Fundamenta Informaticae 71 (2006) 453–475
IOS Press

A Formal Language Analysis of DNA Hairpin Structures

L. Kari and E. Losseva
Department of Computer Science
The University of Western Ontario
London, ON, N6A 5B7, Canada
{lila,elena}@csd.uwo.ca

S. Konstantinidis
Dept. of Mathematics and Computing Science
Saint Mary's University
Halifax, Nova Scotia, B3H 3C3, Canada
s.konstantinidis@smu.ca

P. Sosík*
Facultad de Informática
Universidad Politécnica de Madrid
Campus de Montegancedo s/n
Boadilla del Monte 28660, Madrid, Spain
and Institute of Computer Science
Silesian University, 74601 Opava
Czech Republic
petr.sosik@fpf.slu.cz

G. Thierrin
Department of Mathematics
The University of Western Ontario
London, ON, N6A 5B7, Canada
thierrin@uwo.ca

Abstract. The concept of hairpin structures in formal languages is motivated from the biocomputing and bioinformatics fields. Hairpin (-free) DNA structures have numerous applications to DNA computing and molecular genetics in general. A word is called hairpin-free if it cannot be written in the form $xvy\theta(v)z$, with certain additional conditions, for an involution θ (a function θ with the property that θ^2 equals the identity function). A particular involution, the so-called Watson-Crick involution, can characterize binding of two DNA strands.

We study algebraic and decision properties, finiteness and descriptional complexity of hairpin (-free) languages. We show an existence of polynomial-time algorithms deciding hairpin-freeness of regular and context-free sets. Two related DNA secondary structures are considered, taking into the account imperfect bonds (bulges, mismatches) and multiple hairpins. Finally, effective methods for design of long hairpin-free DNA words are given.

Keywords: DNA computing, DNA hairpin, involution, formal language

*Address for correspondence: Facultad de Informática, Universidad Politécnica de Madrid, Campus de Montegancedo s/n, Boadilla del Monte 28660, Madrid, Spain

1. Introduction

The primary motivation for the study of hairpin-free structures arises from the areas of DNA computing and bioinformatics, where such structures are important for the design of information-encoding DNA molecules. The reader is referred to [1, 22] for an overview of the DNA computing paradigm. A single strand of deoxyribonucleic acid (DNA) consists of a sugar-phosphate backbone and a sequence of nucleotides attached to it. This sequence is oriented and, by convention, its beginning and its end are called 5' and 3' end, respectively. There are four types of nucleotides denoted by A, C, T, and G. Two single strands can bind (anneal) to each other if they have opposite polarity (strand's orientation) and are pairwise Watson-Crick complementary: A is complementary to T, and C to G. The ability of DNA strands to anneal to each other allows for creation of various secondary structures. A *DNA hairpin* is a particular type of secondary structure important in many applications. An example of a hairpin structure is shown in Figure 1.

Figure 1. A single-stranded DNA molecule forming a hairpin loop.

Hairpin-like secondary structures play an important role in insertion/deletion operations with DNA. Hairpin-freeness is crucial in the design of primers for the PCR reaction [6]. Hairpins are the main tool used in the Whiplash PCR computing techniques [25]. In [27] hairpins serve as a binary information medium for DNA RAM. Last, but not least, hairpins are basic components of recently investigated "smart drugs" [3]. Therefore, in the above mentioned applications, one needs to construct (sets of) hairpin (-free) DNA molecules, or to test existing sets of DNA molecules for hairpin-freeness and study their properties. We refer e.g. to [16, 17, 20, 21, 23] for a characterization and design of DNA languages with or without hairpins and other undesired bonds. Coding properties of hairpin-free languages have been studied in [14, 15], and their language-theoretic characterizations in [23]. Hairpins have also been studied in the context of bio-operations occurring in single-celled organisms (see the hairpin inversion operation defined as one of the three molecular operations that accomplish gene assembly in ciliates [8, 10]).

In addition, the results presented in this paper also contribute to mathematical characterization of regularities in formal words and languages. In this sense the definition of hairpin-free words can be understood as a generalization of repetition-freeness. A word u is called *hairpin-k-free* if $u = xvy\theta(v)z$ implies $|v| < k$, for a chosen involution θ. Considering the special case when $k = 1$, θ is the identity involution and y is the empty word, we obtain the square-freeness (see below).

For a general overview and fundamental results in combinatorics on words, the reader is referred to [7, 18]. If w is a nonempty word, then ww is called a square and www is called a cube. Important questions about avoiding squares and cubes in infinite words have been answered in [9]. See [19] for combinatorics on finite words. Words of the form $uvyvz$ with a bounded length of y have been studied e.g. in [5]. Unfortunately, many techniques and results known in combinatorics on words are non-applicable in the case of hairpin-free words. One of the main reasons is that in the case of an antimorphic involution, analogies of the famous defect theorem and its consequences are no longer valid.

The paper is organized as follows. Section 2 introduces basic formal concepts and definitions. In Section 3 we present the concept of hairpin-free words and languages and study their elementary properties. Problems related to finiteness of hairpin-free languages are addressed in Section 4, and decision problems in Section 5. In Section 6 we study descriptional complexity of hairpin (-free) languages with respect to possible applications. In the next section, two important variants of the hairpin definition are studied. The first one takes into the account imperfect DNA bonds (mismatches, bulges), the second one is related to hairpin-based nanomachines. Finally, effective methods for design of long hairpin-free DNA words are given in Section 8.

2. Formal language prerequisites

We will use X to denote a finite alphabet and X^* its corresponding free monoid. The cardinality of the alphabet X is denoted by $|X|$. For a word w, $|w|$ denotes the length of w. The empty word is denoted by 1, and $X^+ = X^* - \{1\}$. A *language* is an arbitrary subset of X^*. For a word $w \in X^*$ and $k \geq 0$, we denote by w^k the word obtained as catenation of k copies of w. Similarly, X^k is the set of all words from X^* of length k. By convention, $w^0 = 1$ and $X^0 = \{1\}$. We also denote $X^{\leq k} = X^0 \cup X^1 \cup \cdots \cup X^k$. By convention, $X^{\leq k} = \emptyset$ for $k < 0$.

A mapping $\psi : X^* \to X^*$ is called a *morphism (anti-morphism)* of X^* if $\psi(uv) = \psi(u)\psi(v)$ (respectively $\psi(uv) = \psi(v)\psi(u)$) for all $u, v \in X^*$, and $\psi(1) = 1$. See [12] for a general overview of morphisms. An involution $\theta : X \longrightarrow X$ is defined as a map such that θ^2 is the identity function. An involution θ can be extended to a morphism or an antimorphism over X^*. In both cases θ^2 is the identity over X^* and $\theta^{-1} = \theta$. If not stated otherwise, θ refers to an arbitrary morphic or antimorphic involution in this paper.

In our examples we shall refer to the DNA alphabet $\Delta = \{A, C, T, G\}$. By convention, DNA strands are described by strings over this alphabet in orientation from 5' to 3' end. On this alphabet several involutions of interest are defined. The simplest involution is the identity function ϵ. An antimorphic involution which maps each letter of the alphabet to itself is called a *mirror involution* and it is denoted by μ. The DNA *complementarity involution* γ is a morphism given by $\gamma(A) = T$, $\gamma(T) = A$, $\gamma(C) = G$, $\gamma(G) = C$. For example, $\epsilon(ACGCTG) = ACGCTG = \mu(GTCGCA) = \gamma(TGCGAC)$.

Finally, the antimorphic involution $\tau = \mu\gamma$ (the composite function of μ and γ, which is also equal to $\gamma\mu$), called the *Watson-Crick involution*, corresponds to the DNA bond formation of two single strands. If for two strings $u, v \in \Delta^*$ it is the case that $\tau(u) = v$, then the two DNA strands represented by u, v anneal as Watson-Crick complementary sequences.

