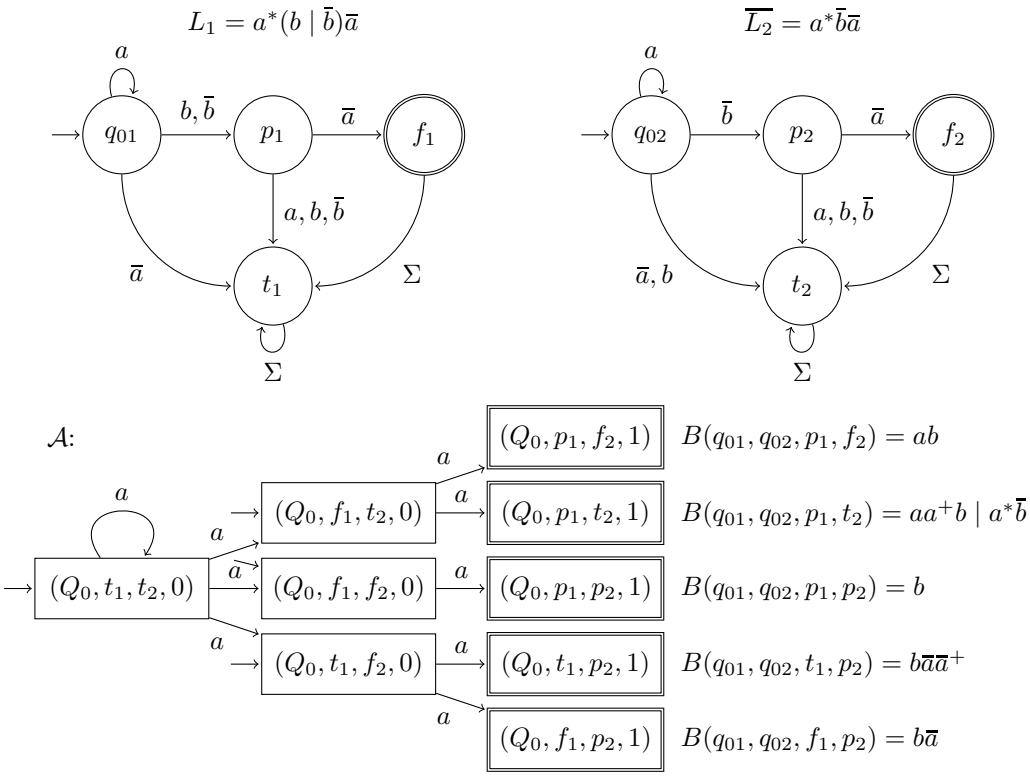


Figure 7: DFAs for L_1 and $\overline{L_2}$ and the resulting NFA $\mathcal{A}$ with 4 initial states and 5 final states associated to the (linear context-free) hairpin completion $\mathcal{H}_k(L_1, L_2) = a^+b\overline{a}^+ \cup \{a^i\overline{b}\overline{a}^j \mid i \geq j \geq 1\}$ with $k = 1$.

3.2.2 Unambiguity and Rational Growth

The next result shows the unambiguity of paths in the automaton $\mathcal{A}$.

Lemma 3.5. *Let $w \in \Sigma^*$ be the label of a path in $\mathcal{A}$ from a bridge $A = (P, p_1, p_2, \ell)$ to $A' = (P', p_1', p_2', \ell')$, then the path is unique. This means that $B = B'$ whenever $w = uv$ and*

$$A \xrightarrow{u} B \xrightarrow{v} A', \qquad A \xrightarrow{u} B' \xrightarrow{v} A'.$$

Proof. It is enough to consider $u = a \in \Sigma$. Let $B = (Q, q_1, q_2, m)$. Then we have $Q = P \cdot a$ and $q_i = p_i' \cdot \overline{v}$. If $\ell = 0$ and $p_i \notin \mathcal{F}_i$ for $i = 1, 2$, then $m = 0$, too; otherwise $m = \ell + 1$. Thus, B is determined by A, A', and u, v. We conclude $B = B'$. □

We will now show that the automaton $\mathcal{A}$ encodes the hairpin completion in a natural way. For languages U and V we define the language V^U as follows:

$$V^U = \{uv\overline{u} \mid u \in U, v \in V\}.$$

Clearly, if U and V are regular, then V^U is linear context-free, but not regular, in general. (The notation V^U is adopted from group theory where exponentiation denotes conjugation and the canonical involution refers to taking inverses.)

Lemma 3.6. *For each pair $\tau = (I, F) \in \mathcal{I} \times \mathcal{F}$ with $F = ((d_1, d_2), e_1, e_2, k)$ let R_τ be the (regular) set of words which label a path from the initial bridge I to the final bridge F, and let $B_\tau = B(d_1, d_2, e_1, e_2)$.*

The hairpin completion $\mathcal{H}_k(L_1, L_2)$ is a disjoint union

$$\mathcal{H}_k(L_1, L_2) = \bigcup_{\tau \in \mathcal{I} \times \mathcal{F}} B_\tau^{R_\tau}.$$

Moreover, for each word $\pi \in B_\tau^{R_\tau}$ there is a unique factorization $\pi = \rho\beta\overline{\rho}$ with $\rho \in R_\tau$ and $\beta \in B_\tau$.

Proof. Let $\pi \in \mathcal{H}_k(L_1, L_2)$. There exists some factorization $\pi = \gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ such that $|\alpha| = k$ and there are runs as in Figure 8 in the original DFAs $\mathcal{A}_1$ and $\mathcal{A}_2$ where $f_1' \in \mathcal{F}_1$ or $f_2' \in \mathcal{F}_2$ (or both):

$$\begin{aligned} L_1 &: \quad q_{01} \xrightarrow{\gamma} c_1' \xrightarrow{\alpha} d_1' \xrightarrow{\beta} e_1' \xrightarrow{\overline{\alpha}} f_1' \xrightarrow{\overline{\gamma}} q_1', \\ \overline{L_2} &: \quad q_{02} \xrightarrow{\gamma} c_2' \xrightarrow{\alpha} d_2' \xrightarrow{\overline{\beta}} e_2' \xrightarrow{\overline{\alpha}} f_2' \xrightarrow{\overline{\gamma}} q_2' \end{aligned}$$

Figure 8: Some run defined by $\pi \in \mathcal{H}_k(L_1, L_2)$

Choosing among all these runs the length $|\overline{\gamma}|$ to be minimal, we see that we actually find the following picture according to Figure 9. In other words, either $\gamma\alpha\beta\overline{\alpha}$ is the longest prefix of π belonging to L_1 or $\alpha\beta\overline{\alpha}\overline{\gamma}$ is the longest suffix of π belonging to L_2. The difference to the precedent figure is that between f_i and q_i' for $i = 1, 2$ we never enter a final state.

$$\begin{aligned} L_1 &: \quad q_{01} \xrightarrow{\gamma} c_1 \xrightarrow{\alpha} d_1 \xrightarrow{\beta} e_1 \xrightarrow{\overline{\alpha}} f_1 \xRightarrow{\overline{\gamma}} q_1', \\ \overline{L_2} &: \quad q_{02} \xrightarrow{\gamma} c_2 \xrightarrow{\alpha} d_2 \xrightarrow{\overline{\beta}} e_2 \xrightarrow{\overline{\alpha}} f_2 \xRightarrow{\overline{\gamma}} q_2' \end{aligned}$$

Figure 9: The unique run defined by $\pi \in \mathcal{H}_k(L_1, L_2)$ with $|\overline{\gamma}|$ minimal

By the definition of the NFA $\mathcal{A}$, we see that $\rho = \gamma\alpha$ is the unique prefix of π such that $\pi = \rho\beta\overline{\rho}$ with $\rho \in R_\tau$ and $\beta \in B_\tau$ for some τ. Now, as the length $|\overline{\gamma}|$ is fixed by π, we see that all states c_i, d_i, e_i, f_i, and q_i' are uniquely defined by π for $i = 1, 2$. Thus, there is a unique $\tau \in \mathcal{I} \times \mathcal{F}$ with $\pi \in B_\tau^{R_\tau}$. More precisely, we have:

$$\tau = ((Q_0, q_1', q_2', 0),\ ((d_1, d_2), e_1, e_2, k)).$$ □

It was shown in [6] that $\mathcal{H}_k(L_1, L_2)$ is an linear context-free language. With the observations made above we find that $\mathcal{H}_k(L_1, L_2)$ is actually an unambiguous linear context-free language, and hence its growth is a rational function. The *growth* or *generating function* g_L of a formal language L is defined as:

$$g_L(z) = \sum_{m \geq 0} \left| L \cap \Sigma^{\leq m} \right| z^m.$$

We can view g_L as a formal power series or as an analytic function in one complex variable where the radius of convergence is strictly positive. The radius of convergence is at least $1/|\Sigma|$.

It is well-known that the growth of a regular language L is effectively rational, i. e., a quotient of two polynomials. The same is true for unambiguous linear context-free languages. In particular, the growth is either polynomial or exponential. If the growth is exponential, we find an algebraic number $r \in \mathbb{R}$ such that $\left| L \cap \Sigma^{\leq m} \right|$ behaves essentially as r^m, see [4, 5, 12, 22].

Corollary 3.7. *The hairpin completion $\mathcal{H}_k(L_1, L_2)$ is an unambiguous linear context-free language and it has a rational growth function. The growth can be directly calculated by the growth of the regular languages R_τ and B_τ.*

3.2.3 The One-sided Case

Next we consider the case where L_1 or L_2 is finite which also covers the one-sided case. For this case we are able provide a simple necessary and sufficient condition for the regularity of $\mathcal{H}_k(L_1, L_2)$.

Proposition 3.8.

i.) If the language $L(\mathcal{A})$ is finite, then $\mathcal{H}_k(L_1, L_2)$ is regular.

ii.) If the language $L(\mathcal{A})$ is infinite and either L_1 is finite or L_2 is finite, then $\mathcal{H}_k(L_1, L_2)$ is not regular.

Proof. Statement *i.)* follows directly by Lemma 3.6.

For *ii.)* let $L(\mathcal{A})$ be infinite. We find a path

$$I \xrightarrow{u} A \xrightarrow{v} A \xrightarrow{w} F$$

in $\mathcal{A}$ where I is an initial bridge, $F = ((d_1, d_2), e_1, e_2)$ is a final bridge, and $A \xrightarrow{v} A$ is a non-trivial loop. Note that A is on level 0 and hence $|w| \geq k$. Let α be the suffix of w of length k and let β be a word from the set $B(d_1, d_2, e_1, e_2)$. We have $\pi_i = uv^i w \beta \overline{w} \overline{v}^i \overline{u} \in \mathcal{H}_k(L_1, L_2)$ for all $i \geq 0$. Moreover, if a prefix of π_i belongs to L_1, it is a prefix of $uv^i w \beta \overline{\alpha}$ and if a suffix of π_i belongs to L_2, it is a suffix of $\alpha \beta \overline{w} \overline{v}^i \overline{u}$.

By contradiction, assume $\mathcal{H}_k(L_1, L_2)$ is regular and L_1 is finite. Let $s \geq 1$ such that the power v^s is idempotent in the syntactic monoid of $\mathcal{H}_k(L_1, L_2)$, hence

$$\pi = uv^{st}w\beta\overline{w}\overline{v}^s\overline{u} \in \mathcal{H}_k(L_1, L_2)$$

where t is huge. In particular we need that π is at least twice as long as the longest word in L_1 and that v^{st} covers more than half of π. The longest suffix of π that belongs to L_2 is still a suffix of $\alpha\beta\overline{w}\overline{v}^s\overline{u}$ which is far too short to build the hairpin; hence a prefix from L_1 has to build the hairpin and it has to cover more than half of π — a contradiction. By a symmetric argument L_2 is infinite, too. □

We check this property. Although, strictly speaking, Test 0 is redundant for the general case:

Test 0: Decide whether or not $L(\mathcal{A})$ is finite. If it is finite, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is regular. If it is not finite but L_1 or L_2 is finite, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

3.2.4 Test 1

For convenience we may assume in the following that $\mathcal{A}$ accepts an infinite language and that all bridges are reachable from an initial bridge and lead to some final bridge.

Let K be the set of non-trivial strongly connected components of the automaton $\mathcal{A}$ (read as a directed graph). Every non-trivial strongly connected component is on level 0 and, moreover, as $\mathcal{A}$ accepts an infinite language, there is at least one. For $\kappa \in K$ let N_κ be the number of states in the component κ. Note that $\sum_{\kappa \in K} N_\kappa \leq N$. By putting some linear order on the set of bridges, we assign to each $\kappa \in K$ the least bridge A_κ and some shortest, non-empty word v_κ such that $A_\kappa \xrightarrow{v_\kappa} A_\kappa$.

The next lemma tells us that for a regular hairpin completion $\mathcal{H}_k(L_1, L_2)$ every strongly connected component $\kappa \in K$ is a simple cycle and hence the word v_κ is uniquely defined.

Lemma 3.9. *Let the hairpin completion $\mathcal{H}_k(L_1, L_2)$ be regular, $\kappa \in K$ be a strongly connected component, and $A_\kappa \xrightarrow{w} F$ be a path from A_κ to a final bridge F. Then the word w is a prefix of some word in v_κ^+.*

In addition, the word v_κ is uniquely defined and the loop $A_\kappa \xrightarrow{v_\kappa} A_\kappa$ visits every other bridge $B \in \kappa$ exactly once. Thus it forms a Hamiltonian cycle of κ and $|v_\kappa| = N_\kappa$.

Proof. Let $A = A_\kappa$ and $v = v_\kappa$. Consider a path labeled by w from A to a final bridge $F = ((d_1, d_2), e_1, e_2, k)$. As all bridges are reachable, we find a word u

and an initial bridge I such that

$$I \xrightarrow{u} A \xrightarrow{v} A \xrightarrow{w} F$$

and the automaton $\mathcal{A}$ accepts uv^iw for all $i \geq 0$. We see next that $uv^iw\beta\overline{w}\overline{v}^i\overline{u} \in \mathcal{H}_k(L_1, L_2)$ for all $i \geq 0$ and all $\beta \in B(d_1, d_2, e_1, e_2)$. As $\mathcal{H}_k(L_1, L_2)$ is regular, there are $s \geq 1$ and $t > |w\beta|$ such that $uv^{st}w\beta\overline{w}\overline{v}^s\overline{u} \in \mathcal{H}_k(L_1, L_2)$, by pumping. This means that the hairpin completion is forced to use a prefix in L_1, because the longest suffix belonging to L_2 is too short to create the hairpin completion. Due to the definition of $\mathcal{A}$, we conclude that uv^sw must be a prefix of $uv^{st}w$. This implies that w is a prefix of a power of v and thus the first statement of our lemma.

Recall that $A \xrightarrow{v} A$ is a shortest, non-trivial loop around A hence $|v| \leq N_\kappa$ is obvious. Let $B \in \kappa \setminus \{A\}$ and $x = x_1x_2$ such that $A \xrightarrow{x_1} B \xrightarrow{x_2} A$. For some $i, j \geq 1$ we have $|v^i| = |x^j|$. Thus, $v^i = x^j$ by the first statement. By the unique-path-property stated in Lemma 3.5 we obtain that the loop $A \xrightarrow{x^j} A$ just uses the shortest loop $A \xrightarrow{v} A$ several times. In particular, B is on the shortest loop around A. This yields $|v| \geq N_\kappa$ and hence the second statement. □

Example 3.10. In the example given in Figure 7 the state $(Q_0, t_1, t_2, 0)$ forms the only strongly connected component and the corresponding path is labeled with a. As one can easily observe, the automaton $\mathcal{A}$ satisfies the properties stated in Lemma 3.9 (even though the hairpin completion is not regular).

The next test tries to falsify the property of Lemma 3.9. Hence it gives a sufficient condition that $\mathcal{H}_k(L_1, L_2)$ is not regular.

Test 1: Decide whether there is $\kappa \in K$ and a path $A_\kappa \xrightarrow{w} F$ such that w is not a prefix of a word in v_κ^+. If there is such a path, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

3.2.5 Test 2 and 3

Henceforth we may assume that Test 1 was successful. We fix a strongly connected component $\kappa \in K$ of $\mathcal{A}$. We let $A = A_\kappa = ((p_1, p_2), q_1, q_2, 0)$ and $v = v_\kappa$ as above and we assume $A \xrightarrow{v} A$ forms an Hamiltonian cycle in κ. By u we denote some word leading from an initial bridge $((q_{01}, q_{02}), q_1', q_2', 0)$ to A. (For the following test we do not need to know u we just need to know it exists.) The main idea is to investigate runs through the DFAs for L_1 and $\overline{L_2}$ where $s, t \geq n$ according to Figure 10.