A nondeterministic finite automaton (*NFA*) is a quintuple $M = (S, X, s_0, F, P)$, where S is the finite and nonempty set of states, s_0 is the start state, F is the set of final states, and P is the set of productions of the form $sx \to t$, for $s, t \in S$, $x \in X$. If for every two productions $sx_1 \to t_1$ and $sx_2 \to t_2$ of an NFA we have that $x_1 \neq x_2$ then the automaton is called a *DFA* (deterministic finite automaton). The language accepted by the automaton M is denoted by $L(M)$. The *size* $|M|$ of the automaton M is the number $|S| + |P|$.

An analogous notation we use for a *pushdown automaton (PDA)* and a *deterministic pushdown automaton (DPDA)*. We refer the reader to [13, 26] for their definitions and for further elements of formal language theory.

3. Involutions and hairpins

Definition 3.1. If θ is a morphic or antimorphic involution of X^* and k is a positive integer, then a word $u \in X^*$ is said to be θ-k-hairpin-free or simply hp(θ,k)-free if $u = xvy\theta(v)z$ for some $x, v, y, z \in X^*$ implies $|v| < k$.

Notice that the words 1 and $a \in X$ are hp(θ,1)-free. More generally, words of length less than $2k$ are hp(θ,k)-free. If we interpret this definition for the DNA alphabet Δ and the Watson-Crick involution τ, then a hairpin structure with the length of bond (i.e., the number of complementary nucleotide pairs bound together) greater than or equal to k is a word that is not hp(τ,k)-free.

Definition 3.2. Denote by $hpf(\theta, k)$ the set of all hp(θ,k)-free words in X^*. The complement of $hpf(\theta, k)$ is $hp(\theta, k) = X^* - hpf(\theta, k)$.

Notice that $hp(\theta, k)$ is the set of words in X^* which are hairpins of the form $xvy\theta(v)z$ where the length of v is at least k. It is also the case that $hp(\theta, k+1) \subseteq hp(\theta, k)$ for all $k > 0$.

Definition 3.3. A language L is called θ-k-hairpin-free or simply hp(θ, k)-free if $L \subseteq hpf(\theta, k)$.

It is easy to see from the definition that a language L is hp(θ, k)-free if and only if $X^*vX^*\theta(v)X^* \cap L = \emptyset$ for all v with $|v| \geq k$. An analogous definition was given in [14] where a θ-k-hairpin-free language is called θ-subword-k-code. The authors focused on their coding properties and relations to other types of codes. Restrictions on the length of a hairpin were also considered, namely that $1 \leq |y| \leq m$ for some $m \geq 1$. The reader can verify that our Proposition 3.3 remains valid and the results in Section 6 change only slightly if we apply this additional restriction.

Example.

1. Let $X = \{a, b\}$ with $\theta(a) = b, \theta(b) = a$. Then $hpf(\theta, 1) = a^* \cup b^*$.

   This example shows that in general the product of hp$(\theta, 1)$-free words is not an hp$(\theta, 1)$-free word. Indeed, a and b are hp$(\theta, 1)$-free, but the product ab is not.

2. If $\theta = \gamma$ is the DNA complementary involution over Δ^*, then:

$$hpf(\theta, 1) = \{A, C\}^* \cup \{A, G\}^* \cup \{T, C\}^* \cup \{T, G\}^*$$

3. Let $\theta = \mu$ be the mirror involution and let $u \in hpf(\theta, 1)$. Since $\theta(a) = a$ for all $a \in X$, u cannot contain two occurrences of the same letter a. This implies that $hpf(\theta, 1)$ is finite. For example, if $X = \{a, b\}$, then:

$$hpf(\theta, 1) = \{1, a, b, ab, ba\}$$

We focus first on the important special case when $k = 1$. Observe that $hp(\theta, 1) = \bigcup_{a \in X} X^*aX^*\theta(a)X^*$. Recall also the definition of an embedding order: $u \leq_e w$ if and only if

$$u = u_1u_2 \cdots u_n, \ w = v_1u_1v_2u_2 \cdots v_nu_nv_{n+1}$$

for some integer n with $u_i, v_j \in X^*$.

A language L is called *right $\leq_e$-convex* [11, 28] if $u \leq_e w, \ u \in L$ implies $w \in L$. The following result is well known: *All languages (over a finite alphabet) that are right $\leq_e$-convex are regular.*

Proposition 3.1. The language $hp(\theta, 1)$ is right $\leq_e$-convex.

Proof:
If $u = u_1u_2 \in hp(\theta, 1)$ and $v_1, v_2, v_3 \in X^*$, then $w = v_1u_1v_2u_2v_3 \in hp(\theta, 1)$. Therefore, if $u \in hp(\theta, 1)$ and $u \leq_e w$, then w can be constructed from u by a sequence of insertions, and hence $w \in hp(\theta, 1)$. □

Let $L \subseteq X^*$ be a nonempty language and let:

$$S(L) = \{w \in X^* \mid u \leq_e w, u \in L\}.$$

Hence $S(L)$ is the set of all the words $w \in X^*$ in the form $w = x_1u_1x_2u_2 \cdots x_nu_nx_{n+1}$ with $u = u_1u_2 \cdots u_n \in L$ and $x_i \in X^*$.

Recall further that a set H with $\emptyset \neq H \subseteq X^+$ is called a *hypercode* over X^* *iff* $x \leq_e y$ and $x, y \in H$ imply $x = y$. That is, a hypercode is an independent set with respect to the embedding order.

Proposition 3.2. Let θ be a morphic or antimorphic involution. Then there exists a unique hypercode H such that $hp(\theta, 1) = S(H)$.

Proof:
Let $H = \bigcup_{a \in X} a\theta(a)$, then $S(H) = \bigcup_{a \in X} X^*aX^*\theta(a)X^* = hp(\theta, 1)$. The uniqueness of H is immediate. □

Example. Consider the hypercodes for the earlier three examples.

1. For $X = \{a, b\}$ and the involution (morphic or antimorphic) $\theta(a) = b$, $\theta(b) = a$, the hypercode is $H = \{ab, ba\}$.

2. For the DNA complementarity involution γ we have $H = \{AT, TA, CG, GC\}$.

3. The mirror involution over $\{a, b\}^*$ gives the hypercode $H = \{aa, bb\}$.

Proposition 3.1, true for the case $k = 1$, cannot in general be extended to the case $k > 1$ as the language $hp(\theta, 2)$ is not $\leq_e$-convex. However, the weaker regularity property remains valid.

Proposition 3.3. The languages $hp(\theta, k)$ and $hpf(\theta, k)$, $k \geq 1$, are regular.

Proof:
One can easily derive $hp(\theta, k) = \bigcup_{|w| \geq k} X^*wX^*\theta(w)X^* = \bigcup_{|w|=k} X^*wX^*\theta(w)X^*$. Every language $X^*wX^*\theta(w)X^*$with $|w| = k$ is regular, hence $hp(\theta, k)$ is a union of a finite number of regular languages. Therefore both $hp(\theta, k)$ and its complement $hpf(\theta, k)$ are regular. □

4. Finiteness of hairpin-free languages

In this section we give the necessary and sufficient conditions under which the language $hpf(\theta, k)$ is finite, for a chosen $k \geq 1$. We study first the interesting special case of μ, the mirror involution, over a binary alphabet X.

Recall that $hp(\mu, k)$ is the set of all words containing two non-overlapping mirror parts of length at least k. In the next proposition we show that the longest hp$(\mu, 4)$-free word is of length 31. This also implies that the language $hpf(\mu, 4)$ is finite. The proof requires several technical lemmata whose proofs can be found in the Appendix. In these lemmata we assume that $|X| = 2$.

Definition 4.1. A *run* in a word w is a subword of w of the form c^k, with $c \in X$ and $k \geq 1$, such that $w = uc^k v$ for some word u that does not end with c, and some word v that does not start with c. If both u and v are nonempty the run is called *internal*.

Lemma 4.1. Suppose that w is a word in $hpf(\mu, 4)$. The following statements hold true.

1. If a^i is any run in w then $i \leq 7$. If the run is internal then $i \leq 5$.
2. The word w cannot contain three different runs $a^{i_1}, a^{i_2}, a^{i_3}$ with $i_1, i_2, i_3 \geq 3$. If w contains two runs a^j and a^i with $i, j \geq 3$ then w starts with $a^j b a^i$, or w ends with $a^i b a^j$. Moreover not both i and j can be greater than 3.
3. The word w cannot contain three different internal runs a^2. If w contains two internal runs a^2 then they occur as in $\cdots b a^2 b a^2 b \cdots$.
4. The above statements also hold if we replace a with b and vice-versa.

Lemma 4.2. Suppose that a word in $hpf(\mu, 4)$ contains a subword w of the form

$$a b^{x_1} a^{y_1} \cdots b^{x_n} a^{y_n} b,$$

with $n \geq 3$ and $x_i, y_i \geq 1$ for each i. Then there are at most three indices i such that $x_i = y_i = 1$.

Lemma 4.3. Suppose that a word w is in $hpf(\mu, 4)$ and contains two runs c^j and c^i with $i, j \geq 3$ and $c \in X$. Then $|w| \leq 31$.

Lemma 4.4. Suppose that a word w is in $hpf(\mu, 4)$ and contains no two runs c^j and c^i with $i, j \geq 3$ and contains two internal runs b^2 and one internal run b^y with $y \geq 3$ and w is of the following form

$$a^{y_0} b^{x_1} a^{y_1} \cdots b^{x_n} a^{y_n} (b^{x_{n+1}} a^{y_{n+1}}),$$

where all y_i's and x_j's are positive except possibly for y_{n+1}. Then $|w| \leq 31$.

Lemma 4.5. If a word w is in $hpf(\mu, 4)$ and of the form

$$a^{y_0} b^{x_1} a^{y_1} \cdots b^{x_n} a^{y_n} (b^{x_{n+1}} a^{y_{n+1}}),$$

such that $y_0, x_{n+1} \geq 3$, and $2 \geq y_{n+1} \geq 0$, and $2 \geq x_i, y_i > 0$ for all $i = 1, \ldots, n$, then $|w| \leq 30$. Moreover, the following word of length 30 satisfies the above premises:

$$a^7 b^2 a b^2 a b a b a b a^2 b a^2 b^7.$$

Proposition 4.1. Let X be a binary alphabet. For every word $w \in X^*$ in $hpf(\mu, 4)$ we have that $|w| \leq 31$. Moreover the following word of length 31 is in $hpf(\mu, 4)$

$$a^7ba^3bababab^2ab^2a^2b^7.$$

Proof:
Without loss of generality we can assume that w starts with a. Then w would be of the form

$$a^{y_0}b^{x_1}a^{y_1}\cdots b^{x_n}a^{y_n}(b^{x_{n+1}}a^{y_{n+1}}),$$

where all y_i's and x_j's are positive except possibly for y_{n+1}. We distinguish the following cases.

Case 1: There are two runs c^i and c^j in w with $i, j \geq 3$. By Lemma 4.3, $|w| \leq 31$ as required.

In the next 7 cases, we assume that the first case does not hold and that there is exactly one run a^δ in w with $\delta \geq 3$.

Case 2: The run a^δ is a^{y_0} and there is a run b^i with $i \geq 3$. If $x_{n+1} \geq 3$ then Lemma 4.5 implies that $|w| \leq 30$. So assume that $x_{n+1} \leq 2$. If there are two internal runs b^2 in w then Lemma 4.4 implies that $|w| \leq 31$. So assume further that there is at most one internal run b^2. Note that if $x_{n+1} = 2$ and $y_{n+1} > 0$ then $b^{x_{n+1}}$ is the run b^2. Let g be the quantity $e|b^2a| + x_{n+1} + y_{n+1}$, where $e = 0$ if $x_{n+1} = 2$ and $y_{n+1} > 0$, and $e = 1$ if $x_{n+1} = 1$ or $y_{n+1} = 0$. Hence, $g \leq 6$. Moreover, $|w| \leq 7 + 3|ba| + 2|ba^2| + |b^5a| + g \leq 31$.

Case 3: The run a^δ is a^{y_0} and there is no run b^i with $i \geq 3$. Using again the quantity g of Case 2, we have that $|w| \leq 7 + 3|ba| + 2|ba^2| + |b^2a| + g \leq 28$.

Case 2': The run a^δ is $a^{y_{n+1}}$ and there is a run b^i with $i \geq 3$. Then the word $\mu(w)$ is of the same form as the word w is and the run b^i occurs in $\mu(w)$. Hence, Case 2 applies to $\mu(w)$ and, therefore, both $\mu(w)$ and w are of length at most 31.

Case 3': The run a^δ is $a^{y_{n+1}}$ and there is no run b^i with $i \geq 3$. Then the word $\mu(w)$ is of the same form as the word w is and no run b^i, with $i \geq 3$, occurs in $\mu(w)$. Hence, Case 3 applies to $\mu(w)$ and, therefore, both $\mu(w)$ and w are of length at most 28.

Case 4: The run a^δ is internal and there is one internal run b^j with $j \geq 3$. Then $j, \delta \leq 5$. If w contains two internal runs b^2 then Lemma 4.4 implies that $|w| \leq 31$. Next assume that w contains at most one internal run b^2 and consider the quantity $g = e|b^2a| + x_{n+1} + y_{n+1}$ as in Case 2. If w contains at most one internal run a^2 then

$$|w| \leq y_0 + 3|ba| + |ba^2| + |ba^5| + |b^5a| + g \leq 2 + 6 + 3 + 6 + 6 + 6 = 29.$$

Next assume further that w contains two internal runs a^2. Then Lemma 4.1 implies that w contains ba^2ba^2b. Also,

$$|w| \leq 2 + 6 + 2|ba^2| + 6 + 6 + g \leq 26 + g.$$

If $x_{n+1} = 2$ and $y_{n+1} > 0$ then $e = 0$ and $|w| \leq 30$. If $y_{n+1} = 0$ then $e = 1$ and $|w| \leq 31$. If $x_{n+1} = 1$ and $y_{n+1} = 1$ then $|w| \leq 31$. Finally, if $x_{n+1} = 1$ and $y_{n+1} = 2$ then w ends with aba^2, which contradicts the fact that w contains ba^2ba^2b.

Case 4': The run a^δ is internal and there is one external run b^j with $j \geq 3$. Then $y_{n+1} = 0$ and the run b^j is $b^{x_{n+1}}$, as $y_0 > 0$. Let w' be the word resulting by exchanging the letters a and b in w. Then the word $\mu(w')$ satisfies the premises of Case 2, which implies that w is of length at most 31.

Case 5: The run a^δ is internal and there is no run b^j with $j \geq 3$. Using again the quantity g of Case 2, we have that $|w| \leq y_0 + 3|ba| + |ba^5| + 2|ba^2| + |b^2a| + g \leq 29$.

Case 6: Here the first case does not hold and there is no run a^δ with $\delta \geq 3$. If there is an internal run b^j with $j \geq 3$ then $|w| \leq y_0 + 3|ba| + |b^5a| + 2|ba^2| + |b^2a| + g \leq 29$. If there is an external run b^j with $j \geq 3$ then $b^j = b^{x_{n+1}}$ and $y_{n+1} = 0$, and one can verify that $|w| \leq 27$. If there is no run b^j with $j \geq 3$ then one can verify that $|w| \leq 23$.

Finally, by inspection one verifies that $a^7ba^3bababab^2ab^2a^2b^7$ is indeed in $hpf(\mu, 4)$. □

Corollary 4.1. Consider a binary alphabet X. Then $hpf(\mu, k)$ is finite if and only if $k \leq 4$.

Proof:
Let $X = \{a, b\}$. By Proposition 4.1, the set $hpf(\mu, 4)$ is finite. Consider the language $L_5 = (aabbab)^+$. Its subwords of length 5 form the set $Sub_5(L_5) = \{aabba, abbab, bbaba, babaa, abaab, baabb\}$. For its mirror image $\mu(L_5)$ we obtain $Sub_5(\mu(L_5)) = \{abbaa, babba, ababb, aabab, baaba, bbaab\}$. As these two sets are mutually disjoint, $L_5 \subseteq hpf(\mu, 5)$.