We investigate the case where $uv^sxyz\overline{x}\overline{v}^t\overline{u} \in \mathcal{H}_k(L_1, L_2)$ for all $s \geq t$ and where (by symmetry) this property is due to the longest prefix belonging to L_1.

$$L_1: \quad q_{01} \xrightarrow{u} p_1 \xrightarrow{v^s} p_1 \xrightarrow{xy} c_1 \xrightarrow{z} d_1 \xrightarrow{\overline{x}} e_1 \xRightarrow{\overline{v}^{n_1}} q_1 \xRightarrow{\overline{v}^*} q_1 \xRightarrow{\overline{u}} q_1'$$

$$\overline{L_2}: \quad q_{02} \xrightarrow{u} p_2 \xrightarrow{v^t} p_2 \xrightarrow{x} c_2 \xrightarrow{\overline{z}} d_2 \xrightarrow{\overline{y}\overline{x}} e_2 \xRightarrow{\overline{v}^{n_2}} q_2 \xRightarrow{\overline{v}^*} q_2 \xRightarrow{\overline{u}} q_2'$$

Figure 10: Runs through $\mathcal{A}_1$ and $\mathcal{A}_2$ based on the loop $A \xrightarrow{v} A$

The following lemma is rather technical. However, the notations are chosen to fit exactly to Figure 10.

Lemma 3.11. *Let $x, y, z \in \Sigma^*$ be words and $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ with the following properties:*

1. *$k \leq |x| < |v| + k$ and x is a prefix of some word in v^+.*
2. *$0 \leq |y| < |v|$ and xy is the longest common prefix of xyz and some word in v^+.*
3. *$z \in B(c_1, c_2, d_1, d_2)$, where $c_1 = p_1 \cdot xy$ and $c_2 = p_2 \cdot x$.*
4. *$q_1 = d_1 \cdot \overline{x}\overline{v}^{n_1}$ and during the computation of $d_1 \cdot \overline{x}\overline{v}^{n_1}$ we see after exactly k steps a final state in $\mathcal{F}_1$ and then never again.*
5. *$q_2 = d_2 \cdot \overline{y}\overline{x}\overline{v}^{n_2}$ and, let $e_2 = d_2 \cdot \overline{y}\overline{x}$, during the computation of $e_2 \cdot \overline{v}^{n_2}$ we do not see a final state in $\mathcal{F}_2$.*

If $\mathcal{H}_k(L_1, L_2)$ is regular, then there exists a factorization $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ where $|\delta| = k$ and $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \in \mathcal{F}_2$ (which implies $\delta\beta\overline{\delta}\overline{\mu}\overline{v}^*\overline{u} \subseteq L_2$).

Proof. The conditions say that $uv^s xyz\overline{x}\overline{v}^t\overline{u} \in \mathcal{H}_k(L_1, L_2)$ for all $s \geq t \geq n$. Moreover, by condition 4 the hairpin completion can be achieved with a prefix in L_1, and the longest prefix of $uv^s xyz\overline{x}\overline{v}^t\overline{u}$ belonging to L_1 is the prefix $uv^s xyz\overline{\alpha}$ where $\overline{\alpha}$ is the prefix of $\overline{x}$ of length k.

If $\mathcal{H}_k(L_1, L_2)$ is regular, then we have $uv^s xyz\overline{x}\overline{v}^{s+1}\overline{u} \in \mathcal{H}_k(L_1, L_2)$, too, as soon as s is large enough, by a simple pumping argument. For this hairpin completion we must use a suffix belonging to L_2. For $z = 1$ this follows from $|y| < |v|$. For $z \neq 1$ we use $|y| < |v|$ and, in addition, that xya with $a = z[1]$ is not a prefix of vx by condition 2.

By 5 the longest suffix of $uv^s xyz\overline{x}\overline{v}^{s+1}\overline{u}$ belonging to L_2 is a suffix of $xyz\overline{x}\overline{v}^{s+1}\overline{u}$. Thus, we can write

$$uv^s xyz\overline{x}\overline{v}^{s+1}\overline{u} = uv^s xyz\overline{x}\overline{v}\overline{v}^s\overline{u} = uv^s\mu\delta\beta\overline{\delta}\overline{\mu}\overline{v}^s\overline{u}$$

where $\delta\beta\overline{\delta}\overline{\mu}\overline{v}^s\overline{u} \in L_2$ and $|\delta| = k$. We obtain $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$. (Recall that our second DFA $\mathcal{A}_2$ accepts $\overline{L_2}$.) As $p_2 = q_{02} \cdot u$ and $p_2 = p_2 \cdot v$, we conclude as desired $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \in \mathcal{F}_2$. □

Example 3.12. Let us take a look at Figure 7 again. Let $A = (Q_0, t_1, t_2, 0)$, $v = a$ and $u = 1$. If we choose $x = a$, $y = 1$ and $z = \overline{b}$ and $(d_1, d_2) = (p_1, p_2)$ we can see, that conditions 1 to 4 of Lemma 3.11 are satisfied but there is no factorization $a\overline{b}\overline{a}\overline{a} = \mu\delta\beta\overline{\delta}\overline{\mu}$ with $|\delta| = k$ such that $q_{02} \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$. Hence, the hairpin completion is not regular.

We perform Test 2 and 3 which, again, try to falsify the property given by Lemma 3.11 for a regular hairpin completion. The tests distinguish whether the word z is empty or non-empty.

Test 2: Decide the existence of words $x, y \in \Sigma^*$ and states $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ satisfying conditions 1 to 5 of Lemma 3.11 with $z = 1$, but where for all factorizations $xy\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ we have $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$. If we find such a situation, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

Test 3: Decide the existence of words $x, y, z \in \Sigma^*$ with $z \neq 1$ and states $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ satisfying conditions 1 to 5 of Lemma 3.11, but where for all factorizations $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ we have $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$. If we find such a situation, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

The following lemma states that if the hairpin completion passes Test 1, 2, and 3, then it is indeed a regular language. Thus, the properties given by Lemma 3.9 and 3.11 together are sufficient for the regularity of $\mathcal{H}_k(L_1, L_2)$.

Lemma 3.13. *Suppose no outcome of Tests 1, 2, and 3 is that $\mathcal{H}_k(L_1, L_2)$ is not regular. Then the hairpin completion $\mathcal{H}_k(L_1, L_2)$ is regular.*

Proof. Let $\pi \in \mathcal{H}_k(L_1, L_2)$. Write $\pi = \gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ with $|\gamma|$ minimal such that either $\gamma\alpha\beta\overline{\alpha} \in L_1$ or $\alpha\beta\overline{\alpha}\overline{\gamma} \in L_2$. By symmetry, we assume $\gamma\alpha\beta\overline{\alpha} \in L_1$. We may assume that $|\gamma| > n^4$ (cf. Proposition 3.8 and Test 0). We can factorize $\gamma = uvw$ with $|uv| \leq n^4$ and $|v| \geq 1$ such that there are runs as in Figure 11 where $f_1 \in \mathcal{F}_1$.

$$
\begin{aligned}
L_1 :&\quad q_{01} \xrightarrow{u} p_1 \xrightarrow{v} p_1 \xrightarrow{w\alpha\beta\overline{\alpha}} f_1 \xRightarrow{\overline{w}} q_1 \xRightarrow{\overline{v}} q_1 \xRightarrow{\overline{u}} q_1' \\
\overline{L_2} :&\quad q_{02} \xrightarrow{u} p_2 \xrightarrow{v} p_2 \xrightarrow{w\alpha\overline{\beta}\overline{\alpha}} f_2 \xRightarrow{\overline{w}} q_2 \xRightarrow{\overline{v}} q_2 \xRightarrow{\overline{u}} q_2'
\end{aligned}
$$

Figure 11: Runs through $\mathcal{A}_1$ and $\mathcal{A}_2$ for the word π

We infer from Test 1 that $w\alpha$ is a prefix of some word in v^+. Hence, we can write $w\alpha\beta = v^m xyz$ with $m \geq 0$ such that $v^m xy$ is the maximal common prefix of $w\alpha\beta$ and some word in v^+, $w\alpha \in v^* x$ with $k \leq |x| < |v| + k$, and $|y| < |v|$.

We see that for some $s \geq t \geq 0$ we can write

$$\pi = uv^s xyz\overline{x}\overline{v}^t\overline{u}.$$

Moreover, $uv^s xyz\overline{x}\overline{v}^t\overline{u} \in \mathcal{H}_k(L_1, L_2)$ for all $s \geq t \geq 0$. There are only finitely many choices for u, v, x, y (due to the lengths bounds) and for each of them there is a regular set R_z associated to the finite collection of bridges such that

$$\pi \in \left\{uv^s xyR_z\overline{x}\overline{v}^t\overline{u} \;\middle|\; s \geq t \geq 0\right\} \subseteq \mathcal{H}_k(L_1, L_2).$$

More precisely, we can choose $R_z = \{1\}$ for $z = 1$ and otherwise we can choose

$$R_z \in \{B(c_1, c_2, d_1, d_2) \cap a\Sigma^* \mid (c_1, c_2, d_1, d_2) \text{ is a bridge and } a \in \Sigma\}.$$

Note that the sets $\{uv^s xyR_z\overline{x}\overline{v}^t\overline{u} \mid s \geq t \geq 0\}$ are not regular in general. If we bound however t by n, then the finite union

$$\bigcup_{0 \leq t \leq n} \left\{uv^s xyR_z\overline{x}\overline{v}^t\overline{u} \;\middle|\; s \geq t\right\}$$

is regular. Thus, we may assume that $t > n$. Let $e_2 = p_2 \cdot x\overline{z}\overline{y}\overline{x}$. We have $e_2 \cdot \overline{v}^n = q_2$ and if we see a final state during the computation of $e_2 \cdot \overline{v}^n$, then for all $t > s \geq n$ and $z \in R_z$ we see that $uv^s xyz\overline{x}\overline{v}^t\overline{u} \in \mathcal{H}_k(L_1, L_2)$, due to a suffix in L_2 and $uv^n v^+ xyR_z\overline{x}\overline{v}^+\overline{v}^n\overline{u} \subseteq \mathcal{H}_k(L_1, L_2)$.

Otherwise, Test 2 or 3 tells us that for all $z \in R_z$ the word $xyz\overline{x}\overline{v}$ has a factorization $\mu\delta\nu\overline{\delta}\overline{\mu}$ such that $|\delta| = k$ and $p_2 \cdot \mu\delta\overline{\nu}\overline{\delta} \in \mathcal{F}_2$. The paths $q_{02} \cdot u = p_2$ and $p_2 \cdot v = p_2$ yield $\delta\nu\overline{\delta}\overline{\mu}\overline{v}^*\overline{u} \subseteq L_2$ and, again, $uv^n v^+ xyR_z\overline{x}\overline{v}^+\overline{v}^n\overline{u} \subseteq \mathcal{H}_k(L_1, L_2)$.

Hence, the hairpin completion $\mathcal{H}_k(L_1, L_2)$ is a finite union of regular languages and, therefore, regular itself. □

3.2.6 Non-deterministic log-space

We now turn to the proof of Theorem 3.2. The NL-hardness is immediate:

Lemma 3.14. *The problem whether the hairpin completion $\mathcal{H}_k(L_1, L_2)$ is regular is* NL*-hard, even for $L_2 = \emptyset$.*

Proof. The well-known NL-complete problem Graph-Reachability [34] can easily be reduced to the following problem for DFAs:

Let $\Sigma = \{a, \overline{a}, b, \overline{b}\}$ be an alphabet with four letters. Decide for a given DFA which accepts a language $L \subseteq \{b, \overline{b}\}^*$ whether or not L is empty.

Now let $L_1 = a^* L \overline{a}^k$. The hairpin completion

$$\mathcal{H}_k(L_1, \emptyset) = \left\{a^i w \overline{a}^j \;\middle|\; i \geq j \geq k \wedge w \in L\right\}$$

is regular if and only if L is empty (because $L \subseteq \{b, \overline{b}\}^*$). □

In the rest of this section we show that Test 1, 2, and 3 can be performed in NL. We will heavily rely on the fact that NL is closed under complement and that graph reachability is solvable in NL, see [34]. Furthermore, we use *single-valued non-deterministic* log-*space transducers* which have been introduced in [2]. For the sake of self-containment, we introduce the concept here, too

A single-valued non-deterministic log-space transducer (NL-transducer) is a non-deterministic Turing machine which works in logarithmic space and which may stop on every input w with some output $r(w)$. Single-valued means that, in case that the machine stops on input w, the output is always the same, independently of non-deterministic moves during the computation. Thus, $w \mapsto r(w)$ is a well-defined partial function from words to words. An NL-transducer is an NL-*reduction* from a language L to L' if we have $w \in L \iff r(w) \in L'$.

The following lemma belongs to folklore. Its proof is exactly the same as for the standard case of deterministic log-space reductions [15] and therefore omitted.

Lemma 3.15. *Let $L' \in$ NL and assume that there exists an* NL*-reduction from L to L'. Then we have $L \in$ NL, too.*

Due to Lemma 3.15 we are free to use several NL-transducers in order to enrich the input. Our first NL-transducer generates a the automaton $\mathcal{A}$.

Lemma 3.16. *There is an* NL*-transducer which outputs the automaton $\mathcal{A}$.*

Proof. Recall that a quadruple $(p_1, p_2, q_1, q_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \mathcal{Q}_1 \times \mathcal{Q}_2$ is a basic bridge if there is a path $p_1 \xrightarrow{w} q_1$ in $\mathcal{A}_1$ and a path $p_2 \xrightarrow{\overline{w}} q_2$ in $\mathcal{A}_2$. In order to test whether these paths exist we have to run a reachability test from (p_1, q_2) to (q_1, p_2) which uses in each step an a-transition in $\mathcal{A}_1$ and simultaneously uses a backwards $\overline{a}$-transition in $\mathcal{A}_2$ for some $a \in \Sigma$. Hence, as graph reachability and its complement are in NL, we can decide in NL whether a quadruple (p_1, p_2, q_1, q_2) is a basic bridge. Therefore, we can output a list of all states and transitions of $\mathcal{A}$ by an NL-transducer. □

Henceforth, by Lemma 3.16, we may assume that $\mathcal{A}$ is written on the input tape. Whenever it is convenient, we may also assume that each bridge is reachable by an initial bridge and leads to some final bridge. Again, this assumption is due to the NL performance of graph reachability. Next we prove that Test 1 can be performed in NL.

Test 1: Decide whether there is $\kappa \in K$ and a path $A_\kappa \xrightarrow{w} F$ such that w is not a prefix of a word in v_κ^+. If there is such a path, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

For the NL algorithm there is no need to compute the set K of $\mathcal{A}$ (even though this is possible with an NL-transducer). We simply guess a bridge which lies in a strongly connected component.

Lemma 3.17. *Test 1 can be performed in* NL.

Proof. We perform a slightly modified test, which fails (i. e., stops with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular) if and only if Test 1 fails. We guess a bridge A, a letter a, and position $1 \leq m \leq n^4$ and check the existence of the two paths:

1. $A \xrightarrow{v} A$ where $m \leq |v| \leq n^4$, $v[m] = a$ and A does not occur in the middle of this path.
2. $A \xrightarrow{w} F$ where $w[i \cdot |v| + m] \neq a$ for some $i \in \mathbb{N}$ and some bridge F.

If the paths exist, we stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

Note that the existence of both paths can be checked without knowledge of v and w. It is enough to remember the triple (A, a, m) and store some integers which are at most n^4. (For path 2 we count the length of the path modulo $|v|$.) This yields the NL performance.

The only difference between Test 1 and the modified version is, that we do not necessarily use a shortest loop around A. Thus, if Test 1 fails, then paths 1 and 2 exist and the modified test fails, too. Vice versa, if Test 1 does not fail, then the strongly connected component including A is a simple cycle, by Lemma 3.9, and hence path 1 is determined as the shortest, non-trivial loop around A and we will not find a path of the form 2. □

As NL is closed under complement, we may now assume that Test 1 did not fail and, by Lemma 3.9, for every bridge A within a strongly connected component, there is a unique loop $A \xrightarrow{v} A$ of the form 1. In NL it is enough to perform Test 2 and Test 3 for one non-deterministically chosen bridge A within a strongly connected component. As the shortest, non-trivial loop $A \xrightarrow{v} A$ is unique, there is an NL-transducer that outputs v and we may assume that v is written on the input tape. Note here, that if $\mathcal{A}$ does not contain any strongly connected components, this transducer will not stop for any bridge A and we will not perform Test 2 or Test 3. Thus, we implicitly test $L(\mathcal{A})$ for infiniteness here.