Finally, notice that for $k > 1$, finiteness of $hpf(\mu, k)$ implies also finiteness of $hpf(\mu, k-1)$. Hence the facts that $hpf(\mu, 4)$ is finite and $hpf(\mu, 5)$ is infinite conclude the proof. □

Proposition 4.2. Let θ be a morphic or antimorphic involution. The language $hpf(\theta, k)$ over a non-singleton alphabet X is finite if and only if one of the following holds:
(a) $\theta = \epsilon$, the identity involution;
(b) $\theta = \mu$, the mirror involution, and either $k = 1$ or $|X| = 2$ and $k \leq 4$.

Proof:

(a) Let θ be a morphism. Assume first that $\theta \neq \epsilon$. Then there are $a, b \in X$, $a \neq b$, such that $\theta(a) = b$. Then $a^+ \subseteq hpf(\theta, k)$ for any $k \geq 1$, hence $hpf(\theta, k)$ is infinite.

Assume now that $\theta = \epsilon$ and let w be any word of length $\geq k|X|^k + k$. Since there exist $|X|^k$ distinct words of length k, there are at least two non-overlapping subwords of length k in w which are identical. Hence $w = xvyvz$ for some $v \in X^k$ and $x, y, z \in X^*$. Therefore $hpf(\epsilon, k)$ is finite since it cannot contain any word longer than $k|X|^k + k$.

(b) Let θ be an anti-morphism. Assuming that $\theta \neq \mu$, the same arguments as above show that $hpf(\theta, k)$ is infinite.

Assume now that $\theta = \mu$. Apparently $hpf(\mu, 1)$ is finite as shown in the examples above. For $|X| = 2$ we know that $hpf(\mu, k)$ is finite *iff* $k \leq 4$ by Corollary 4.1. Finally, for $|X| > 2$ and $k > 1$ the language $hpf(\mu, k)$ is infinite as it always contains the hp$(\mu, 2)$-free set $(abc)^+$ (regardless to renaming the symbols).

□

5. Decision problems

Proposition 3.3 suggests the existence of fast algorithms solving some problems important from the practical point of view. We investigate two such problems now. Let θ be a fixed morphic or antimorphic involution and let $k \geq 1$ be an arbitrary but fixed integer (in practice the value of k would be small).

Hairpin-Freedom Problem.

Input: A nondeterministic automaton M.

Output: Yes/No depending on whether $L(M)$ is hp(θ, k)-free.

Maximal Hairpin-Freedom Problem.

Input: A deterministic automaton M_1 accepting a hairpin-free language, and a NFA M_2.

Output: Yes/No depending on whether there is a word $w \in L(M_2) - L(M_1)$ such that $L(M_1) \cup \{w\}$ is hp(θ, k)-free.

In what follows, we assume that M and M_1 are finite automata in the case of regular languages, and pushdown automata in the case of context-free languages. M_2 is always a nondeterministic finite automaton.

Proposition 5.1. The hairpin-freeness problem for regular languages is decidable in linear time (w.r.t. $|M|$).

Proof:
By definition, $L(M)$ is hp(θ, k)-free *iff* $L(M) \subseteq hpf(\theta, k)$ *iff* $L(M) \cap hp(\theta, k) = \emptyset$. Denote by M_k a NFA accepting the language $hp(\theta, k)$. One can construct an NFA M' of size $\mathcal{O}(|M_k| \cdot |M|)$, accepting the language $L(M) \cap hp(\theta, k)$. Recall that the size of M' is given by the number of its states and transitions (see Section 2). Hence the problem whether $L(M') = \emptyset$ is solvable in time $\mathcal{O}(|M'|) = \mathcal{O}(|M_k| \cdot |M|)$. The automaton M_k is fixed for a chosen k. □

Proposition 5.2. The maximal hairpin-freeness problem for regular languages is decidable in time proportional to $|M_1| \cdot |M_2|$.

Proof:
We want to determine whether there exists a word $w \in hpf(\theta, k)$ such that $w \notin L(M_1)$, but $w \in L(M_2)$. It is decidable in time $\mathcal{O}(|M_1| \cdot |M_2| \cdot |M'_k|)$ whether $(hpf(\theta, k) \cap L(M_2)) - L(M_1) = \emptyset$. The size of an NFA accepting $hpf(\theta, k)$ is denoted by $|M'_k|$. The automaton M'_k is fixed for a chosen k. □

As an immediate consequence, for a given block code K of length l it is decidable in linear time with respect to $|K| \cdot l$, whether there is a word $w \in X^l - K$ such that $K \cup \{w\}$ is hp(θ, k)-free. This is of particular interest since the lab sets of DNA molecules form often a block code.

Notice also that for a finite set S of DNA sequences (which is the case of practical interest) the size of the automaton M (or M_1) is in the worst case proportional to the total length of all sequences in S.

Proposition 5.3. The hairpin-freeness problem for context-free languages is decidable in cubic time (w.r.t. $|M|$).

Proof:
As in the proof of Proposition 5.1, $L(M)$ is hp(θ, k)-free *iff* $L(M) \cap hp(\theta, k) = \emptyset$. Given a PDA M accepting $L(M)$ and an NFA M_k accepting $hp(\theta, k)$, there exist a PDA M' accepting the language $L(M) \cap hp(\theta, k)$. Furthermore, M' can be constructed in time $\mathcal{O}(|M_k| \cdot |M|)$ and so is its size. Details are left to the reader.

Let G be a CFG such that $L(G) = L(M')$. The standard construction of G in [13] (Theorem 7.31) takes cubic time w.r.t. $|M'|$. Finally, it is possible to decide in linear time w.r.t. $|G|$ (see Section 7.4.3 of [13]) whether $L(G) = \emptyset$ or not. □

Proposition 5.4. The maximal hairpin-freeness problem for deterministic context-free languages is decidable in time $\mathcal{O}((|M_1| \cdot |M_2|)^3)$.

Proof:
We want to determine if $\exists w \in hpf(\theta, k)$ such that $w \notin L(M_1)$, but $w \in L(M_2)$. Denote $M_1 = (Q_1, X, \Gamma, q_1, Z_0, F_1, P_1)$, and let $M_2' = (Q_2, X, q_2, F_2, P_2)$ be a NFA accepting the language $hpf(\theta, k) \cap L(M_2)$. Consider the PDA $M = (Q, X, \Gamma, q_0, Z_0, F, P)$, where $Q = Q_1 \times Q_2$, $q_0 = (q_1, q_2)$. For $p \in Q_1, q \in Q_2$, and $Z \in \Gamma$ we define:

(1) $(p, q)aZ \underset{P}{\rightarrow} (p', q')\alpha$ if and only if $paZ \underset{P_1}{\rightarrow} p'\alpha$ and $qa \underset{P_2}{\rightarrow} q'$,

(2) $(p, q)1Z \underset{P}{\rightarrow} (p', q)\alpha$ if and only if $p1Z \underset{P_1}{\rightarrow} p'\alpha$

Let $F = \{(p, q) \mid p \notin F_1 \text{ and } q \in F_2\}$. Then $L(M) = (hpf(\theta, k) \cap L(M_2)) - L(M_1)$, and the size of M is $\mathcal{O}(|M_1| \cdot |M_2|)$. Finally, as in the previous proof, we can decide in time $\mathcal{O}(|M|^3) = \mathcal{O}((|M_1| \cdot |M_2|)^3)$ whether $L(M)$ is empty or not. □

6. Descriptional complexity

In the algorithms presented in Section 5, a construction of automata (NFA or DFA) accepting the languages $hp(\theta, k)$ and $hpf(\theta, k)$ is crucial. Particularly, the sizes of these automata are multiplicative constants for time complexity of the algorithms in Section 5. To investigate descriptional complexity of the languages $hp(\theta, k)$ and $hpf(\theta, k)$, we recall the following technical tool from [4].

Definition 6.1. A set of pairs of strings $\{(x_i, y_i) \mid i = 1, 2, \ldots, n\}$ is called a *fooling set* for a language L if for any i, j in $\{1, 2, \ldots, n\}$,
(1) $x_i y_i \in L$, and
(2) if $i \neq j$ then $x_i y_j \notin L$ or $x_j y_i \notin L$.

Lemma 6.1. Let $\mathcal{F}$ be a fooling set of a cardinality n for a regular language L. Then any NFA accepting L needs at least n states.

Now we can characterize the minimal size of automata accepting languages $hp(\theta, k)$ and $hpf(\theta, k)$. We use the operator $\prod$ for catenation.

Proposition 6.1. The number of states of a minimal NFA accepting the language $hp(\theta, k)$, $k \geq 1$, over an alphabet X of the cardinality $\ell > 1$, is between ℓ^k and $3\ell^k$, and its size is at most $3(\ell^k + \ell^{k+1})$.

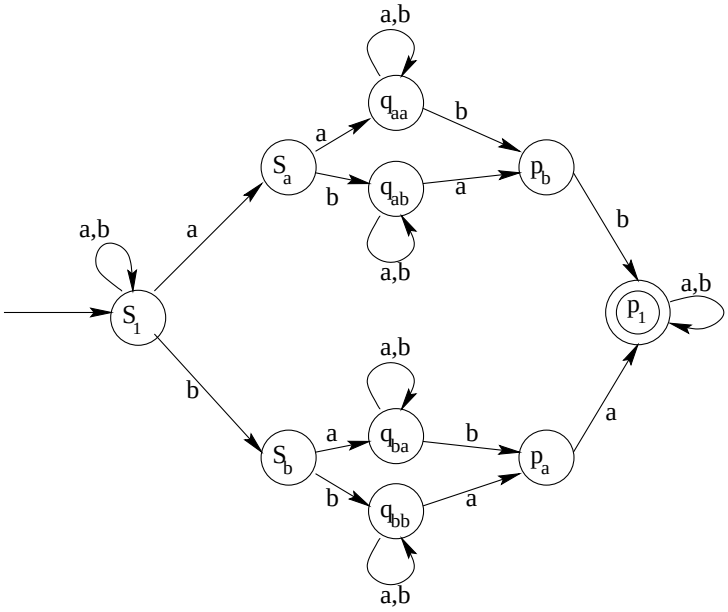

Figure 2. An example of an NFA accepting the language $hp(\theta, 2)$.

Proof:
Let $M_k = (S, X, s_1, F, P)$ be an NFA accepting $hp(\theta, k)$.

(i) The reader can easily verify that the set $\mathcal{F} = \{(w, \theta(w)) \mid w \in X^k\}$ is a fooling set for $hp(\theta, k)$. Therefore $|S| \geq \ell^k$.

(ii) Let

$$S = \{s_w, p_w \mid w \in X^{\leq k-1}\} \cup \{q_w \mid w \in X^k\}.$$

Let further $F = \{p_1\}$. The set of productions P is defined as follows:

$s_v a \to s_w$ if and only if $va = w$,	for each $v \in X^{\leq k-2},\ a \in X$;
$s_v a \to q_w$ if and only if $va = w$,	for each $v \in X^{k-1},\ a \in X$;
$q_w a \to q_w$	for all $w \in X^k,\ a \in X$;
$q_w a \to p_v$ if and only if $\theta(av) = w$,	for each $v \in X^{k-1},\ a \in X$;
$p_w a \to p_v$ if and only if $av = w$,	for each $v \in X^{\leq k-2},\ a \in X$.

Finally, let $s_1 a \to s_1$ and $p_1 a \to p_1$ for all $a \in X$. An example of the automaton M_k for the case $X = \{a, b\}$, $k = 2$ and θ being an antimorphism, $\theta(a) = b$, $\theta(b) = a$, is at Fig. 2. The reader can verify that $L(M_k) = hp(\theta, k)$, and that $|S| \leq 3\ell^k$, $|P| \leq 3\ell^{k+1}$, therefore $|M_k| \leq 3(\ell^k + \ell^{k+1})$. □

Note that for $\ell = 1$ we have $hp(\theta, k) = X^{2k}X^*$, therefore the size of the minimal automaton accepting $hp(\theta, k)$ is $|M_k| = 4k + 2$.

Proposition 6.2. Assume that there are distinct letters $a, b \in X$ such that $a = \theta(b)$. Then the number of states of a minimal NFA accepting $hpf(\theta, k)$, $k \geq 1$, over an alphabet X with the cardinality ℓ, is at least $2^{(\ell-2)^k/2}$.

Proof:
We take into the account only the cases $\ell \geq 3$, the case $\ell = 2$ is trivial. Denote $X_1 = X \setminus \{a, b\}$. We can factorize the set $X_1^k = C_1 \cup C_2 \cup C_3$, where C_1, C_2, C_3 are mutually disjoint sets such that $\theta(C_1) = C_2$ and $\theta(x) = x$ for all $x \in C_3$. Obviously $|C_1| = |C_2|$.

Denote $m = |C_1 \cup C_3|$, then $m \geq (\ell - 2)^k/2$. Consider the set of pairs of strings

$$\mathcal{F} = \big\{ \big(\prod_{w \in D} aw, \prod_{w \in (C_2 \cup C_3) \setminus \theta(D)} aw \big) \;\big|\; D \subseteq (C_1 \cup C_3) \big\}. \tag{1}$$

We show that $\mathcal{F}$ is a fooling set for $hpf(\theta, k)$.

(i) Consider an arbitrary pair $(x, y) \in \mathcal{F}$. Let $z \in X^k$ be a substring of xy. If z contains a, then $\theta(z)$ cannot be in xy as $\theta(a) = b$ and b is not in xy. If z does not contain a, then $z \in X_1^k$ and z is a subword of either x or y. Assume that z is a part of x. Then, by definition of C_1 and C_3, there is no occurrence of $\theta(z)$ in x which would not overlap z. Also, $\theta(z)$ is not a subword of y as $z \in D$ and hence $\theta(z) \notin (C_2 \cup C_3) \setminus \theta(D)$. If z is a subword of y, the situation is analogous. Therefore, $xy \in hpf(\theta, k)$.

(ii) Let (x, y), (x', y') be two distinct elements of $\mathcal{F}$, associated with the sets $D, D' \subseteq (C_1 \cup C_3)$ in the sense of (1). Let us assume without loss of generality that there is a $z \in D \setminus D'$. Then $\theta(z) \in (C_2 \cup C_3)$ and $\theta(z) \notin \theta(D')$, hence $\theta(z)$ is a subword of y'. Simultaneously z is a subword of x, therefore $xy' \notin hpf(\theta, k)$.

We can conclude that $|\mathcal{F}| = 2^m \geq 2^{(\ell-2)^k/2}$, and hence the statement follows by Lemma 6.1. □

Corollary 6.1. Let X be an alphabet such that $|X| = \ell$, $\ell \geq 2$. Let there be distinct letters $a, b \in X$ such that $a = \theta(b)$. Then the number of states of a minimal DFA over the alphabet X, accepting either $hp(\theta, k)$ or $hpf(\theta, k)$, $k \geq 1$, is between $2^{(\ell-2)^k/2}$ and $2^{3\ell^k}$.

Proof:
Observe that the numbers of states of minimal DFA's accepting $hp(\theta, k)$ and $hpf(\theta, k)$ are the same since these languages are mutual complements. Then the lower bound follows by Proposition 6.2. The upper bound follows by Proposition 6.1 and by the subset construction of a DFA equivalent to the NFA M_k mentioned there. □

Considering the important special case of the DNA alphabet when $\ell = 4$, Proposition 6.1 and Corollary 6.1 give the following result.

Corollary 6.2. Consider the DNA alphabet $\Delta = \{A, C, T, G\}$ and the Watson-Crick involution τ.

(i) The size of a minimal NFA accepting $hp(\tau, k)$ is at most $15 \cdot 4^k$. The number of its states is between 4^k and $3 \cdot 4^k$.

(ii) The number of states of either a minimal DFA or an NFA accepting $hpf(\tau, k)$ is between $2^{2^{k-1}}$ and $2^{3 \cdot 2^{2k}}$.