We now prove the NL performance of Test 2 and 3 for a given loop $A \xrightarrow{v} A$ with $A = (p_1, p_2, q_1, q_2, 0)$.

Test 2: Decide the existence of words $x, y \in \Sigma^*$ and states $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ satisfying

1. $k \leq |x| < |v| + k$ and x is a prefix of some word in v^+,
2. $0 \leq |y| < |v|$ and xy is a prefix of some word in v^+,
3. $d_1 = p_1 \cdot xy$ and $d_2 = p_2 \cdot x$,

4. $q_1 = d_1 \cdot \overline{x}\overline{v}^{n_1}$ and during the computation of $d_1 \cdot \overline{x}\overline{v}^{n_1}$ we see after exactly k steps a final state in $\mathcal{F}_1$ and then never again, and

5. $q_2 = d_2 \cdot \overline{y}\overline{x}\overline{v}^{n_2}$ and, let $e_2 = d_2 \cdot \overline{y}\overline{x}$, during the computation of $e_2 \cdot \overline{v}^{n_2}$ we do not see a final state in $\mathcal{F}_2$

but where for all factorizations $xy\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ we have $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$. If we find such a situation, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

Test 3: Decide the existence of words $x, y, z \in \Sigma^*$ with $z \neq 1$ and states $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ satisfying

1. $k \leq |x| < |v| + k$ and x is a prefix of some word in v^+,

2. $0 \leq |y| < |v|$ and xy is the longest common prefix of xyz and some word in v^+,

3. $z \in B(c_1, c_2, d_1, d_2)$, where $c_1 = p_1 \cdot xy$ and $c_2 = p_2 \cdot x$,

4. $q_1 = d_1 \cdot \overline{x}\overline{v}^{n_1}$ and during the computation of $d_1 \cdot \overline{x}\overline{v}^{n_1}$ we see after exactly k steps a final state in $\mathcal{F}_1$ and then never again, and

5. $q_2 = d_2 \cdot \overline{y}\overline{x}\overline{v}^{n_2}$ and, let $e_2 = d_2 \cdot \overline{y}\overline{x}$, during the computation of $e_2 \cdot \overline{v}^{n_2}$ we do not see a final state in $\mathcal{F}_2$

but where for all factorizations $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ we have $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$. If we find such a situation, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

Lemma 3.18. *For a given loop $A \xrightarrow{v} A$ with $A = (p_1, p_2, q_1, q_2, 0)$ Test 2 and Test 3 can be performed in* NL.

Proof. For both tests we guess the lengths of the words x and y which satisfy condition 1 and 2 and therefore xy is a prefix of a word in v^+. Thus, we can remember x and y because v is available by the input. For Test 2 we compute $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ by condition 3 and for Test 3 we guess states (d_1, d_2). We verify that conditions 4 and 5 hold, which is easy because we can reconstruct x and y.

For Test 2 we only have to check whether for all factorizations $xy\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ with $|\delta| = k$ the condition $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$ holds. This can easily be done in NL because we have full access to the word $xy\overline{x}\overline{v}$.

Test 3 is a bit more tricky. We guess $a \in \Sigma$ and we check that xya is not a prefix of a word in v^+. We have to verify that a path from c_1 to d_1 exists which is labelled by some non-empty word $z \in a\Sigma^*$ and that a path from c_2 to d_2 exists which is labelled by $\overline{z}$. This can be achieved by a graph reachability algorithm which uses forward transitions in $\mathcal{A}_1$ and backwards transitions in $\mathcal{A}_2$ (just like in the proof of Lemma 3.16). In a factorization $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$

we cannot have that xy is a proper prefix of $\mu\delta$ otherwise xya would be a prefix of vx, due to the length constraint of y. But this was excluded by the choice of a. Thus, $\mu\delta$ is a prefix of xy and $\overline{\delta}\overline{\mu}$ is a suffix of $\overline{y}\overline{x}$. This means, to ensure that there is no factorization with $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \in \mathcal{F}_2$, we do not need to remember the word z. We just compute $d_2 \cdot \overline{y}\overline{x}$ and during this computation we validate that there are no final states in $\mathcal{F}_2$ after k or more steps. □

3.2.7 Time Complexity Analysis

Now, we prove the time complexity of the algorithm. Recall from Theorem 3.3 that we claimed the problem whether the hairpin completion $\mathcal{H}_k(L_1, L_2)$ is regular is solvable in time

i.) $\mathcal{O}(n^2)$ if $L_1 = \emptyset$ or $L_2 = \emptyset$.

ii.) $\mathcal{O}(n^6)$ if $L_1 = \overline{L_2}$.

iii.) $\mathcal{O}(n^8)$ in general.

The first step of the algorithm is to construct the automaton $\mathcal{A}$. The crucial part is to decide for all quadruples $(p_1, p_2, q_1, q_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \mathcal{Q}_1 \times \mathcal{Q}_2$ whether it is a basic bridge. The construction of the states and transitions of $\mathcal{A}$ is plain if we have access to a table containing all basic bridges. At this point we also store within the table whether a quadruple is an a-bridge for $a \in \Sigma$. We need this information for Test 3.

Lemma 3.19. *A table containing all basic bridges and a-bridges can be computed in time $\mathcal{O}(n_1^2 n_2^2)$.*

Proof. Let $\mathcal{Q}_1 \times \mathcal{Q}_2$ be the set of states of a non-deterministic auxiliary automaton. In this automaton a have transitions $(p_1, q_2) \xrightarrow{a} (q_1, p_2)$ for $a \in \Sigma$ if $p_1 \cdot a = q_1$ and $p_2 \cdot a = q_2$. Hence this automaton uses usual transitions in $\mathcal{A}_1$ and backwards transitions in $\mathcal{A}_2$. Its construction can obviously be performed within the time bound. Note that the number of transitions of this automaton is bounded by $n_1 n_2 \cdot |\Sigma|$.

A quadruple $(p_1, p_2, q_1, q_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \mathcal{Q}_1 \times \mathcal{Q}_2$ is a basic bridge if there is a word w which labels a path $(p_1, q_2) \xrightarrow{w} (q_1, p_2)$ and it is an a-bridge if $w \in a\Sigma^*$. Therefore, we run a depth-first search for all $a \in \Sigma$ and $(p_1, q_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ starting with an a-transition and for every reachable state (q_1, p_2) we mark (p_1, p_2, q_1, q_2) as a basic bridge and as an a-bridge. This yields the time performance. □

Recall that the number of transitions in $\mathcal{A}$ is in $\mathcal{O}(N)$ where $N \leq n^2$ if $L_1 = \emptyset$ or $L_2 = \emptyset$, $N \leq n^3$ if $L_1 = \overline{L}_2$, and $N \leq n^4$ in general. Henceforth, we assume that every bridge in $\mathcal{A}$ is reachable from and initial bridge and leads to some final bridge, this can easily be verified in time $\mathcal{O}(N)$.

For *i.)* it suffices to perform Test 0, which is a finiteness test for $L(\mathcal{A})$, in time $\mathcal{O}(N)$. Since the language $L(\mathcal{A})$ is infinite if and only if the automaton $\mathcal{A}$ contains a non-trivial loop, we can use the well-known algorithm of Tarjan [40] which decomposes a directed graph into its strongly connected components in linear time.

Lemma 3.20. *It is decidable whether $L(\mathcal{A})$ is infinite in time $\mathcal{O}(N)$.*

Proof. This is immediate by the argumentation above. □

Furthermore, by the algorithm of Tarjan, we can compute the list of all non-trivial strongly connected components K of the automaton $\mathcal{A}$ in time $\mathcal{O}(N)$. We assign to each strongly connected component $\kappa \in K$ a bridge A_κ. We also assign to κ a (shortest) non-trivial loop $A_\kappa \xrightarrow{v_\kappa} A_\kappa$. By Lemma 3.9, for a regular hairpin completion this loop is unique and $|v_\kappa| = N_\kappa$. Therefore, we simply assign some loop which does not use the bridge A_κ in the middle and we stop with output that $\mathcal{H}_k(L_1, L_2)$ is not regular in case $|v_\kappa| = N_\kappa$. So far we used $\mathcal{O}(n^4)$ time. Now, we prove the time performance of Test 1.

Test 1: Decide whether there is $\kappa \in K$ and a path $A_\kappa \xrightarrow{w} F$ such that w is not a prefix of a word in v_κ^+. If there is such a path, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

Lemma 3.21. *Test 1 can be performed in time $\mathcal{O}(N^2)$.*

Proof. Let $\kappa \in K$, $A = A_\kappa$, and $v = v_\kappa$ as above. We assign to each bridge that is reachable from A a subset of marks from $\{0, \ldots, N_\kappa - 1\}$. A mark i is assigned to a bridge B if B is reachable from A with a word in $v^* v[1, i]$. Test 1 yields that $\mathcal{H}_k(L_1, L_2)$ is not regular if and only if we there is a bridge that is marked by i and has an outgoing a-transition where $a \neq v[i+1]$. The marking algorithm can be performed by a depth-first search that runs in time $\mathcal{O}(N \cdot N_\kappa)$. Summing over all strongly connected components we deduce a time complexity in $\mathcal{O}(\sum_{\kappa \in K} N \cdot N\kappa) \subseteq \mathcal{O}(N^2)$. □

For Test 2 and 3 let us fix one strongly connected component $\kappa \in K$ and let $v = v_\kappa$ and $A = A_\kappa = ((p_1, p_2), q_1, q_2, 0)$.

Test 2: Decide the existence of words $x, y \in \Sigma^*$ and states $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ satisfying

1. $k \leq |x| < |v| + k$ and x is a prefix of some word in v^+,
2. $0 \leq |y| < |v|$ and xy is a prefix of some word in v^+,
3. $d_1 = p_1 \cdot xy$ and $d_2 = p_2 \cdot x$,

4. $q_1 = d_1 \cdot \overline{x}\overline{v}^{n_1}$ and during the computation of $d_1 \cdot \overline{x}\overline{v}^{n_1}$ we see after exactly k steps a final state in $\mathcal{F}_1$ and then never again, and

5. $q_2 = d_2 \cdot \overline{y}\overline{x}\overline{v}^{n_2}$ and, let $e_2 = d_2 \cdot \overline{y}\overline{x}$, during the computation of $e_2 \cdot \overline{v}^{n_2}$ we do not see a final state in $\mathcal{F}_2$

but where for all factorizations $xy\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ we have $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$. If we find such a situation, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

Lemma 3.22. *Test 2 can be performed in time $\mathcal{O}(N^2)$.*

Proof. For the fixed strongly connected component κ, we have to compute all words x and y such that there are runs

$$p_1 \xrightarrow{xy} d_1 \xrightarrow{\overline{x}\overline{v}^{n_1}} q_1, \qquad\qquad p_2 \xrightarrow{x} d_2 \xrightarrow{\overline{y}\overline{x}\overline{v}^{n_2}} q_2$$

and the conditions 1 to 5 are satisfied. Moreover, in addition we demand that during the computation of $d_2 \cdot \overline{y}\overline{x}\overline{v}^{n_2}$ we do not meet any final state after more than $k-1$ steps. (In case such a final state exists, either condition 5 is breached or a factorization $xy\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ with $|\delta| = k$ and $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$ exists.) By backwards searches in $\mathcal{A}_1$ and $\mathcal{A}_2$ starting at states q_1 and q_2, respectively, and searching for paths labelled by suffixes of $\overline{v}^+$, we compute all pairs (x, xy) satisfying these conditions in time $\mathcal{O}(N \cdot N_\kappa)$.

At this stage we also compute the position $\ell(x, xy)$ of the last final state during the run $p_2 \cdot vx\overline{y}\overline{x}$ and we let $\ell(x, xy) = 0$ if no such state exists. (Note that $0 \leq \ell(x, xy) < N_\kappa + |x| + k$.) If a factorization $xy\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ with $|\delta| = k$ and $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \in \mathcal{F}_2$ exists, then $|xy\overline{x}\overline{v}| - \ell(x, xy)$ gives us a lower bound for the length of μ.

Let $m(x, xy)$ be the length of the longest μ such that a factorization $xy\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ with $|\delta| = k$ exists (without the condition $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \in \mathcal{F}_2$).

There is a factorization $xy\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ with $|\delta| = k$ and $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \in \mathcal{F}_2$ if and only if $m(x, xy) \geq |xy\overline{x}\overline{v}| - \ell(x, xy)$ and $\ell(x, xy) - k \geq |xy\overline{x}\overline{v}|/2$.

We need to precompute the values $m(x, xy)$ efficiently, which turns out to be a little bit tricky. For $0 \leq i \leq |N|_\kappa$ we let $v_i = v[i+1, N_\kappa]v[1, i]$ be the conjugate of v starting at the $(i+1)$-st letter. We wish to match position in v_i^2 with positions in $\overline{v}^2$. For each $0 \leq j \leq N_\kappa$ we store the maximal $m \leq N_\kappa$ such that $v_i^2[j, j+m] = \overline{v}^2[j, j+m]$ in a table entry $M(i, j)$, see Figure 12. For each i one run (from right to left) over the words v_i^2 and $\overline{v}^2$ is enough. It takes $\mathcal{O}(N_\kappa^2)$ time to build the table M. Now, if we know the length m' of the longest common prefix of $v_{|xy|}$ and $\overline{x}\overline{v}$, then $m(x, xy) = |xy| + m' - k$ (yet at most $|xy\overline{x}\overline{v}|/2 - k$). The length of m' is stored in $M(|xy\overline{x}| \bmod N_\kappa, (-|\overline{x}|) \bmod N_\kappa)$, hence we have access to $m(x, xy)$ in constant time.

All in all Test 4 can be performed in $\mathcal{O}(\sum_{\kappa \in K} N \cdot N\kappa) \subseteq \mathcal{O}(N^2)$. □

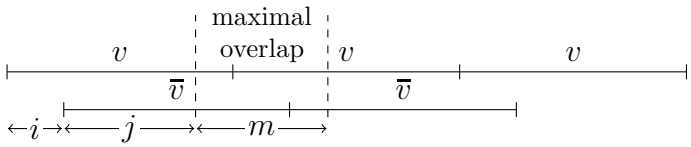


Figure 12: Matching positions of v_i^2 with $\overline{v}^2$

Test 3: Decide the existence of words $x, y, z \in \Sigma^*$ with $z \neq 1$ and states $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ satisfying

1. $k \leq |x| < |v| + k$ and x is a prefix of some word in v^+,
2. $0 \leq |y| < |v|$ and xy is the longest common prefix of xyz and some word in v^+,
3. $z \in B(c_1, c_2, d_1, d_2)$, where $c_1 = p_1 \cdot xy$ and $c_2 = p_2 \cdot x$,
4. $q_1 = d_1 \cdot \overline{x}\overline{v}^{n_1}$ and during the computation of $d_1 \cdot \overline{x}\overline{v}^{n_1}$ we see after exactly k steps a final state in $\mathcal{F}_1$ and then never again, and
5. $q_2 = d_2 \cdot \overline{y}\overline{x}\overline{v}^{n_2}$ and, let $e_2 = d_2 \cdot \overline{y}\overline{x}$, during the computation of $e_2 \cdot \overline{v}^{n_2}$ we do not see a final state in $\mathcal{F}_2$

but where for all factorizations $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ we have $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$. If we find such a situation, then stop with the output that $\mathcal{H}_k(L_1, L_2)$ is not regular.

Lemma 3.23. *Test 3 can be performed in time $\mathcal{O}(n_{12}n_1^2n_2^2n)$ which is $\mathcal{O}(n^6)$ if $L_1 = \overline{L_2}$ and $\mathcal{O}(n^7)$ in general.*

Proof. Let κ be a fixed strongly connected component, as above. In order to perform Test 3 we create two tables T_1 and T_2. The table T_1 holds all pairs $(c_2, d_1) \in \mathcal{Q}_2 \times \mathcal{Q}_1$ such that a word x exists with

1. $k \leq |x| < |v| + k$ and x is a prefix of a word in v^+,
2. $p_2 \cdot x = c_2$,
3. $d_1 \cdot \overline{x}\overline{v}^{n_1} = q_1$, and during the computation of $d_1 \cdot \overline{x}\overline{v}^{n_1}$ we see a final state after exactly k steps and then never again.

We call x a witness for $(c_2, d_1) \in T_1$. The table T_2 holds all triples $(c_1, d_2, a) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \Sigma$ such that a prefix $y' < v$ exists with

1. $y'a$ is no prefix of v,
2. $p_1 \cdot y' = c_1$,

3. $d_2 \cdot \overline{y'}\overline{v}^{n_2} = q_2$, and during the computation of $d_2 \cdot \overline{y'}\overline{v}^{n_2}$ we do not see a final state after k or more steps.

We call y' a witness for $(c_1, d_2, a) \in T_2$. By backwards computing in the second component, the tables T_1 and T_2 can be created in $\mathcal{O}(N_\kappa n_1)$ and $\mathcal{O}(N_\kappa n_2)$, respectively.

We claim, that Test 3 yields that $\mathcal{H}_k(L_1, L_2)$ is not regular if and only if there exists a pair $(c_2, d_1) \in T_1$ and a triple $(c_1, d_2, a) \in T_2$ such that (c_1, c_2, d_1, d_2) is an a-bridge.

First assume $(c_2, d_1) \in T_1$, $(c_1, d_2, a) \in T_2$, and (c_1, c_2, d_1, d_2) is indeed an a-bridge. Let x and y' be the the witnesses for $(c_2, d_1) \in T_1$ and $(c_1, d_2, a) \in T_2$, respectively. Choose $z \in B(c_1, c_2, d_1, d_2) \cap a\Sigma^*$ and y such that xy is a prefix of some word in v^+, $|xy| \equiv |y'| \pmod{|v|}$, and $|y| < |v|$. Verify that x, y, z and (d_1, d_2) satisfy the conditions 1 to 5 of Test 3. However, for any factorization $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ with $|\delta| = k$, the word $\mu\delta$ has to be a prefix of xy, since xya is no prefix of vx. During the computation of $d_2 \cdot \overline{y'}\overline{v}^{n_2}$ we did not see a final state after more than $k-1$ steps. The same holds for the computation of $d_2 \cdot \overline{y}\overline{x}\overline{v}^{n_2}$ and, therefore, we have $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$.

Now assume that $x, y, z \in \Sigma^*$, $z \neq 1$, and $(d_1, d_2) \in \mathcal{Q}_1 \times \mathcal{Q}_2$ exist, which satisfy the conditions 1 to 5 of Test 3 but where for all factorizations $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ we have $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \notin \mathcal{F}_2$. Choose $y' < v$ such that $|xy| \equiv |y'| \pmod{|v|}$. Let $c_2 = p_2 \cdot x$, $c_1 = p_1 \cdot y'$ and $a \in \Sigma$ be the first letter of z. Obviously, (c_1, c_2, d_1, d_2) is an a-bridge and x is a witness for $(c_2, d_1) \in T_1$. If we saw a final state after more than $k-1$ steps during the computation of $d_2 \cdot \overline{y'}\overline{v}^{n_2}$, then a factorization $xyz\overline{x}\overline{v} = \mu\delta\beta\overline{\delta}\overline{\mu}$ with $p_2 \cdot \mu\delta\overline{\beta}\overline{\delta} \in \mathcal{F}_2$ would exist and hence y' is a witness for $(c_1, d_2, a) \in T_2$.

Since the table of a-bridges is precomputed (cf. Lemma 3.19), this test can be performed in time $\mathcal{O}(|T_1| \cdot |T_2|)$. The set of all first components of T_1 (resp. T_2) is bounded by both, the size of N_κ and n_2 (resp. n_1). Therefore, we have $|T_1| \in \mathcal{O}(n_1 \cdot \min(N_\kappa, n_2))$ and $|T_2| \in \mathcal{O}(n_2 \cdot \min(N_\kappa, n_1))$. By symmetry, assume $n_2 \leq n_1$.

Test 3 can be performed in time

$$\mathcal{O}\left(\sum_{\kappa \in K} \left(N_\kappa n_1 + N_\kappa n_2 + n_1 n_2 \cdot \min(N_\kappa, n_1) \cdot \min(N_\kappa, n_2)\right)\right) \subseteq$$
$$\mathcal{O}\left(n_{12} n_1^2 n_2 + n_{12} n_1 n_2^2 + \sum_{\kappa \in K, N_\kappa \geq n_2} n_1^2 n_2^2 + \sum_{\kappa \in K, N_\kappa < n_2} N_\kappa^2 n_1 n_2\right)$$

(Recall that $n_1 \leq n \leq n_{12} \leq n_1 n_2 \leq n^2$ and $\sum_{\kappa \in K} N_\kappa \leq N = n_{12} n_1 n_2$.)

Since there are at most $n_{12} n_1$ strongly connected components with a size of more than n_2 states, we have

$$\sum_{\kappa \in K, N_\kappa \geq n_2} n_1^2 n_2^2 \leq n_{12} n_1^3 n_2^2.$$

For the last term we can use the approximation

$$\sum_{\kappa\in K, N_\kappa<n_2} N_\kappa^2 n_1 n_2 \leq \sum_{\kappa\in K, N_\kappa<n_2} N_\kappa n_1 n_2^2 \leq n_{12} n_1^2 n_2^3.$$

We conclude, Test 3 can be performed in time $\mathcal{O}(n_{12} n_1^2 n_2^2 n)$. □

3.3 Varieties

Let L_1 and L_2 be languages whose syntactic monoids belong to **A** (resp. **LDA**). Equivalently, we may assume that there is a morphism $h\colon \Sigma^* \to M$ with $M \in \mathbf{A}$ (resp. $M \in \mathbf{LDA}$) which accepts both languages L_1 and L_2. In this section we prove:

Theorem 3.24. *Let $h\colon \Sigma^* \to M$ be a morphism recognizing L_1 and L_2 and let $\mathcal{V}$ be the variety* **A** *or* **LDA**. *If $M \in \mathcal{V}$ and $\mathcal{H}_k(L_1, L_2)$ is regular, the syntactic monoid of $\mathcal{H}_k(L_1, L_2)$ belongs to $\mathcal{V}$ as well.*

The proof can be found in Section 3.3.3. Section 3.3.1 and 3.3.2 are preliminary for the proof. In the Section 3.3.1 we provide a lemma which tells us that a language v^* is recognizable by a monoid in **LDA** if v is primitive. In Section 3.3.2 we introduce the technique of relativization of first-order formulae.

3.3.1 Kleene Star of Primitive Words

Later we consider languages of the from v^* where v is primitive. Here we show that these languages are definable in $\mathrm{FO}^2[<, +1]$ and hence they are recognizable by monoids in **LDA** and **A**.

More precisely, we can proof that $M(v^*)$ is in **LDA** (resp. **A**) if and only if v is primitive. The proof of the only-if-part is omitted here since it is not needed for Theorem 3.24. For the interested reader we refer to [32].

Lemma 3.25. *If v is a primitive word, then v^+ and v^* are definable in* $\mathrm{FO}^2[<, +1]$.

Proof. Let $\ell = |v|$ and for $0 \leq i < \ell$ define $v_i = v[i+1, \ell]v[1, i]$ as the conjugate of v starting with the $(i+1)$-st letter. As v is primitive, the conjugates are mutually different.

Let $U = \Sigma^\ell \setminus \{v_i \mid 0 \leq i < \ell\}$ denote the set of words of length ℓ which are no factor of a word in v^+. We claim that v^+ is defined by the $\mathrm{FO}^2[<, +1]$ formula

$$\varphi \;=\; \exists x \forall y\big(x \leq y \wedge \overrightarrow{\lambda}_v(x)\big) \wedge \exists x \forall y\big(y \leq x \wedge \overleftarrow{\lambda}_v(x)\big) \wedge \bigwedge_{u\in U} \forall x\, \neg \overrightarrow{\lambda}_u(x).$$

Note that $w \in L(\varphi)$ if and only if v is a prefix and a suffix of w and it contains no factor from U. Obviously, we have $v^+ \subseteq L(\varphi)$.

Vice versa, let $w \in L(\varphi)$ and assume $w \notin v^+$ by contradiction. If w is a prefix of a power of v, the conjugate v_i with $i = |w| \bmod \ell \neq 0$ is a suffix of w. But $v_0 = v$ is a suffix of w as well which contradicts the primitivity of v.

Otherwise, let $w = w_1 a w_2$ with $a \in \Sigma$ such that w_1 is the longest common prefix of w and a word in v^+. The suffix of $w_1 a$ of length ℓ must not be in U hence it is a conjugate v_j of v. Let $i = |w_1 a| \bmod \ell$. The conjugates v_i and v_j equal at positions 1 to $\ell - 1$, but differ at position ℓ. Therefore, we would have $|v_i|_a \neq |v_j|_a$ which is impossible as v_i and v_j are conjugates of each other.

We conclude $v^+ = L(\varphi)$ is definable in $\mathrm{FO}^2[<, +1]$. The empty word is defined by the formula $\forall x \bot$ and hence $v^* = L(\varphi \vee \forall x \bot)$ is definable in $\mathrm{FO}^2[<, +1]$, too. □

3.3.2 Relativization

Let us introduce a technique called relativization, see also [39]. This technique allows us to apply a first-order formula to a specified factor in a word. However, the positions that limit this factor have to be uniquely defined by other first-order formulae.

Let $\varphi \in \mathrm{FO}[<]$ be a formula with m free variables and let $\chi \in \mathrm{FO}[<]$ be a formula with one free variable which is true at at most one position for every word. The *relativization* of φ by $\leq \chi$ is denoted by $\langle \varphi \rangle_{\leq \chi}$ and defined such that

$$w, j_1, \ldots, j_m \models \langle \varphi \rangle_{\leq \chi}$$

if and only if there is a position ℓ satisfying $w, \ell \models \chi$ and

$$w[1, \ell], j_1, \ldots, j_m \models \varphi.$$

Note that all positions $j_1, \ldots, j_m$ have to be at most ℓ. The relativizations $\langle \varphi \rangle_{<\chi}$, $\langle \varphi \rangle_{\geq \chi}$, and $\langle \varphi \rangle_{>\chi}$ are defined respectively.

Lemma 3.26. *Let $\varphi \in \mathrm{FO}^2[<, +1]$ be a sentence and $\chi \in \mathrm{FO}^2[<, +1]$ be a formula with one free variable which is true at at most one position in every word. There is a sentence $\psi \in \mathrm{FO}^2[<, +1]$ which is equivalent to $\langle \varphi \rangle_{\leq \chi}$.*

The same is true for the relativizations $\langle \varphi \rangle_{<\chi}$, $\langle \varphi \rangle_{\geq \chi}$, and $\langle \varphi \rangle_{>\chi}$.

Proof. We will prove the lemma for $\langle \varphi \rangle_{\leq \chi}$, only. Let w be some word. If $w \models \langle \varphi \rangle_{\leq \chi}$, then $\chi(x)$ is true at exactly one position in w. Therefore, we let $\psi = \exists x \, \chi(x) \wedge \psi'$ and, henceforth, we assume that $w, \ell \models \chi$.

By induction on the structure of φ, we prove that there is a sentence $\psi' \in \mathrm{FO}^2[<, +1]$ such that $w \models \psi'$ if and only if $w[1, \ell] \models \varphi$. For the induction, we

have to allow free variables, but we ensure the invariant that free variables are restricted to be at most ℓ. Obviously, the claim is true for atomic formulae:

$$\langle \top \rangle_{\leq\chi} \equiv \top, \qquad \langle \lambda_a(x) \rangle_{\leq\chi} \equiv \lambda_a(x),$$
$$\langle x < y \rangle_{\leq\chi} \equiv x < y, \qquad \langle x = y + 1 \rangle_{\leq\chi} \equiv x = y + 1.$$

It is also plain that the claim holds for boolean combinations of formulae $\varphi_1, \varphi_2 \in \mathrm{FO}^2[<, +1]$:

$$\langle \neg\varphi_1 \rangle_{\leq\chi} \equiv \neg \langle \varphi_1 \rangle_{\leq\chi} \qquad \langle \varphi_1 \wedge \varphi_2 \rangle_{\leq\chi} \equiv \langle \varphi_1 \rangle_{\leq\chi} \wedge \langle \varphi_2 \rangle_{\leq\chi}$$
$$\langle \varphi_1 \vee \varphi_2 \rangle_{\leq\chi} \equiv \langle \varphi_1 \rangle_{\leq\chi} \vee \langle \varphi_2 \rangle_{\leq\chi}.$$

Since a universal quantifier may always be expressed by an existential quantifier and negation, it suffices to consider the case $\varphi = \exists x\, \varphi'(x)$. Here, we let

$$\psi' = \exists x \big(\exists y \big(\chi(y) \wedge x \leq y \big) \wedge \langle \varphi'(x) \rangle_{\leq\chi} \big)$$

It is easy to see, that $w \models \psi'$ if and only if there is a position $j \leq \ell$ such that $w, j \models \langle \varphi'(x) \rangle_{\leq\chi}$, respectively $w[1, \ell], j \models \varphi'(x)$ by induction hypothesis, which is equivalent to $w[1, \ell] \models \varphi$. □

Using the relativization technique it is easy to show that the languages definable in $\mathrm{FO}^2[<, +1]$ are closed under concatenation with words.

Lemma 3.27. *Let $\varphi \in \mathrm{FO}^2[<, +1]$ be a sentence and v be a word. The languages $vL(\varphi)$ and $L(\varphi)v$ are definable in $\mathrm{FO}^2[<, +1]$, too.*

Proof. We let $\chi(x)$ uniquely define the last letter of the first occurrence of v:

$$\chi(x) = \overleftarrow{\lambda}_v(x) \wedge \forall y \big(\neg \overleftarrow{\lambda}_v(y) \vee x \leq y \big).$$

By Lemma 3.26 there is a sentence $\psi \in \mathrm{FO}^2[<, +1]$ such that

$$\psi \equiv \exists x \forall y \big(x \leq y \wedge \overrightarrow{\lambda}_v(x) \big) \wedge \langle \varphi \rangle_{>\chi}$$

Obviously, we have $L(\psi) = vL(\varphi)$. The construction for $L(\varphi)v$ is symmetric. □

3.3.3 Proof of Theorem 3.24

Let $\mathcal{V}$ be the variety **A** or **LDA** and let $h \colon \Sigma^* \to M$ be a morphism with $M \in \mathcal{V}$ recognizing L_1 and L_2. For words u and v we write $u \sim v$ if $h(u) = h(v)$.

We reuse the definitions and results from the proof of Theorem 3.4 in Section 3.1. Let us recall that a triple (α, B, C) defines a linear language $L_{\alpha,B,C}$ and the hairpin completion is a finite union of languages of this form. We fixed a constant $n \geq \max\{k, |M(C)|\}$. Here, we assume in addition $n \geq |M|$ which

implies that $v^n \sim v^{n+1}$ since M is aperiodic. By premise $\mathcal{H}_k(L_1, L_2)$ is regular and hence we derive from Section 3.1.3 that $L_{\alpha,B,C}$ is a finite union of languages of the forms

$$R_{u,v} = uv^n v^* B \overline{v}^* \overline{v}^n \overline{u} \qquad \text{and} \qquad \gamma\alpha B \overline{\alpha}\overline{\gamma}$$

where $(u, v) \in V$ (i. e., $|u| \leq n$, $1 \leq |v| \leq n$, and $uv^n v^* \subseteq C\alpha$). Also recall that $B = h^{-1}(s)$ for some element $s \in M$ and, therefore, B is recognized by M. Since both varieties **A** and **LDA** are closed under concatenation with words, languages of the second form are recognized by a monoid in $\mathcal{V}$ (cf. Lemma 3.27). In order to prove Theorem 3.24 it suffices to construct a language R' which is recognized by a monoid in $\mathcal{V}$ such that $R_{u,v} \subseteq R' \subseteq \mathcal{H}_k(L_1, L_2)$.

First assume $\mathcal{V} = \mathbf{A}$. If the language v^* is aperiodic, we are done. By the discussion in Section 3.3.1 and by Lemma 3.25 this is true if and only if v is primitive. The next lemma tells us, if v is not primitive, we may replace v by its primitive root.

Lemma 3.28. *Let $(u, v) \in V$ and let r be the primitive root of v, then $(u, r) \in V$. (Assuming M is aperiodic.)*

Proof. Since $(u, v) \in V$, we have $|u| \leq n$, $1 \leq |r| \leq |v| \leq n$ and $uv^n v^* \subseteq C\alpha$. We need to prove that $ur^n r^* \subseteq C\alpha$. Recall that $ur^i \in C\alpha$ if and only if

1. $ur^i\beta\overline{\alpha}$ is the longest prefix of $ur^i\beta\overline{r}^i\overline{u}$ which belongs to L_1,
2. if a suffix of $ur^i\beta\overline{r}^i\overline{u}$ belongs to L_2 it is a suffix of $\alpha\beta\overline{r}^i\overline{u}$,
3. $ur^i \sim uv^n$, and $\overline{r}^i\overline{u} \sim \overline{v}^n\overline{u}$.