Note: after careful inspection of the automaton in the proof of Proposition 6.1, one can improve these bounds slightly for the case $\ell = 4$. Particularly, the actual size in the case (i) above is at most $\frac{25}{3} \cdot 4^k + \frac{14}{3}$ and the number of states does not exceed $\frac{5}{3} \cdot 4^k - \frac{2}{3}$.

The above results indicate that the size of a minimal NFA for $hp(\tau, k)$ grows exponentially with k. However, one should recall that k is the *minimal* length of bond allowing for a stable hairpin. Therefore k is rather low in practical applications (often $k = 2$ or $k = 3$) and the construction of the automaton remains computationally tractable.

7. Variants of hairpins

7.1. Scattered hairpins

It is a known fact that parts of two DNA molecules could form a stable bond even if they are not exact mutual Watson-Crick complements. They may contain some mismatches and even may have different lengths. Hybridizations of this type are addressed e.g. in [2] and [20]. Motivated by this observation, we consider now a generalization of hairpins.

Definition 7.1. Let θ be an involution of X^* and let k be a positive integer. A word $u = wy$, for $u, w, y \in X^*$, is θ-k-scattered-hairpin-free or simply shp(θ, k)-free if for all $t \in X^*$, $t \leq_e w$, $\theta(t) \leq_e y$ implies $|t| < k$.

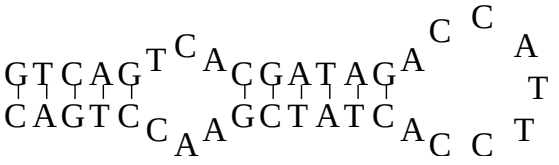

Figure 3. An example of a scattered hairpin – a word in $shp(\tau, 11)$.

Definition 7.2. We denote by $shpf(\theta, k)$ the set of all shp(θ, k)-free words in X^*, and by $shp(\theta, k)$ its complement $X^* - shpf(\theta, k)$.

Definition 7.3. A language L is called θ-k-scattered-hairpin-free or simply shp(θ,k)-free if $L \subseteq shpf(\theta,k)$.

Lemma 7.1. $shp(\theta, k) = S\left(\bigcup_{w \in X^k} w\theta(w) \right)$.

Based on the above immediate result, analogous statements as in Section 3 hold also for scattered hairpins. Proofs are straightforward and left to the reader.

Proposition 7.1. (i) The language $shp(\theta, k)$ is right $\leq_e$ -convex.

(ii) The languages $shp(\theta, k)$ and $shpf(\theta, k)$ are regular.

(iii) There exists a unique hypercode H such that $shp(\theta, k) = S(H)$.

As in Section 5, we can also define the *scattered-hairpin-freeness problem* and *maximal scattered-hairpin-freeness problem.* Then we easily obtain the following results whose proofs are analogous to those in Section 5.

Corollary 7.1. (i) The scattered-hairpin-freeness problem is decidable in linear time for regular languages and in cubic time for context-free languages.

(ii) The maximal scattered-hairpin-freeness problem is decidable in time $\mathcal{O}(|M_1| \cdot |M_2|)$ for regular languages and in time $\mathcal{O}((|M_1| \cdot |M_2|)^3)$ for deterministic context-free languages.

Also the size of the minimal automaton accepting the language $shp(\theta, k)$ is similar to the case of $hp(\theta, k)$ in Section 6.

Proposition 7.2. The number of states of a minimal NFA accepting the language $shp(\theta, k)$, $k \geq 1$, over an alphabet X with the cardinality ℓ, is between ℓ^k and $3\ell^k$, and its size is at most $7\ell^k + 3\ell^{k+1}$.

Proof:
Let $M_k = (S, X, s_1, F, P)$ be an NFA accepting $shp(\theta, k)$. The statement is trivial for the cases $\ell = 1$ or $k = 1$. Assume for the rest of the proof that $k \geq 2$ and $\ell \geq 2$.

(i) The reader can easily verify that the set $F = \{(w, \theta(w)) \mid w \in X^k\}$ is a fooling set for $hp(\theta, k)$. Therefore $|S| \geq \ell^k$.

(ii) Let

$$S = \{s_w, p_w \mid w \in X^{\leq k-1}\} \cup \{q_w \mid w \in X^k\}.$$

Let further $F = \{p_1\}$. The set of productions P is defined as follows:

$$\begin{array}{ll}
s_v a \to s_w \text{ if and only if } va = w, & \text{for each } v \in X^{\leq k-2},\ a \in X; \\
s_v a \to q_w \text{ if and only if } va = w, & \text{for each } v \in X^{k-1},\ a \in X; \\
q_w a \to p_v \text{ if and only if } \theta(av) = w, & \text{for each } v \in X^{k-1},\ a \in X; \\
p_w a \to p_v \text{ if and only if } av = w, & \text{for each } v \in X^{\leq k-2},\ a \in X. \\
ra \to r & \text{for all } r \in S,\ a \in X.
\end{array}$$

By the above construction similar to that of Proposition 6.1, $L(M_k) = shp(\theta, k)$, and $|S| \leq 3\ell^k$, $|P| \leq 4\ell^k + 3\ell^{k+1}$, therefore $|M_k| \leq 7\ell^k + 3\ell^{k+1}$.

□

7.2. Hairpin frames

In this section we point out the following two facts. First, long DNA and RNA molecules can form complicated secondary structures as that shown in Figure 4. Second, simple hairpins can be useful in various DNA computing techniques and nanotechnologies, as in [3, 25, 27] and others. Hence, sometimes it is desirable to design DNA strands forming simple hairpins but avoiding more complex structures. This motivates another extension of the results from Section 3.

Definition 7.4. The pair $(v, \theta(v))$ of a word u in the form $u = xvy\theta(v)z$, for $x, v, y, z \in X^*$, is called an *hp-pair* of u. The sequence of hp-pairs $(v_1, \theta(v_1))$, $(v_2, \theta(v_2))$, $\cdots$, $(v_j, \theta(v_j))$ of the word u in the form:

$$u = x_1v_1y_1\theta(v_1)z_1x_2v_2y_2\theta(v_2)z_2 \cdots x_jv_jy_j\theta(v_j)z_j$$

is called an *hp-frame of degree* j of u or simply an *hp(j)-frame* of u.

An hp-pair is an hp-frame of degree 1. The definition of hairpin frames characterizes secondary structures containing several complementary sequences such as that in Fig. 4.

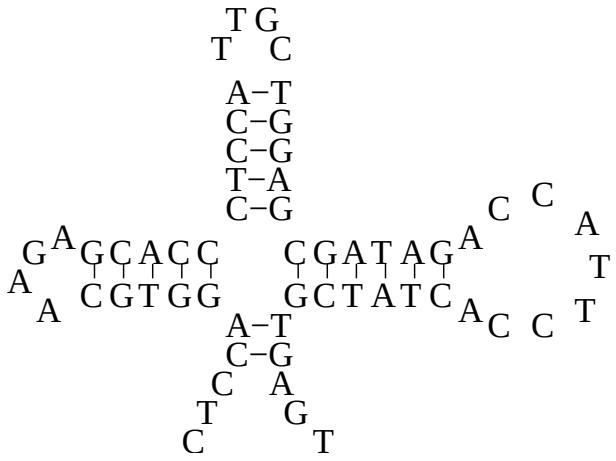

Figure 4. An example of a hairpin frame – a word in $hp(\tau, fr, 3)$.

A word u is said to be an *hp(fr,j)-word* if it contains at least one hp-frame of degree j. Observe that there may be more ways of finding hp-pairs in u, resulting in hp-frames of various degrees. Obviously, any hp(fr,j)-word is also hp(fr,i) for all $1 \leq i \leq j$.

Definition 7.5. For an involution θ we denote by $hp(\theta, fr, j)$ the set of all hp(fr,j)-words $u \in X^*$, and by $hpf(\theta, fr, j)$ its complement in X^*.

The results in Section 3, concerning the languages $hp(\theta, 1)$ and $hpf(\theta, 1)$, can easily be extended for the case of hairpin frames. Proofs are left to the reader.

Lemma 7.2. $hp(\theta, fr, j) = hp(\theta, 1)^j = \left(\bigcup_{a \in X} X^*aX^*\theta(a)X^* \right)^j$.