Let $i \geq n$. We have $r^i \sim v^n$ and $\overline{r}^i \sim \overline{v}^n$ because M is aperiodic. Hence condition 3 is satisfied and $ur^i\beta\overline{\alpha} \in L_1$.

By contradiction, assume that condition 1 is breached, i. e., there is $\alpha < x \leq \overline{r}^i\overline{u}$ such that $ur^i\beta x \in L_1$. In case $x \leq \overline{r}^i$, we have $uv^i\beta x \in L_1$, too, and hence $uv^i \notin C\alpha$. Otherwise, let $x = \overline{r}^i y \leq \overline{r}^i\overline{u}$. We obtain $uv^n\beta\overline{v}^n y \in L_1$ and, again, $uv^n \notin C\alpha$. Both cases yield a contradiction.

The argument for condition 2 is symmetric and we conclude $(u, r) \in V$. □

Let $(u, v) \in V$ and let r be the primitive root of v. Obviously, we have $R_{u,v} \subseteq R_{u,r}$ and, therefore, we may assume for all pairs $(u, v) \in V$ that v is primitive. Moreover, for $\mathcal{V} = \mathbf{A}$ we proved Theorem 3.24.

Now let $\mathcal{V} = \mathbf{LDA}$ and let $(u, v) \in V$ with v primitive. We obtain the language R' by replacing B by a modified language B'. Let

$$N = \left\{ s \in M \;\middle|\; \exists i, j \geq 0 \colon h(v)^i \cdot s \cdot h(\overline{v})^j = h(B) \right\}.$$

In other words, x belongs to $h^{-1}(N)$ if and only if there are $i, j \geq 0$ such that $v^i x \overline{v}^j \in B$. Note that we may assume $i, j \leq n$. We define $B' = h^{-1}(N) \setminus (v\Sigma^* \cup \Sigma^*\overline{v})$ as the language where we exclude all words with prefix v or suffix $\overline{v}$ from $h^{-1}(N)$. We let $R' = uv^n v^* B' \overline{v}^* \overline{v}^n \overline{u}$ and claim that $R_{u,v} \subseteq R' \subseteq \mathcal{H}_k(L_1, L_2)$ and $M(R') \in \mathbf{LDA}$ which will complete the proof of Theorem 3.24. It is obvious that B is a subset of $v^* B' \overline{v}^*$ and hence $R_{u,v} \subseteq R'$.

Now let $\pi = uv^i x \overline{v}^j \overline{u} \in R'$ with $i, j \geq n$ and $x \in B'$. Choose $0 \leq i', j' \leq n$ such that $\beta = v^{i'} x v^{j'} \in B$. We have $v^{i-i'}\beta \sim v^n \beta$ (since $v^n \sim v^{n+1}$). Hence, if $i \geq j - j'$, we can use the prefix $uv^{i-i'}\beta\alpha \in L_1$ to build the hairpin. Otherwise there is a suffix y of $\alpha\beta\overline{v}$ such that $y\overline{v}^{j-j'-1}\overline{u} \in L_2$ and it can be used to build the hairpin (cf. Section 3.1.2). We conclude $\pi \in \mathcal{H}_k(L_1, L_2)$.

The next lemma tells us that R' is definable in $\mathrm{FO}^2[<, +1]$, thus its syntactic monoid belongs to **LDA**.

Lemma 3.29. *R' is definable in $\mathrm{FO}^2[<, +1]$.*

Proof. We will prove that $v^* B' \overline{v}^*$ is definable in $\mathrm{FO}^2[<, +1]$ which implies that R' is also definable in $\mathrm{FO}^2[<, +1]$. As $h^{-1}(N)$ is recognized by M there is a formula $\psi_N \in \mathrm{FO}^2[<, +1]$ which defines $h^{-1}(N)$. By a slight modification of ψ_N we can easily define B' in $\mathrm{FO}^2[<, +1]$ (e. g., all words that do not start with v are defined by $\exists x \forall y\big(x \leq y \wedge \neg \overrightarrow{\lambda}_v(x)\big)$).

By Lemma 3.25 there are $\mathrm{FO}^2[<, +1]$-sentences defining v^* and $\overline{v}^*$. If we can state two $\mathrm{FO}^2[<, +1]$-formulae $\chi_1(x)$ and $\chi_2(x)$ which in a word $w \in v^* B' \overline{v}^*$ uniquely define the positions ℓ_1 and ℓ_2, respectively, such that $w[1, \ell_1] \in v^*$, $w[\ell_1{+}1, \ell_2] \in B'$, and $w[\ell_2{+}1, |w|] \in \overline{v}^*$, we can use the relativization technique to complete the proof, see Section 3.3.2. If B' contains prefixes of v or suffixes of $\overline{v}$, they have to be treated separately. At first assume B' does not contain those words. The position ℓ_1 is the first position where the predicate $\overleftarrow{\lambda}_v$ is satisfied but $\ell_1 + 1$ does not satisfy $\overrightarrow{\lambda}_v$. Hence χ_1 can be defined in $\mathrm{FO}^2[<, +1]$ and χ_2 can be defined symmetrically.

Now assume v_1, a (proper) prefix of v, belongs to B' and $v_1 v_2 = v$. If $\overline{v} \neq v_2 v_1$ we can define the limits χ_1 and χ_2 almost in the same way as above. Otherwise we have $v^* v_1 \overline{v}^* = (v_1 v_2)^* v_1 (v_2 v_1)^* = v^* v_1$ which is definable in $\mathrm{FO}^2[<, +1]$ as well. □

4 The Hairpin Lengthening of Regular Languages

The decidability problem we solved for the hairpin completion in Section 3 can as well be considered for the hairpin lengthening, namely, is it decidable whether the hairpin lengthening $\mathcal{HL}_k(L_1, L_2)$ of two regular languages L_1 and

L_2 is regular. Even though we were not able to solve the problem, we provide some interesting partial results. In Section 4.1 we use the approach from the Section 3 to solve the one-sided case (i. e., $L_1 = \emptyset$ or $L_2 = \emptyset$). However, our approach does not seem to work for the two-sided case. The fact that $\mathcal{HL}_k(L_1, L_2)$ is a linear context-free language [27] suggests that we start under similar conditions as we did for the hairpin completion, but we are no longer able to provide an unambiguous (linear) context-free grammar that describes $\mathcal{HL}_k(L_1, L_2)$. Indeed, in Section 4.2 we give an example for a regular language L whose right- and two-sided hairpin lengthening are inherent ambiguous context-free languages. Moreover, the degree of ambiguity of every context-free grammar describing $\mathcal{HL}_k(L, \emptyset)$ or $\mathcal{HL}_k(L, L)$ is unbounded.

4.1 The One-sided Case

We prove that the problem whether the one-sided hairpin lengthening is regular is decidable. Furthermore, we provide the same complexity results as we did for the one-sided hairpin completion (cf. Theorem 3.2 and 3.3). Our proof follows the idea and construction in Section 3.2 and leads to a result which is quite similar to the one in Proposition 3.8. We cut short on some of the arguments that were used in the same way before, so if you miss some explanation, you should probably read Section 3.2 first.

Theorem 4.1. *Let L be a regular language. The decision problem whether the right-sided hairpin lengthening $\mathcal{HL}_k(L, \emptyset)$ (resp. left-sided hairpin lengthening $\mathcal{HL}_k(\emptyset, L)$) is regular is*

i.) NL-*complete*

ii.) solvable in (deterministic) time $\mathcal{O}(n^2)$

if L (resp. $\overline{L}$) is given as DFA with n states.

Proof. We provide a proof for the right-sided hairpin completion, only. The proof of NL-hardness is analogous to the prove of Lemma 3.14 in Section 3.2.6.

Let $\mathcal{A}_1 = (\mathcal{Q}_1, \Sigma, E_1, \{q_0\}, \mathcal{F}_1)$ be the DFA accepting L (hence $n = |\mathcal{Q}|$). We start by constructing an NFA $\mathcal{A}$ like we did in Section 3.2.1, but this time we only have to consider pairs of states. For each pair $(p, q) \in \mathcal{Q}_1 \times \mathcal{Q}_1$ we define a language

$$B(p, q) = \{w \in \Sigma^* \mid p \cdot w = q\}$$

and we call (p, q) a basic bridge if this language is non-empty. We may assume that (q_0, q) is a basic bridge for every $q \in \mathcal{Q}$ (i. e., every state is reachable from the initial state). In our automaton $\mathcal{A}$ a pair (p, q) that forms a basic bridge

exists on different levels in $[k] = \{0, \ldots, k\}$. Hence, the state space of $\mathcal{A}$ (or set of bridges) is given by

$$\{(p, q, \ell) \in \mathcal{Q}_1 \times \mathcal{Q}_1 \times [k] \mid (p, q) \text{ is a basic bridge}\}$$

and the a-transitions for $a \in \Sigma$ are given by the arcs:

$$\begin{aligned}
(p, q \cdot \overline{a}, 0) &\xrightarrow{a} (p \cdot a, q, 0) && \text{for } q \cdot \overline{a} \notin \mathcal{F}_1, \\
(p, q \cdot \overline{a}, 0) &\xrightarrow{a} (p \cdot a, q, 1) && \text{for } q \cdot \overline{a} \in \mathcal{F}_1, \\
(p, q \cdot \overline{a}, \ell) &\xrightarrow{a} (p \cdot a, q, \ell + 1) && \text{for } 1 \leq \ell < k.
\end{aligned}$$

There is no bridge $(p, q, 0)$ with $q \in \mathcal{F}_1$ has an outgoing arc to level 0, bridges on level k have no outgoing arcs, and the number of arcs is limited by $|\Sigma| \cdot n^2 \cdot k \in \mathcal{O}(n^2)$. The set of initial bridges $\mathcal{I}$ contains all states $(p, q, 0)$ on level 0 (note that for the hairpin completion we asked $p = q_0$, but for the hairpin lengthening we can start with an arbitrary state). The set of final bridges $\mathcal{F}$ contains all states (p, q, k) on level k.

Remark 4.2. The list of basic bridges can be computed in $\mathcal{O}(n^2)$ time: Start a depth-first search from each state $p \in \mathcal{Q}_1$ and for every reachable state $q \in \mathcal{Q}_1$ output (p, q). In NL it is possible to test whether a pair of states (p, q) is a basic bridge, by a graph reachability algorithm. Therefore, we can construct the automaton $\mathcal{A}$ in $\mathcal{O}(n^2)$ time and there is a NL-transducer that outputs the automaton. Henceforth, we may assume that the automaton $\mathcal{A}$ is written on the input tape (cf. Lemma 3.15). We may also assume that every bridge in $\mathcal{A}$ has a path leading to a final bridge.

The automaton $\mathcal{A}$ encodes the hairpin lengthening $\mathcal{HL}_k(L, \emptyset)$ in the following way. For $\pi \in \mathcal{HL}_k(L, \emptyset)$ there is a factorization $\pi = \delta\gamma\alpha\beta\overline{\alpha}\overline{\gamma}$ such that $|\alpha| = k$ and there is a run through the DFA $\mathcal{A}_1$ as described in Figure 13 where $f \in \mathcal{F}_1$, i. e., we choose a factorization where $\delta\gamma\alpha\beta\overline{\alpha}$ is the longest prefix of π that belongs to L. We have that $\delta \in B(q_0, p)$, $\beta \in B(d, e)$, and in $\mathcal{A}$ there is a path $(p, q, 0) \xrightarrow{\gamma} (c, f, 0) \xrightarrow{\alpha} (d, e, k)$.

$$q_0 \xrightarrow{\delta} p \xrightarrow{\gamma} c \xrightarrow{\alpha} d \xrightarrow{\beta} e \xrightarrow{\overline{\alpha}} f \xRightarrow{\overline{\gamma}} q$$

Figure 13: Run defined by $\pi \in \mathcal{HL}_k(L, \emptyset)$

Vice versa, let $(p, q, 0) \in \mathcal{I}$ and $(d, e, k) \in \mathcal{F}$ be bridges and $(p, q, 0) \xrightarrow{\gamma\alpha} (d, e, k)$ a path in $\mathcal{A}$ with $|\alpha| = k$. For every $\delta \in B(q_0, p)$ and $\beta \in B(d, e)$ we have a situation as in Figure 13 with $f \in \mathcal{F}_1$ and hence $\delta\gamma\alpha\beta\overline{\alpha}\overline{\gamma} \in \mathcal{HL}_k(L, \emptyset)$.

We claim that the hairpin lengthening is regular if and only if the language accepted by the automaton $\mathcal{A}$ is finite. The if-part is plain. For the only-if-part

assume that $L(\mathcal{A})$ is infinite which implies there is a path

$$A \xrightarrow{v} A \xrightarrow{w} F$$

where $v \neq 1$ and F is a final state. As $A \xrightarrow{v} A$ is a non-trivial loop, we see that A is on level 0 and hence an initial state. Let $A = (p, q, 0)$ and $F = (d, e, k)$. For $\delta \in B(q_0, p)$, $\beta \in B(d, e)$ and $i \geq 0$ we have

$$\pi_i = \delta v^i w \beta \overline{w} \overline{v}^i \in \mathcal{HL}_k(L, \emptyset).$$

Let α be the suffix of w of length k. The prefix $\delta v^i w \beta \overline{\alpha}$ belongs to L and it is the longest prefix of π_i with this property. By contradiction, assume that $\mathcal{HL}_k(L, \emptyset)$ is regular. By a pumping argument, there are $s \geq 1$ and huge $t > |\delta w \beta|$ such that $\delta v^s w \beta \overline{w} \overline{v}^{st} \in \mathcal{HL}_k(L, \emptyset)$. The longest prefix of $\delta v^s w \beta \overline{w} \overline{v}^{st}$ that belongs to L is still the prefix $\delta v^s w \beta \overline{\alpha}$. This prefix is too short to form the hairpin — a contradiction.

Remark 4.3. The language $L(\mathcal{A})$ is infinite if and only if $\mathcal{A}$ contains a non-trivial loop. By the algorithm of Tarjan [40] we can be test whether $\mathcal{A}$ contains a non-trivial loop in linear time (with respect to the number of arcs in $\mathcal{A}$), hence in time $\mathcal{O}(n^2)$. For the NL algorithm we simply guess a bridge A on level 0 and a loop $A \xrightarrow{v} A$ with $v \neq 1$. □

4.2 Inherent Ambiguity

Let us recall that a grammar G has an unbounded degree of ambiguity if for all $m \in \mathbb{N}$ we find a word w with degree of ambiguity $d_G(w) \geq m$. We prove that there is a regular language L and all grammars that generate the hairpin lengthening of L have an unbounded degree of ambiguity, hence the hairpin lengthening of L is inherent ambiguous. In the proof we use the well-known Ogden's Lemma.

Lemma 4.4 (Ogden's Lemma)**.** *For each context-free grammar G with axiom S there is $n \in \mathbb{N}$ such that for every word $z \in L(G)$, if any n or more distinct positions in z are designated as distinguished, then there is some non-terminal A and there are words u, v, w, x, y such that:*

1. $S \overset{*}{\Longrightarrow} uAy \overset{*}{\Longrightarrow} uvAxy \overset{*}{\Longrightarrow} uvwxy = z$.
2. *w contains at least one of the distinguished positions.*
3. *Either u and v both contain distinguished positions, or x and y both contain distinguished positions.*
4. *vwx contains at most n distinguished positions.*

For a proof and exemplary application of the lemma, see [33].

The lemma implies, if we consider a factor z' of $z \in L(G)$ with $|z'| \geq n$, there is a factorization $z = uvwxy$ as above and the factor v or the factor x is completely covered by z'. Furthermore, the factor which is covered by z' has at most a length of n.

Theorem 4.5. *The hairpin lengthening $\mathcal{HL}_k(L_1, L_2)$ of two regular languages L_1 and L_2 may be inherent ambiguous context-free. Moreover, there is a regular language L and all context-free grammars generating $\mathcal{HL}_k(L, L)$ or $\mathcal{HL}_k(L, \emptyset)$ have an unbounded degree of ambiguity.*

Proof. Let $\Sigma = \{a, \overline{a}, b, \overline{b}\}$, let $\alpha = a^k$, and consider

$$L = (b^+\alpha)^+\overline{\alpha}.$$

Note first that every word in L starts with b and there is no $\overline{b}$ in it. Therefore, we are only able to form hairpins on the right side and $\mathcal{HL}_k(L, L) = \mathcal{HL}_k(L, \emptyset)$. Also note that a word of the form

$$b^{\ell_1}\alpha b^{\ell_2}\alpha \cdots b^{\ell_m}\alpha\overline{\alpha}\overline{b}^j$$

with $m, j, \ell_1, \ldots, \ell_m \geq 1$ belongs to $\mathcal{HL}_k(L, L)$ if and only if there is $1 \leq i \leq m$ such that $j \leq \ell_i$. We do not care about the other words in the hairpin lengthening since this condition suffices to show that every context-free grammar G that generates $\mathcal{HL}_k(L, L)$ or $\mathcal{HL}_k(L, \emptyset)$ has an unbounded degree of ambiguity.

Let $G = (V, \Sigma, P, S)$ be a context-free grammar generating $\mathcal{HL}_k(L, L)$. Let n be the parameter we obtain from Ogden's Lemma 4.4 and let $m > 1$ be the degree of ambiguity we are aiming for. We will examine derivations of words

$$z_{s,t} = (b^n\alpha)^{s-1}b^t\alpha(b^n\alpha)^{m-s}\overline{\alpha}\overline{b}^t.$$

where $1 \leq s \leq m$ and $t \geq n$. The word $z_{s,t}$ has m b-blocks, where the s-th b-block and the $\overline{b}$-block are of length t and all other b-blocks are of length n.

We let $t = n! + n$ and apply Ogden's Lemma to the word $z_{s,t}$ where n consecutive positions in the $\overline{b}$-block are designated as distinguished. Hence we find a derivation

$$S \overset{*}{\Longrightarrow} uAy \overset{*}{\Longrightarrow} uvAxy \overset{*}{\Longrightarrow} uvwxy = z_{s,t}$$

where v is covered by the $\overline{b}$-block or x is covered by the $\overline{b}$-block (and the covered factor is at most of length n). By pumping upwards we see that x is covered by the $\overline{b}$-block, v is covered by the s-th b-block, and $|v| \geq |x|$. By pumping downwards we obtain $|v| = |x|$. In the following we call $A \overset{*}{\Longrightarrow} uAv$ a *derivation loop*.

First assume there are two different derivation loops of this kind in $S \overset{*}{\Longrightarrow} z_{s,t}$ and they are not cyclic shifts of powers of the same smaller derivation loop. That means we may write the derivation as

$$S \overset{*}{\Longrightarrow} u_1 A y_1 \overset{*}{\Longrightarrow} u_1 v_1 A x_1 y_1 \overset{*}{\Longrightarrow} u_1 v_1 w_1 x_1 y_1 = z_{s,t} \quad \text{or}$$
$$S \overset{*}{\Longrightarrow} u_2 B y_2 \overset{*}{\Longrightarrow} u_2 v_2 B x_2 y_2 \overset{*}{\Longrightarrow} u_2 v_2 w_2 x_2 y_2 = z_{s,t}$$

where $1 \leq |v_i| = |x_i| \leq n$, v_i is covered by the s-th b-block, x_i is covered by the $\overline{b}$-block (for $i = 1, 2$), and taking the A-loop $|v_2| + 1$ times leads to a different derivation than taking the B-loop $|v_1| + 1$ times. Though, both derivations create the word $z_{s,t+|v_1|\cdot|v_2|}$. This implies that the word $z_{s,t+|v_1|\cdot|v_2|\cdot m}$ has a degree of ambiguity of at least m.

Now we assume that for every $1 \leq s \leq m$ we do not find a situation as above. Hence $S \overset{*}{\Longrightarrow} z_{s,t}$ uses a derivation loop $A_s \overset{*}{\Longrightarrow} v A_s x$ at least $\frac{n!}{|v|} = \ell$ times (where v is covered by the s-th b-block, x is covered by the $\overline{b}$-block, and $1 \leq |v| = |x| \leq n$):

$$S \overset{*}{\Longrightarrow} u A_s y \overset{*}{\Longrightarrow} u v A_s x y \overset{*}{\Longrightarrow} u v^\ell A_s x^\ell y \overset{*}{\Longrightarrow} u v^\ell w x^\ell y = z_{s,t}.$$

This derivation exists since there are less than n positions in the $\overline{b}$-block which are not pumpable, by Ogden's Lemma, and all pumpable positions use the same derivation loop, by assumption.

This yields a derivation for $z_{s,n} = uwy$ which uses the non-terminal A_s. Note that for $1 \leq r \leq n$ and $r \neq s$ we have $z_{s,n} = z_{r,n}$, but we cannot have that a derivation of $z_{s,n}$ uses A_s and A_r at the same time; otherwise we could pump one time at A_s and one time at A_r and obtain a word where the $\overline{b}$-block is longer than each b-block. We conclude, the word $z_{s,n}$ has at least m different derivations. □

5 The Iterated Bounded Hairpin Completion

In this section we consider the iterated parameterized hairpin completion with finite bounds. The main result of this section is that the regular languages are effectively closed under iterated bounded hairpin completion, which was stated as an open problem in [17]. Yet we will provide a more general result:

Theorem 5.1. *Let L be a formal language and $\ell, r \in \mathbb{N}$. The iterated parameterized hairpin completion $\mathcal{H}_k^*(L, \ell, r)$ can be effectively represented by an expression using L and the operations union, intersection with regular sets, and concatenation with regular sets.*

A proof for the theorem can be found in Section 5.1.

Consequentially, all language classes which are closed under these operations are also closed under iterated parameterized hairpin completion with finite bounds, and if the closure under all three operations is effective, then the closure under iterated parameterized hairpin completion with finite bounds is effective, too; this applies to all four Chomsky classes. From [16,17] it is known that the classes of context-free, context-sensitive, and recursively enumerable languages are closed under iterated bounded hairpin completion, but the status for regular languages was unknown. Since the iterated bounded hairpin completion is a special case of the iterated parameterized hairpin completion with finite bounds we can answer this question now.

Corollary 5.2. *Let $\mathcal{C}$ be a class of languages. If $\mathcal{C}$ is closed under union, intersection with regular sets, and concatenation with regular sets, then $\mathcal{C}$ is also closed under iterated bounded hairpin completion. Moreover, if $\mathcal{C}$ is effectively closed under union, intersection with regular sets, and concatenation with regular sets, then the closure under iterated bounded hairpin completion is effective.*

In particular, the class of regular languages is effectively closed under iterated bounded hairpin completion.

In Section 5.2 we investigate the sizes of NFAs that accept the iterated bounded hairpin completion of regular languages and we provide a exponential lower and upper bound. We also discuss how our results may be adapted to solve the membership problem for the iterated bounded hairpin completion of a regular language.

5.1 Representation

This section is devoted to the proof of Theorem 5.1. First we introduce the concept of *α-prefixes* which is essential for the following proof.

5.1.1 α-Prefixes

Let α be a word of length k. For $v, w \in \Sigma^*$ we say v is an α-prefix of w if $v\alpha \leq w$. We denote the set of all α-prefixes of length at most ℓ by

$$P_\alpha(w, \ell) = \{v \mid v\alpha \leq w \wedge |v| \leq \ell\}.$$

The idea behind this notation is: For a word $w \in \alpha\Sigma^*\overline{\alpha}$ with $|w| - k \geq \ell, r$, the set of (non-iterated) parameterized hairpin completions of w is given by

$$\mathcal{H}_\alpha(\{w\}, \ell, 0) = P_\alpha(\overline{w}, \ell)w \qquad \text{and} \qquad \mathcal{H}_\alpha(\{w\}, 0, r) = w\overline{P_\alpha(w, r)}.$$

In the following proof we are interested in α-prefixes of words which have α as a prefix. This leads to some useful properties.

Lemma 5.3. *Let $\alpha \in \Sigma^k$, $\ell \in \mathbb{N}$, and $w \in \alpha\Sigma^*$.*

i.) For all $v \in P_\alpha(w, \ell)$ we have $\alpha \leq v\alpha$.

ii.) For all $u, v \in P_\alpha(w, \ell)$ we have

$$|u| \leq |v| \iff u\alpha \leq v\alpha \iff u \in P_\alpha(v\alpha, \ell).$$

iii.) If $v\alpha$ is a prefix of some word in $P_\alpha(w, \ell)^*\alpha$, then $v \in P_\alpha(w, \ell)^*$.

Proof. If two words x, y are prefixes of w and $|x| \leq |y|$, then $x \leq y$. This yields *i.)* and *ii.)*.

For *iii.)* let $v\alpha \leq x_1 \cdots x_m\alpha$ where $x_1, \ldots, x_m \in P_\alpha(w, \ell)$. We can factorize $v = x_1 \cdots x_{i-1}y$ such that $y \leq x_i$ for some i with $1 \leq i \leq m$. By *i.)* and induction, we see that α is a prefix of $x_{i+1} \cdots x_m\alpha$ and hence $y\alpha \leq x_i\alpha \leq w$ which implies $y \in P_\alpha(w, \ell)$ and, moreover, $v \in P_\alpha(w, \ell)^*$. □

5.1.2 Proof of Theorem 5.1

Let L be a formal language and $\ell, r \in \mathbb{N}$. We will state an effective representation for $\mathcal{H}_k^*(L, \ell, r)$ using L and the operations union, intersection with regular sets, and concatenation with regular sets.

Let us begin with a basic observation. Every word w which is a hairpin completion of some other word has a factorization $w = \delta\beta\overline{\delta}$ with $|\delta| \geq k$, therefore, the prefix of w of length k and the suffix of w of length k are complementary. Let us call this prefix α, hence, we have $w \in \alpha\Sigma^*\overline{\alpha}$. Every word which is a right hairpin completion of w has still the prefix α and since the suffix of length k is complementary, it has the suffix $\overline{\alpha}$ as well. For left hairpin completions we have a symmetric argument and, by induction, every word which is an iterated hairpin completion of w has prefix α and suffix $\overline{\alpha}$.

Thus, we can split up the (non-iterated) parameterized hairpin completion $\mathcal{H}_k(L, \ell, r)$ into finitely many languages $L_\alpha = \mathcal{H}_k(L, \ell, r) \cap \alpha\Sigma^*\overline{\alpha}$ where $\alpha \in \Sigma^k$, and each of them has a effective representation using L and the operations union, intersection with regular sets, and concatenation with regular sets. Moreover,

$$\mathcal{H}_k^*(L_\alpha, \ell, r) = \mathcal{H}_\alpha^*(L_\alpha, \ell, r) \subseteq \alpha\Sigma^*\overline{\alpha}$$

and the iterated parameterized hairpin completion equals

$$\begin{aligned}
\mathcal{H}_k^*(L, \ell, r) &= L \cup \mathcal{H}_k^*(\mathcal{H}_k(L, \ell, r), \ell, r) \\
&= L \cup \mathcal{H}_k^*\Big(\bigcup_{\alpha \in \Sigma^k} L_\alpha, \ell, r\Big) \\
&= L \cup \bigcup_{\alpha \in \Sigma^k} \mathcal{H}_\alpha^*(L_\alpha, \ell, r).
\end{aligned}$$

Henceforth, let $\alpha \in \Sigma^k$ be fixed. In order to prove Theorem 5.1 we will state a suitable representation for $\mathcal{H}^*_\alpha(L_\alpha, \ell, r)$. For the rest of the proof we will heavily rely on the fact that every word in $\mathcal{H}^*_\alpha(L_\alpha, \ell, r)$ has the prefix α and the suffix $\overline{\alpha}$. The representation is defined recursively. We have

$$\mathcal{H}^*_\alpha(L_\alpha, 0, 0) = L_\alpha.$$

By symmetry, we may assume that $\ell \geq r$ and $\ell \geq 1$. We will state a representation for $\mathcal{H}^*_\alpha(L_\alpha, \ell, r)$ using $\mathcal{H}^*_\alpha(L_\alpha, \ell-1, r)$ and the operations union, intersection with regular sets, and concatenation with regular sets. Therefore, consider a word

$$z \in \mathcal{H}^*_\alpha(L_\alpha, \ell, r) \setminus \mathcal{H}^*_\alpha(L_\alpha, \ell-1, r).$$

For some $n \geq 1$ there is a sequence $w_0, \ldots, w_n = z$ where $w_0 \in L_\alpha$ and for all i such that $1 \leq i \leq n$ either w_i is a left hairpin completion of w_{i-1} and $|w_i| \leq |w_{i-1}|+\ell$ or w_i is a right hairpin completion of w_{i-1} and $|w_i| \leq |w_{i-1}|+r$. Furthermore, there is an index $j \geq 1$ such that $w_{j-1} = w \in \mathcal{H}^*_\alpha(L_\alpha, \ell-1, r)$ and $w_j = vw \notin \mathcal{H}^*_\alpha(L_\alpha, \ell-1, r)$. Note that this implies $|v| = \ell$ and $w \in \alpha\Sigma^*\overline{\alpha}\,\overline{v}$. Let $s = n - j$ and consider the factorization

$$z = x_s \cdots x_1 v w \overline{y_1} \cdots \overline{y_s}$$

where $x_i \cdots x_1 v w \overline{y_1} \cdots \overline{y_i} = w_{j+i}$ and either

1. $y_i = 1$, $|x_i| \leq \ell$, and $x_i\alpha \leq y_{i-1} \cdots y_1 v\alpha$ or
2. $x_i = 1$, $|y_i| \leq r$, and $y_i\alpha \leq x_{i-1} \cdots x_1 v\alpha$.

for all i such that $0 \leq i \leq s$.

The crucial point is that vw has the prefix $v\alpha$, the suffix $\overline{\alpha}\,\overline{v}$, and $|v| = \ell \geq r$. Therefore, the factors $x_1, \ldots, x_s$ and $y_1, \ldots, y_s$ are controlled by the triple (v, ℓ, r) in the following way:

Lemma 5.4. *$x_i \in P_\alpha(v\alpha, \ell)^*$ and $y_i \in P_\alpha(v\alpha, r)^*$ for all i such that $1 \leq i \leq s$.*

Proof. We prove the claim by induction on i. Let i such that $1 \leq i \leq s$. Our induction hypothesis is $x_j \in P_\alpha(v\alpha, \ell)^*$ and $y_j \in P_\alpha(v\alpha, r)^*$ for all j such that $1 \leq j < i$. We distinguish between the two cases above:

1. We have $y_i = 1 \in P_\alpha(v\alpha, r)^*$ and, by induction hypothesis,

$$x_i\alpha \leq y_{i-1} \cdots y_1 v\alpha \in P_\alpha(v\alpha, r)^* v\alpha \subseteq P_\alpha(v\alpha, \ell)^*\alpha.$$

In combination with Lemma 5.3 this yields $x_i \in P_\alpha(v\alpha, \ell)^*$.

2. We have $x_i = 1 \in P_\alpha(v\alpha, \ell)^*$ and

$$y_i\alpha \leq x_{i-1} \cdots x_1 v\alpha \in P_\alpha(v\alpha, \ell)^*\alpha,$$

hence $y_i \in P_\alpha(v\alpha, \ell)^*$. Since $|y_i| \leq r$, all factors of y_i are at most of length r, too, and $y_i \in P_\alpha(v\alpha, r)^*$. □

For $u \in \Sigma^\ell$ let us define the language

$$L(u, \ell, r) = P_\alpha(u\alpha, \ell)^* u \, (\mathcal{H}_\alpha^*(L_\alpha, \ell - 1, r) \cap \alpha\Sigma^*\overline{\alpha}\overline{u}) \, \overline{P_\alpha(u\alpha, r)}^*.$$

Note that, by induction, for every u the representation for $L(u, \ell, r)$ is effectively given. By Lemma 5.4, the word z is included in $L(v, \ell, r)$ and for every word $z' \in \mathcal{H}_\alpha^*(L_\alpha, \ell, r) \setminus \mathcal{H}_\alpha^*(L_\alpha, \ell - 1, r)$ there exists $v' \in \Sigma^\ell$ such that $z' \in L(v', \ell, r)$. Therefore,

$$\mathcal{H}_\alpha^*(L_\alpha, \ell, r) \subseteq \mathcal{H}_\alpha^*(L_\alpha, \ell - 1, r) \cup \bigcup_{u \in \Sigma^\ell} L(u, \ell, r)$$

and for the right hand side we have an effective representation. Of course, we intend to replace the inclusion by an equality sign.