Proposition 7.3. (i) The language $hp(\theta, fr, j)$ is right $\leq_e$ -convex.

(ii) The languages $hp(\theta, fr, j)$ and $hpf(\theta, fr, j)$ are regular.

(iii) There exists a unique hypercode H such that $hp(\theta, fr, j) = S(H)$.

Corollary 7.2. (i) The *hp(fr,j)-freeness* problem is decidable in linear time for regular languages and in cubic time for context-free languages.

(ii) The *maximal hp(fr,j)-freeness* problem is decidable in time $\mathcal{O}(|M_1| \cdot |M_2|)$ for regular languages and in time $\mathcal{O}((|M_1| \cdot |M_2|)^3)$ for deterministic context-free languages.

Proposition 7.4. The size of a minimal NFA accepting the language $hp(\theta, fr, j)$, $j \geq 1$, over an alphabet X with the cardinality ℓ, is at most $4\ell j + 2j + 1$.

Proof:
The statement follows by the construction of an NFA $M = (S, X, s_1, F, P)$ accepting the language $hp(\theta, fr, j)$. Let

$$S = \{s_0, s_1, \ldots, s_j\} \cup \{p_i^k \mid 1 \leq i \leq j,\ 1 \leq k \leq \ell\}.$$

Let further $F = \{s_j\}$, and denote $X = \{a_1, \ldots, a_\ell\}$. The set of productions P is defined as follows:

$$\begin{array}{ll} s_{i-1}a_k \to p_i^k,\ p_i^k\theta(a_k) \to s_i & \text{for all } 1 \leq i \leq j,\ 1 \leq k \leq \ell; \\ sa \to s & \text{for all } s \in S,\ a \in X. \end{array}$$

The reader can verify that $L(M_k) = hp(\theta, fr, j)$, and that $|M| = 4\ell j + 2j + 1$. □

Unlike the cases of hairpins or scattered hairpins, the size of the minimal NFA accepting $hp(\theta, fr, j)$ is $\mathcal{O}(j\ell)$. However, if we considered also a minimal length k of the hairpin bonds, we would obtain the same exponential size of the automaton as in Section 6, but multiplied by j.

8. Construction of long hairpin-free words

In this section we discuss the problem of constructing long hp(θ, k)-free words for the cases where θ is the Watson-Crick involution and $\theta = \epsilon$. This question is relevant to various encoding problems of DNA computing. For example, in [27] the authors consider n-bit memory elements that are represented by DNA words of the form

$$u_1 v_1 w_1 \theta(v_1) \cdots u_n v_n w_n \theta(v_n) u_{n+1},$$

such that (i) all the u's and v's have length 20 and the w's have length 7, and (ii) the only bonds permitted in a word of this form are the bonds between v_i and $\theta(v_i)$ for all $i = 1, \ldots, n$. This encoding problem can be solved if we first construct a long hp(θ, k)-free word w of length $(20+20+7)n+20 = 47n+20$. Then w can be written in the form

$$u_1 v_1 w_1 \cdots u_n v_n w_n u_{n+1}$$

and is such that no bonds can occur between any two subwords of length k of w. Here k is the parameter that represents the smallest length of a block of nucleotides that can form a stable bond with a corresponding block of complementary nucleotides – see also the relevant discussion in [20].

For the case where θ is the Watson-Crick involution we consider the method of [20] for constructing $(\theta, H_{0,k})$-bond-free languages L. Such a language L has the property that, for any two subwords u and v of L of length k, one has that $u \neq \theta(v)$. Note that each word of L is a hp(θ, k)-free word. Moreover, if L is infinite then it contains arbitrarily long words, hence, also words of length $47n + 20$, for any n, as required in the encoding problem discussed in the beginning of this section. We also note that if L is (θ, k)-bond-free then it is (θ, k')-bond-free for any $k' \geq k$. The method of [20] is based on the *subword closure* language operation $\otimes$: Let S be a set of words of length k. Then $S^{\otimes}$ is the set of all words w of length at least k such that any subword of w of length k belongs to S. We note that given the set S one can construct a deterministic finite automaton accepting $S^{\otimes}$ in linear time [20]. The method is as follows. Let S be any set of words of length k such that $S \cap \theta(S) = \emptyset$. Then $S^{\otimes}$ is a $(\theta, H_{0,k})$-bond-free language. In our case, we wish to choose S such that $S^{\otimes}$ is infinite. For example, let S_2 be the set $\{AA, AC, CA, CC, AG, GA\}$. In [20] the authors show an automaton accepting $S_2^{\otimes}$. As $S_2^{\otimes}$ contains the set $(ACCAGAC)^+$ it follows that $S_2^{\otimes}$ is infinite as well.

For the case of $\theta = \epsilon$, we consider a totally different approach. Let $H(K)$ denote the minimum Hamming distance between any two different codewords of a code K. A language K is said to be a *solid code* if (i) no word of K is a subword of another word of K, and (ii) a proper and nonempty prefix of K cannot be a suffix of K. K is a *uniform code* if $K \subseteq X^n$ for an $n \in \mathbb{N}$. See [24] or Chapter 8 in [26] for background information on codes.

Proposition 8.1. Let $k \geq 2$ and let K be a uniform solid code of length k. If $H(K) > \lfloor k/2 \rfloor$, or $H(K) = \lfloor k/2 \rfloor$ and there are no different codewords with a common prefix of length $\lfloor k/2 \rfloor$, then the word $w_1 ... w_n$ is hp(θ, k)-free for all $n \leq \mathrm{card}(K)$ and for all pairwise different codewords $w_1, ..., w_n$.

Proof:
Assume there is $v \in X^k$ such that $w_1 ... w_n = xvyvz$ for some words x, y, z. If $|x|$ is a multiple of k then $v = w_j$ for some $j \geq 1$. As the w_i's are different, $|y|$ cannot be a multiple of k. Hence, $v = s_t p_{t+1}$, where $t > j$ and s_t is a proper and nonempty suffix of w_t and p_{t+1} is a proper and nonempty prefix of w_{t+1}; a contradiction. Now suppose $|x|$ is not a multiple of k. Then, $v = s_j p_{j+1}$ for some nonempty suffix s_j and prefix p_{j+1}. Again, the second occurrence of v cannot be in K. Hence, $v = s_t p_{t+1}$ for some $t \geq j$. Hence, $s_j p_{j+1} = s_t p_{t+1}$. If $|s_j| \neq |s_t|$, say $|s_j| > |s_t|$, then a prefix of p_{t+1} is also a suffix of s_j; which is impossible. Hence, $s_j = s_t$ and $p_{j+1} = p_{t+1}$.

Note that $H(K) \geq \lfloor k/2 \rfloor$ and, therefore, $\lfloor k/2 \rfloor \leq H(\{p_{j+1}s_{j+1}, p_{t+1}s_{t+1}\}) = H(\{s_{j+1}, s_{t+1}\})$ $\leq |s_{j+1}| = k - |p_{j+1}|$. Hence, $|p_{j+1}| \leq \lceil k/2 \rceil$. Similarly, $|s_j| \leq \lceil k/2 \rceil$. Also, as $k = |s_j| + |p_{j+1}|$, one has that $|s_j|, |p_{j+1}| \in \{\lfloor k/2 \rfloor, \lceil k/2 \rceil\}$. If $H(K) = \lfloor k/2 \rfloor$ then $p_{j+1} = p_{t+1}$ implies that w_{j+1} and w_{t+1} have a common prefix of length $\lfloor k/2 \rfloor$; a contradiction. If $H(K) > \lfloor k/2 \rfloor$ then both p_{j+1} and s_j are shorter than $\lceil k/2 \rceil$ which contradicts with $k = |s_j| + |p_{j+1}|$. □

Suppose the alphabet size $|X|$ is $l > 2$. We can choose any symbol $a \in X$ and consider the alphabet $X_1 = X - \{a\}$. Then for any uniform code $F \subseteq X_1^{k-1}$ it follows that the code $F\{a\}$ is a uniform solid code of length $k : F\{a\} \subseteq X^k$. We are interested in cases where the code F is a linear code of type $[k-1, m, d]$. That is, F is of length $k-1$, cardinality $(l-1)^m$, and $H(F) = d$, and there is an $[m \times (k-1-m)]$ matrix G over X_1 such that $F = \{w * [I_m | G] \mid w \in X_1^m\}$, where I_m is the $(m \times m)$ identity matrix and $*$ is the multiplication operation between an m-dimensional row vector and $(m \times m)$ matrix. Thus, $u \in F$ *iff* $u = wx$ for some $w \in X_1^m$ and $x \in X_1^{k-1-m}$ and $x = w * G$.