Lemma 5.5. *$L(u, \ell, r) \subseteq \mathcal{H}_\alpha^*(L_\alpha, \ell, r)$ for all $u \in \Sigma^\ell$.*

Proof. We start by proving a special case of the claim that is successfully used later to derive the result. Consider a word w' together with the factorization

$$w' = x_m \cdots x_1 w \overline{y_1} \cdots \overline{y_n}$$

with $m \geq 0$, $n \geq 1$ and where for some word $u \in \Sigma^*$

1. $w \in \mathcal{H}_\alpha^*(L_\alpha, \ell, r) \cap u\alpha\Sigma^*\overline{\alpha}\overline{u}$,
2. $x_1, \ldots, x_m \in P_\alpha(u\alpha, \ell)$,
3. $y_1, \ldots, y_n \in P_\alpha(u\alpha, r)$, and
4. $m = 0$ or $|y_j| \leq |x_m|$ for all j such that $1 \leq j \leq n$.

We claim $w' \in \mathcal{H}_\alpha^*(L_\alpha, \ell, r)$, too. Indeed, if $m = 0$, it is plain that w' is an n-iterated right hairpin completion of w. Otherwise $x_m \cdots x_1 w$ is an m-iterated left hairpin completion of w. By the fourth property and Lemma 5.3, we have $y_1, \ldots, y_n \in P_\alpha(x_m\alpha, r)$. Hence, w' is an n-iterated right hairpin completion of $x_m \cdots x_1 w$ and we conclude $w' \in \mathcal{H}_\alpha^*(L_\alpha, \ell, r)$.

Now, let $u \in \Sigma^\ell$ and $z \in L(u, \ell, r)$. There is a factorization

$$z = x_s \cdots x_1 w \overline{y_1} \cdots \overline{y_t}$$

where

1. $w \in u\,(\mathcal{H}^*_\alpha(L, \ell-1, r) \cap \alpha\Sigma^*\overline{\alpha}\overline{u}) \subseteq \mathcal{H}^*_\alpha(L_\alpha, \ell, r) \cap u\alpha\Sigma^*\overline{\alpha}\overline{u}$,
2. $x_1, \ldots, x_s \in P_\alpha(u\alpha, \ell)$, and
3. $y_1, \ldots, y_t \in P_\alpha(u\alpha, r)$.

If $t = 0$, the word z is an s-iterated left hairpin completion of w. Otherwise, let $n \geq 1$ be the maximal index such that $|y_n| \geq |y_j|$ for all $1 \leq j \leq t$, and let m be the maximal index such that $|y_n| \leq |x_m|$ or 0 if no such index exists. Let $w' = x_m \cdots x_1 w \overline{y_1} \cdots \overline{y_n}$. Note that w' satisfies the conditions of the special case we discussed above and hence $w' \in \mathcal{H}^*_\alpha(L_\alpha, \ell, r)$.

With $u' = y_n$ we obtain

$$z = x_s \cdots x_{m+1} w' \overline{y_{n+1}} \cdots \overline{y_t}$$

where, by the choice of n, m and by Lemma 5.3,

1. $w' \in \mathcal{H}^*_\alpha(L_\alpha, \ell, r) \cap u'\alpha\Sigma^*\overline{\alpha}\overline{u'}$,
2. $x_{m+1}, \ldots, x_s \in P_\alpha(u'\alpha, \ell)$, and
3. $y_{n+1}, \ldots, y_t \in P_\alpha(u'\alpha, r)$.

At this point we may continue inductively and deduce $z \in \mathcal{H}^*_\alpha(L_\alpha, \ell, r)$. □

The previous lemma tells us, if $\ell \geq r$, the iterated parameterized hairpin completion of L_α can be represented by

$$\mathcal{H}^*_\alpha(L_\alpha, \ell, r) = \mathcal{H}^*_\alpha(L_\alpha, \ell-1, r) \cup \bigcup_{u \in \Sigma^\ell} L(u, \ell, r).$$

Symmetrically, if $r > \ell$, let us define

$$R(u, \ell, r) = P_\alpha(u\alpha, \ell)^*\,(\mathcal{H}^*_\alpha(L_\alpha, \ell, r-1) \cap u\alpha\Sigma^*\overline{\alpha})\,\overline{u}\overline{P_\alpha(u\alpha, r)}^*.$$

The iterated parameterized hairpin completion of L_α can be represented by

$$\mathcal{H}^*_\alpha(L_\alpha, \ell, r) = \mathcal{H}^*_\alpha(L_\alpha, \ell, r-1) \cup \bigcup_{u \in \Sigma^r} R(u, \ell, r).$$

We conclude, the iterated parameterized hairpin completion of a language L can be represented by an effective expression using L and the operations union, intersection with regular sets, and concatenation with regular sets.

5.2 The Size of NFAs

Let L be a regular language and $\ell, r \in \mathbb{N}$ be finite bounds. In this section we analyze the size of NFAs accepting the iterated parameterized hairpin completion $\mathcal{H}_k^*(L, \ell, r)$ with respect to the size of an NFA accepting L and the bounds ℓ and r. By the size of an NFA we mean its number of states. Recall that k is treated as a constant. (Assuming $k \leq \ell$ or $k \leq r$ would induce the same complexity, but this is not shown here.) Our results are the following.

Theorem 5.6.

i.) Let $m \geq 1$. There is a regular language L such that neither the language $\mathcal{H}_k(L, m, m)$ nor the language $\mathcal{H}_k^*(L, m, m)$ can be detected by an NFA with less than 2^m states.

ii.) Let L be a regular language which is accepted by an NFA of size n. Let $\ell, r \in N$ and let $m = \max\{\ell, r\}$. There is an NFA accepting the iterated parameterized hairpin completion $\mathcal{H}_k^*(L, \ell, r)$ whose size is in $2^{\mathcal{O}(m^2)}n$.

Proof of i. Let $\Sigma = \{a, \overline{a}, b, \overline{b}, c, \overline{c}\}$ and $L = c\{\overline{a}, \overline{b}\}^* a^k \overline{a}^k$. For any word $w \in L$ there is no possibility of building hairpin on the left and the only possible hairpin on the right is to bind the suffix $\overline{a}^k$ to a^k if $|w| \leq m + 2k$. Therefore, we have

$$\mathcal{H}_k(L, m, m) = \bigcup_{v \in \{\overline{a}, \overline{b}\}^{\leq m-1}} cva^k\overline{a}^k\overline{v}\overline{c}.$$

Now let $w = cva^k\overline{a}^k\overline{v}\overline{c}$ with $v \in \{\overline{a}, \overline{b}\}^{\leq m-1}$. The only way to build a hairpin is to bind its prefix to its suffix, hence

$$\mathcal{H}_k^*(L, m, m) = L \cup \mathcal{H}_k(L, m, m).$$

We claim that an NFA accepting $\mathcal{H}_k(L, m, m)$ or $\mathcal{H}_k^*(L, m, m)$ has a size of at least 2^m. We prove the claim for the language $\mathcal{H}_k(L, m, m)$; the argumentation for $\mathcal{H}_k^*(L, m, m)$ is exactly the same.

Consider an NFA accepting $\mathcal{H}_k(L, m, m)$ and let $\mathcal{Q}$ denote its set of states. For a word $u \in \Sigma^*$ we denote by $\mathcal{P}(u) \subseteq \mathcal{Q}$ the set of states which are reachable from an initial state with a path labelled by u. Now let $v \in \{\overline{a}, \overline{b}\}^{\leq m-1}$. Since $cva^k\overline{a}^k\overline{v}\overline{c} \in \mathcal{H}_k(L, m, m)$, there is a state $q \in \mathcal{P}(cva^k\overline{a}^k)$ such that a path from q to a final state exists which is labelled by $\overline{v}\overline{c}$. For all words $u \in \{\overline{a}, \overline{b}\}^{\leq m-1}$ with $u \neq v$ the state q does not belong to $\mathcal{P}(cua^k\overline{a}^k)$ because $cua^k\overline{a}^k\overline{v}\overline{c} \notin \mathcal{H}_k(L, m, m)$. Each word $v \in \{\overline{a}, \overline{b}\}^{\leq m-1}$ yields such a state q, they are mutually different, and none of them is an initial state (as $\overline{v}\overline{c} \notin \mathcal{H}_k(L, m, m)$). Therefore, the number of states $|\mathcal{Q}|$ has to be greater than $\left|\{\overline{a}, \overline{b}\}^{\leq m-1}\right| = 2^m - 1$. □

In order to prove the second claim of Theorem 5.6 we implicitly use some well-known constructions of NFAs which accept concatenation, union, or intersection of regular languages. Consider two NFAs which accept the languages L_1, L_2 and which are of size n_1, n_2, respectively. There is an NFA accepting the concatenation L_1L_2 which is of size $n_1 + n_2$, an NFA accepting the union $L_1 \cup L_2$ which is of size $n_1 + n_2$, and an NFA accepting the intersection $L_1 \cap L_2$ which is of size $n_1 \cdot n_2$. For details on how these NFAs may be constructed see, e.g., [15].

Proof of ii. Let L be a regular language which is accepted by an automaton of size n and let $\ell, r \in \mathbb{N}$. The parameterized hairpin completion of L is given by

$$\mathcal{H}_k(L, \ell, r) = \bigcup_{\alpha \in \Sigma^k} \bigcup_{\gamma \in \Sigma^{\leq \ell}} \gamma(\alpha \Sigma^* \overline{\alpha}\overline{\gamma} \cap L) \cup \bigcup_{\alpha \in \Sigma^k} \bigcup_{\gamma \in \Sigma^{\leq r}} (\gamma \alpha \Sigma^* \overline{\alpha} \cap L)\overline{\gamma}.$$

For $\gamma, \alpha \in \Sigma^*$ there is an NFA accepting $\gamma(\alpha\Sigma^*\overline{\alpha}\overline{\gamma} \cap L)$ which has a size in $\mathcal{O}(|\gamma\alpha| \cdot n)$. Hence, the parameterized hairpin completion of L can be accepted by an NFA which has a size in $\mathcal{O}(|\Sigma|^m \, m \cdot n) \subseteq 2^{\mathcal{O}(m)}n$ where $m = \max\{\ell, r\}$.

For $\alpha \in \Sigma^k$ the language $L_\alpha = \mathcal{H}_k(L, \ell, r) \cap \alpha\Sigma^*\overline{\alpha}$ can also be accepted by an NFA which has a size in $2^{\mathcal{O}(m)}n$. Let $N_{i,j}$ denote the minimal size of an NFA accepting $\mathcal{H}^*_\alpha(L_\alpha, i, j)$ for $i, j \in \mathbb{N}$. Since $\mathcal{H}_k(L_\alpha, 0, 0) = L_\alpha$, we have $N_{0,0} \in 2^{\mathcal{O}(m)}n$. For $i \geq j$ let us recall that

$$\mathcal{H}^*_\alpha(L_\alpha, i, j) = \mathcal{H}^*_\alpha(L_\alpha, i-1, j) \cup \bigcup_{u \in \Sigma^i} L(u, i, j),$$

$$L(u, i, j) = P_\alpha(u\alpha, \ell)^* u \, (\mathcal{H}^*_\alpha(L_\alpha, i-1, j) \cap \alpha\Sigma^*\overline{\alpha}\overline{u}) \, \overline{P_\alpha(u\alpha, r)}^*.$$

The size of a minimal NFA accepting $L(u, i, j)$ is in $\mathcal{O}(i \cdot N_{i-1,j})$ whence

$$N_{i,j} \in \mathcal{O}(|\Sigma|^i \, i \cdot N_{i-1,j}) \subseteq 2^{\mathcal{O}(i)}N_{i-1,j}.$$

Symmetrically, for $j > i$ we have $N_{i,j} \in 2^{\mathcal{O}(i)}N_{i,j-1}$. By unfolding the recursion we obtain

$$N_{\ell,r} \in \prod_{i=1}^{\ell} 2^{\mathcal{O}(i)} \cdot \prod_{j=1}^{r} 2^{\mathcal{O}(j)} \cdot 2^{\mathcal{O}(m)}n = \prod_{i=1}^{m} 2^{\mathcal{O}(i)} \cdot n = 2^{\mathcal{O}(\sum_{i=1}^{m} i)}n = 2^{\mathcal{O}(m^2)}n.$$

Now, the iterated parameterized hairpin completion is given by

$$\mathcal{H}^*_k(L, \ell, r) = L \cup \bigcup_{\alpha \in \Sigma^k} \mathcal{H}^*_\alpha(L_\alpha, \ell, r)$$

and there is an NFA accepting $\mathcal{H}^*_k(L, \ell, r)$ which has a size in $\mathcal{O}(N_{\ell,r} + n) \subseteq 2^{\mathcal{O}(m^2)}n$. □

Statement 2 of Theorem 5.6 also yields an algorithm to solve the membership problem for the iterated bounded hairpin completion of a regular language.

Corollary 5.7. *Let L be a regular language, given by an NFA of size n, and let $\ell, r \in \mathbb{N}$. The problem whether an input word w belongs to $\mathcal{H}_k^*(L, \ell, r)$ can be decided in linear time $c \cdot |w|$, where the constant c depends on the size n and the bounds ℓ, r. More precisely, for $m = \max\{\ell, r\}$ we have $c \in 2^{\mathcal{O}(m^2)}n^2$.*

Proof. Following the proof of Statement 2 of Theorem 5.6, we can construct an NFA $\mathcal{A} = (\mathcal{Q}, \Sigma, E, \mathcal{I}, \mathcal{F})$ accepting the iterated hairpin completion $\mathcal{H}_k^*(L, \ell, r)$ which is of a size in $2^{\mathcal{O}(m^2)}n$. Let us denote the size of this NFA by N. Note that the construction can be preformed in time $\mathcal{O}(|E|) \subseteq \mathcal{O}(N^2) \subseteq 2^{\mathcal{O}(m^2)}n^2$.

The input w can be accepted by an online power-set construction of the NFA A: We start with the set of states $\mathcal{P}_0 = \mathcal{I}$. When we read the i-th letter a of the input w we construct the set $\mathcal{P}_i$ by following all outgoing edges of states in $\mathcal{P}_{i-1}$ which are labelled by a. As every state in $\mathcal{P}_{i-1}$ has at most N outgoing edges labelled by a, one step can be performed in $\mathcal{O}(N^2) \subseteq 2^{\mathcal{O}(m^2)}n^2$ time. The algorithm stops after w is read and $\mathcal{P}_{|w|}$ is computed. The input w belongs to $\mathcal{H}_k^*(L, \ell, r)$ if and only if $\mathcal{P}_{|w|}$ contains a final state from $\mathcal{F}$. □

So far, the best known time complexity of the membership problem for the iterated (unbounded) hairpin completion of a regular language L is quadratic with respect to the length of the input word, by an algorithm from [26]. This algorithm can easily be adapted to solve the membership problem for the iterated bounded hairpin completion in quadratic time. Hence, if we measure the time complexity with respect to the length of the input word only, we have an improvement from quadratic to linear time (in the bounded case).

6 Iterated Hairpin Completions of Singletons

The class of iterated hairpin completions of singletons is defined as

$$\mathrm{HCS}_k = \{\mathcal{H}_k^*(\{w\}) \mid w \in \Sigma^*\}.$$

We solve the problem whether HCS_k includes non-regular or non-context-free languages, which was asked in [31], by stating a singleton whose iterated hairpin completion is not context-free. Furthermore, we show that the result also holds if we consider the one-sided case.

As we treat the unbounded case again, the length of the factor γ of a hairpin completion with the usual factorization is no longer bounded by a constant. Note that, by the results of Section 5, the possibility of creating arbitrary long prefixes or suffixes has to play an essential role in following proof.

Theorem 6.1. *The iterated one- and two-sided hairpin completions of a singleton are in* NL *but not context-free, in general.*

Proof. The membership to NL follows by the fact that NL is closed under iterated bounded hairpin completion, which has been proved in [6]. For convenience, we give a sketch of the proof, here.

Consider a language $L \in \mathrm{NL}$. The iterated hairpin completion $\mathcal{H}_k^*(L)$ can be accepted by a non-deterministic Turing machine that works as follows. We use two pointers i and j which mark the beginning and the end of a factor of the input w, respectively.

1. We start with $i = 1$ and $j = |w|$, hence $w[i, j] = w$.
2. Non-deterministically either continue with step 3 or skip to step 5.
3. Either guess i' such that $i < i' < j$ and verify that $w[i, j]$ is a left hairpin completion of $w[i', j]$ or guess j' such that $i < j' < j$ and verify that $w[i, j]$ is a right hairpin completion of $w[i, j']$. If the verification is successful, continue with $i = i'$ (resp. $j = j'$).
4. Repeat step 2.
5. Accept if and only if $w[i, j] \in L$.

Obviously, this Turing machine accepts $\mathcal{H}_k^*(L)$. In order to perform step 1-4, we only have to store some pointers on the input word; this can be done in $\log |w|$ space. Since $L \in \mathrm{NL}$ step 5 can be performed in $\log |w|$ space, too, and hence $\mathcal{H}_k^*(L) \in \mathrm{NL}$.

For the one-sided hairpin completions $\mathcal{H}_k^*(L, \infty, 0)$ and $\mathcal{H}_k^*(L, 0, \infty)$ we can use almost the same algorithm. The only difference is that the pointer i is always 1 (resp. j is always $|w|$).

We will now state a singleton $\{w\}$ and proof that the iterated two-sided and right-sided hairpin completions $\mathcal{H}_k^*(\{w\})$ and $\mathcal{H}_k^*(\{w\}, 0, \infty)$ are not context-free. There is an analogous proof for the left-sided case; we just have to use the singleton $\{\overline{w}\}$. Let $\Sigma = \{a, \overline{a}, b, \overline{b}, c, \overline{c}\}$, $\alpha = a^k$, and

$$w = \alpha b \alpha \overline{\alpha} \alpha c \overline{\alpha}.$$

Since context-free languages are closed under intersection with regular languages, it suffices to show for a regular language R that the intersections $R \cap \mathcal{H}_k^*(\{w\})$ and $R \cap \mathcal{H}_k^*(\{w\}, 0, \infty)$ are not context-free. Let $u = \overline{b}\overline{\alpha}$ and $v = \alpha\overline{\alpha}\overline{b}\overline{\alpha}$. Note that $\overline{u}\alpha \leq \overline{v}\alpha \leq w$. Let

$$R = w u^+ v \overline{u}^+ \overline{w} \overline{u}^+ \overline{w}$$

and consider a word $z \in R$. For some $r, s, t \geq 1$ we have

$$z = \underbrace{\alpha b \alpha \overline{\alpha} \alpha c \overline{\alpha}}_{w} \underbrace{(\overline{b}\overline{\alpha})^r}_{u^r} \underbrace{\alpha \overline{\alpha} \overline{b} \overline{\alpha}}_{v} \underbrace{(\alpha b)^s}_{\overline{u}^s} \underbrace{\alpha \overline{c} \overline{\alpha} \alpha \overline{\alpha} \overline{b} \overline{\alpha}}_{\overline{w}} \underbrace{(\alpha b)^t}_{\overline{u}^t} \underbrace{\alpha \overline{c} \overline{\alpha} \alpha \overline{\alpha} \overline{b} \overline{\alpha}}_{\overline{w}}.$$

At first, note that w is a prefix of z and it does not occur as another factor in z (there is only one c in z). Thus, if z belongs to $\mathcal{H}_k^*(\{w\})$, it must be an iterated right hairpin completion of w and hence

$$R \cap \mathcal{H}_k^*(\{w\}) = R \cap \mathcal{H}_k^*(\{w\}, 0, \infty).$$

Next, we will show that z is an iterated hairpin completion of w if and only if $r = s = t$. The proof is a straight forward construction of z. We try to find a sequence $w = w_0, w_1, \ldots, w_n = z$ for some $n \geq 0$ where $w_i \neq w_{i-1}$ is a right hairpin completion of w_{i-1} for $1 \leq i \leq n$. This implies that every w_i is a prefix of z.

Fortunately, for each of the words $w_0, \ldots, w_{r+1}$ there is exactly one choice which satisfies these conditions:

$$\begin{aligned}
w_0 = w &= \alpha b \alpha \overline{\alpha} \alpha c \overline{\alpha} \\
w_1 = wu &= \alpha b \alpha \overline{\alpha} \alpha c \overline{\alpha} \overline{b} \overline{\alpha} \\
w_2 = wu^2 &= \alpha b \alpha \overline{\alpha} \alpha c \overline{\alpha} (\overline{b} \overline{\alpha})^2 \\
&\vdots \qquad \vdots \\
w_r = wu^r &= \alpha b \alpha \overline{\alpha} \alpha c \overline{\alpha} (\overline{b} \overline{\alpha})^r \\
w_{r+1} = wu^r v &= \alpha b \alpha \overline{\alpha} \alpha c \overline{\alpha} (\overline{b} \overline{\alpha})^r \alpha \overline{\alpha} \overline{b} \overline{\alpha}
\end{aligned}$$

If $s \neq r$, none of the right hairpin completions of w_{r+1} is a prefix of z (except for w_{r+1} itself). Otherwise, we find exactly one right hairpin completion which satisfies the conditions:

$$w_{r+2} = wu^r v \overline{u}^r \overline{w} = \alpha b \alpha \overline{\alpha} \alpha c \overline{\alpha} (\overline{b} \overline{\alpha})^r \alpha \overline{\alpha} \overline{b} \overline{\alpha} (\alpha b)^r \alpha \overline{c} \overline{\alpha} \alpha \overline{\alpha} \overline{b} \overline{\alpha}.$$

The argument for the last step is the same. If and only if $t = r$, we find a prefix of z which is a right hairpin completion of w_{r+2} and this is $w_{r+3} = z$.

We conclude z is an iterated hairpin completion of w if and only if $r = s = t$ and hence

$$R \cap \mathcal{H}_k^*(\{w\}) = \{ wu^r v \overline{u}^r \overline{w} \overline{u}^r \overline{w} \mid r \geq 1 \}.$$

The intersection $R \cap \mathcal{H}_k^*(\{w\})$ belongs to a family of context-sensitive languages which are well known to be non-context-free. From this it follows that $\mathcal{H}_k^*(\{w\})$ and $\mathcal{H}_k^*(\{w\}, 0, \infty)$ are non-context-free, too. □

7 Final Remarks and Open Problems

We proved that the regularity problem for hairpin completions of regular languages is decidable and that it is NL-complete. In particular, it can be solved efficiently in parallel, because NL is contained in *Nick's Class* $\mathbf{NC}_2$, see

e. g., [34]. The time performance, we provide, is $\mathcal{O}(n^8)$ where n is the size of the input DFAs. This seems quite large, however, in the algorithm we construct an automaton $\mathcal{A}$ that is already of size n^4. Hence, with respect to the size of $\mathcal{A}$ we only use quadratic time which seems optimal as we consider pairs of states in this automaton and it is unclear how to avoid this bound. For an improvement of the asymptotical time bound a completely new approach is probably required.

We also considered the hairpin completions of varieties of languages. We showed that if the hairpin completion of a language recognized by a monoid in **A** (resp. **LDA**) is regular, then its hairpin completion is recognized by a monoid in **A** (resp. **LDA**), too. Another natural variety of languages is induced by the variety **DA**, which is a proper subclass of **LDA**. This class is also characterized by first-order formulae in $\mathrm{FO}^2[<]$. By Example 3.1 there are languages within this class whose hairpin completion is not regular. It is open whether we can extend our result to the variety **DA**.

We extended the decidability of the regularity problem to the one-sided hairpin lengthening of regular languages. However, the regularity problem for the (two-sided) hairpin lengthening of regular languages is still far open. Our results indicate that this problem is more difficult to solve than the regularity problem for hairpin completions, as we showed that the hairpin lengthening of regular languages may be inherent ambiguous linear, whereas the hairpin completion of regular languages is unambiguous linear.

For the iterated bounded hairpin completion we proved that all classes which are (effectively) closed under quite basic operations are also (effectively) closed under iterated bounded hairpin completion. Together with our the result that the iterated hairpin completion of even a single word may lead to a non-context-free language, this shows that the length bounds are a strong restriction for the iterated hairpin completion. Note that for the iterated hairpin lengthening a length bound means no restriction as one long hairpin lengthening step can be obtained by a series of smaller hairpin lengthening steps, see [27]. It is open whether the regular languages are closed under iterated hairpin lengthening.

As we proved that the iterated hairpin completion of a singleton may be non-context-free, two new questions arise naturally. Does a singleton exist whose iterated hairpin completion is context-free but not regular? Can we decide for a given singleton whether its iterated hairpin completion is non-regular (or non-context-free)?

Another interesting problem is, whether the iterated hairpin completion of two languages have a common element. Even for two given singletons it is not known, if this problem is decidable at all, see [31]. The result of Section 6 shows that this is a non-trivial question. However, in the bounded case we can decide this problem for two regular languages. We just need to create the

NFAs and test whether the intersection is empty. As the size of the NFAs are exponential with respect to the length bounds, this does not seem to be the best way to decide the problem.

Publications

Theorem 3.2 and 3.3 and the results in Section 3.2 have been obtained by joined work with Volker Diekert and have been presented at the CIAA 2010 [8]. The result of NL-completeness (Theorem 3.2) has been submitted to the special issue of the IJFCS dedicated to CIAA 2010 and a preprint is available on arXiv:1101.4824.

These results are an improvement of our former approach which has been obtained by a joined work with Volker Diekert and Victor Mitrana and was presented at the ICTAC 2009 [9]. Back then, we proved the polynomial time performance without being precise on the degree of the polynomial. In my diploma thesis I used this first approach to prove that the time performance is bounded by a polynomial of degree 14.

The results in Section 5 and 6 have been presented at the DLT 2010 [20] and have been accepted for publication in TCS in 2011 [21]; a preprint is also available on arXiv:1010.3640.

The results in Section 3.1, 3.3, and 4 have not yet been published elsewhere.

References

[1] J. Almeida. A syntactical proof of locality of **DA**. *International Journal of Algebra and Computation*, 6(2):165–177, 1996.

[2] C. Àlvarez and B. Jenner. A note on logspace optimization. *Comput. Complex.*, 5:155–166, April 1995.

[3] B. S. Baker and R. V. Book. Reversal-bounded multi-pushdown machines. *Foundations of Computer Science, Annual IEEE Symposium on*, 0:207–211, 1972.

[4] J. Berstel and Ch. Reutenauer. *Rational series and their languages*. Springer-Verlag New York, Inc., New York, NY, USA, 1988.

[5] T. Ceccherini-Silberstein. On the growth of linear languages. *Advances in Applied Mathematics*, 35(3):243 – 253, 2005.

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

[7] R. Deaton, R. Murphy, M. Garzon, D. Franceschetti, and S. Stevens. Good encodings for DNA-based solutions to combinatorial problems. *Proc. of DNA-based computers DIMACS Series*, 44:247–258, 1998.

[8] V. Diekert and S. Kopecki. Complexity results and the growths of hairpin completions of regular languages (extended abstract). In M. Domaratzki and K. Salomaa, editors, *CIAA*, volume 6482 of *Lecture Notes in Computer Science*, pages 105–114. Springer, 2010.

[9] V. Diekert, S. Kopecki, and V. Mitrana. On the hairpin completion of regular languages. In M. Leucker and C. Morgan, editors, *ICTAC*, volume 5684 of *Lecture Notes in Computer Science*, pages 170–184. Springer, 2009.

[10] M. Garzon, R. Deaton, P. Neathery, R. Murphy, D. Franceschetti, and E. Stevens. On the encoding problem for DNA computing. *The Third DIMACS Workshop on DNA-Based Computing*, pages 230–237, 1997.

[11] M. Garzon, R. Deaton, L. Nino, S. Stevens Jr., and M. Wittner. Genome encoding for DNA computing. *Proc. Third Genetic Programming Conference*, pages 684–690, 1998.

[12] P. Gawrychowski, D. Krieger, N. Rampersad, and J. Shallit. Finding the growth rate of a regular or context-free language in polynomial time. In *Developments in Language Theory*, pages 339–358, 2008.

[13] S. A. Greibach. A note on undecidable properties of formal languages. *Mathematical Systems Theory*, 2(1):1–6, 1968.

[14] M. Hagiya, M. Arita, D. Kiga, K. Sakamoto, and S. Yokoyama. Towards parallel evaluation and learning of boolean μ-formulas with molecules. In *Second Annual Genetic Programming Conf.*, pages 105–114, 1997.

[15] J. E. Hopcroft and J. D. Ulman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[16] M. Ito, P. Leupold, F. Manea, and V. Mitrana. Bounded hairpin completion. *Inf. Comput.*, 209:471–485, March 2011.

[17] M. Ito, P. Leupold, and V. Mitrana. Bounded hairpin completion. In *LATA '09: Proceedings of the 3rd International Conference on Language and Automata Theory and Applications*, pages 434–445, Berlin, Heidelberg, 2009. Springer-Verlag.

[18] L. Kari, S. Konstantinidis, E. Losseva, P. Sosík, and G. Thierrin. Hairpin structures in DNA words. In A. Carbone and N. A. Pierce, editors,

DNA, volume 3892 of *Lecture Notes in Computer Science*, pages 158–170. Springer, 2005.

[19] L. Kari, K. Mahalingam, and G. Thierrin. The syntactic monoid of hairpin-free languages. *Acta Inf.*, 44(3-4):153–166, 2007.

[20] S. Kopecki. On the iterated hairpin completion. In Y. Gao, H. Lu, S. Seki, and S. Yu, editors, *Developments in Language Theory*, volume 6224 of *Lecture Notes in Computer Science*, pages 438–439. Springer Berlin / Heidelberg, 2010.

[21] S. Kopecki. On the iterated hairpin completion. *Theoretical Computer Science*, 2011. Accepted Manuscript.

[22] W. Kuich. On the entropy of context-free languages. *Information and Control*, 16:173–200, 1970.

[23] P. M. Lewis, R. E. Stearns, and J. Hartmanis. Memory bounds for recognition of context-free and context-sensitive languages. In *Proceedings of the 6th Annual Symposium on Switching Circuit Theory and Logical Design (SWCT 1965)*, FOCS '65, pages 191–202, Washington, DC, USA, 1965. IEEE Computer Society.

[24] K. Lodaya, P. K. Pandya, and S. S. Shah. Around dot depth two. In *Proc. of the 14th Int. Conf. on Developments in Language Theory (DLT'10)*, volume 6224, pages 305–316. Springer-Verlag, 2010.

[25] F. Manea. A series of algorithmic results related to the iterated hairpin completion. *Theor. Comput. Sci.*, 411(48):4162–4178, 2010.

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

[27] F. Manea, C. Martín-Vide, and V. Mitrana. Hairpin lengthening. In F. Ferreira, B. Löwe, E. Mayordomo, and L. M. Gomes, editors, *CiE*, volume 6158 of *Lecture Notes in Computer Science*, pages 296–306. Springer, 2010.

[28] F. Manea and V. Mitrana. Hairpin completion versus hairpin reduction. In S. B. Cooper, B. Löwe, and A. Sorbi, editors, *CiE*, volume 4497 of *Lecture Notes in Computer Science*, pages 532–541. Springer, 2007.

[29] F. Manea, V. Mitrana, and T. Yokomori. Some remarks on the hairpin completion. In E. Csuhaj-Varju and Z. Esik, editors, *12th International Conference AFL 2008 Proceedings*, pages 302–312, 2008.

[30] F. Manea, V. Mitrana, and T. Yokomori. Two complementary operations inspired by the DNA hairpin formation: Completion and reduction. *Theor. Comput. Sci.*, 410(4-5):417–425, 2009.

[31] F. Manea, V. Mitrana, and T. Yokomori. Some remarks on the hairpin completion. *Int. J. Found. Comput. Sci.*, 21(5):859–872, 2010.

[32] R. McNaughton and S. Papert. *Counter-Free Automata*. The MIT Press, Cambridge, Mass., 1971.

[33] W. Ogden. A helpful result for proving inherent ambiguity. *Mathematical Systems Theory (now called: Theory of Computing Systems)*, 2:191–194, 1968. 10.1007/BF01694004.

[34] Ch. H. Papadimitriou. *Computatational Complexity*. Addison Wesley, 1994.

[35] J.-É. Pin. *Varieties of Formal Languages*. North Oxford Academic, London, 1986.

[36] K. Sakamoto, H. Gouzu, K. Komiya, D. Kiga, S. Yokoyama, T. Yokomori, and M. Hagiya. Molecular computation by DNA hairpin formation. *Science*, 288(5469):1223–1226, 2000.

[37] K. Sakamoto, D. Kiga, K. Komiya, H. Gouzu, S. Yokoyama, S. Ikeda, and M. Hagiya. State transitions by molecules, 1998.

[38] R. Stearns. A regularity test for pushdown machines. *Information and Control*, 11(3):323 – 340, 1967.

[39] H. Straubing. *Finite automata, formal logic, and circuit complexity*. Birkhauser Verlag, Basel, Switzerland, 1994.

[40] R. E. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.

[41] L. G. Valiant. Regularity and related problems for deterministic pushdown automata. *Journal of the ACM*, 22, 1975.

[42] E. Winfree. Whiplash PCR for O(1) computing. In *University of Pennsylvania*, pages 175–188, 1998.