Proposition 8.2. Let F be a linear code over X_1 of type $[k-1, m, \lfloor k/2 \rfloor]$. If $m \leq \lfloor k/2 \rfloor$ or k is even then the word $w_1..w_n$ is hp(θ, k)-free for all $n \leq \mathrm{card}(F)$ and for all pairwise different codewords $w_1, ..., w_n$ in $F\{a\}$.

Proof:
It is sufficient to show that $H(F\{a\}) = \lfloor k/2 \rfloor$ and there are no different words in $F\{a\}$ with a common prefix of length $\lfloor k/2 \rfloor$. Obviously $H(F\{a\}) = H(F) = \lfloor k/2 \rfloor$. As F is generated by a matrix $[I_m|G]$, where G is a matrix in $X_1^{m\times(k-1-m)}$, it follows that there can be no different words in F with a common prefix of length m. If $m \leq \lfloor k/2 \rfloor$ then there can be no different words in $F\{a\}$ with a common prefix of length $\lfloor k/2 \rfloor$. If k is even, consider the well known bound on $|F|$: $|F| \leq |X_1|^{k-1-\lfloor k/2 \rfloor+1}$. Hence, $|X_1|^m \leq |X_1|^{\lfloor k/2 \rfloor}$ which gives $m \leq \lfloor k/2 \rfloor$. Hence, again, we are done. □

By the above one can construct an hp(θ, k)-free word of length nk, for some $n \leq \mathrm{card}(F)$, for every choice of n different words in $F\{a\}$. It is interesting that, for $k = 13$ and $|X| = 4$, the famous Golay code G_{12} of type $[12, 6, 6]$ satisfies the premises of the above Proposition.

Acknowledgements

Research was partially supported by the Canada Research Chair Grant to L.K., NSERC Discovery Grants R2824A01 to L.K. and R220259 to S.K., and by the Czech Science Foundation, grant No. 201/06/0567 to P.S.

References

[1] Amos, M.: *Theoretical and Experimental DNA Computations*, Springer-Verlag, Berlin, 2005.

[2] Andronescu, M., Dees, D., Slaybaugh, L., Zhao, Y., Condon, A., Cohen, B., Skiena, S.: Algorithms for testing that sets of DNA words concatenate without secondary structure. *Proc. 8th Workshop on DNA-Based Computers* (M. Hagiya, A. Ohuchi, Eds.), LNCS 2568, Springer-Verlag, Berlin, 2002, 182–195.

[3] Benenson, Y., Gil, B., Ben-Dor, U., Adar, R., Shapiro, E.: An autonomous molecular computer for logical control of gene expression. *Nature*, **429**, 2004, 423–429.

[4] Birget, J. C.: Intersection and union of regular languages and state complexity, *Information Processing Letters*, **43**, 1992, 185–190.

[5] Brodal, G. S., Lyngsø, R. B., Pedersen, C. N. S., Stoye, J.: Finding maximal pairs with bounded gap. *Proc. 10th Annual Symposium on Combinatorial Pattern Matching (CPM)* (M. Crochemore and M. Paterson, Eds.), LNCS 1645, Springer-Verlag, Berlin, 1999, 134–149.

[6] Calladine, C. R., Drew, H. R.: *Understanding DNA: The Molecule and How It Works*, 2nd edition, Academic Press, San Diego, 1999.

[7] Choffrut, C., Karhumäki, J.: Combinatorics of words, in [26], 329–438.

[8] Daley, M., Kari, L.: Some properties of ciliate bio-operations, *Proc. of DLT 2002* (M. Ito, M. Toyama, Eds.), LNCS 2450, Springer-Verlag, Berlin, 2003, 116–127.

[9] Dekking, F. M.: On repetitions of blocks in binary sequences, *J. Combin. Theory Ser. A*, **20**, 1976, 292–299.

[10] Ehrenfeucht, A., Harju, T., Petre, I., Prescott, D., Rozenberg, G.: *Computation in Living Cells: Gene Assembly in Ciliates*, Springer-Verlag, Berlin, 2004.

[11] Haines, L. H.: On free monoids partially ordered by embedding, *J. of Combinatorial Theory*, **6**, 1969, 94–98.

[12] Harju, T., Karhumäki, J.: Morphisms, in [26], 439–510.

[13] Hopcroft, J., Ullman, J., Motwani, R.: Introduction to Automata Theory, Languages, and Computation, 2nd ed., Addison-Wesley, 2001.

[14] Jonoska, N., Kephart, D., Mahalingam, K.: Generating DNA code words, *Congressus Numerantium*, **156**, 2002, 99–110.

[15] Jonoska, N., Mahalingam, K.: Languages of DNA based code words, *Proc. DNA Computing, 9th International Workshop on DNA Based Computers* (J. Chen and J. H. Reif, Eds.), LNCS 2943, Springer-Verlag, Berlin, 2004, 61–73.

[16] Kobayashi, S.: Testing structure-freeness of regular sets of biomolecular sequences, *Proc. DNA 10* (C. Feretti, G. Mauri, C. Zandron, Eds.), LNCS 3384, Springer-Verlag, Berlin, 2005, 192–201.

[17] Kijima, A., Kobayashi, S.: Efficient algorithms for testing structure-freeness of finite sets of biomolecular sequences, *Proc. DNA 11* (A. Carbone et al., Eds.), London, The University of Western Ontario, 2005, 278–288.

[18] Lothaire, M.: *Algebraic Combinatorics on Words*, Cambridge University Press, 2002.

[19] de Luca, A.: On the combinatorics of finite words, *Theoretical Computer Science*, **218**, 1999, 13–39.

[20] Kari, L., Konstantinidis, S., Sosík, P.: Bond-free languages: formalizations, maximality and construction methods, *Proc. DNA 10* (C. Feretti, G. Mauri, C. Zandron, Eds.), LNCS 3384, Springer-Verlag, Berlin, 2005, 169–181.

[21] Mauri, G., Ferretti, C.: Word Design for Molecular Computing: A Survey, *Proc. DNA Computing, 9th International Workshop on DNA Based Computers* (J. Chen and J. H. Reif, Eds.), LNCS 2943, Springer-Verlag, Berlin, 2004, 37–46.

[22] Păun, G., Rozenberg, G., Salomaa, A.: *DNA Computing: New Computing Paradigms*, Springer Verlag, Berlin, 1998.

[23] Păun, G., Rozenberg, G., Yokomori, T.: Hairpin languages, *Intern. J. Found. Computer Sci.*, **12** (6), 2001, 849–857.

[24] Roman, S.: *Coding and Information Theory*, Springer-Verlag, New York, 1992.

[25] Rose, J. A., Deaton, R. J., Hagiya, M., Suyama, A.: PNA-mediated Whiplash PCR, *Proc. DNA Computing, 7th International Workshop on DNA Based Computers* (N. Jonoska and N. C. Seeman, Eds.), LNCS 2340, Springer-Verlag, Berlin, 2002, 104–116.

[26] Rozenberg, G., Salomaa, A. (Eds.): *Handbook of Formal Languages*, vol. 1, Springer Verlag, Berlin, 1997.

[27] Takahashi, N., Kameda, A., Yamamoto, M., Ohuchi, A.: Aqueous computing with DNA hairpin-based RAM. *Proc. DNA 10, Tenth International Meeting on DNA Computing* (G. Mauri, C. Ferretti, Eds.), University of Milano-Bicocca, 2004, 50–59.

[28] Thierrin, G.: Convex languages, *Proc. IRIA Symp. North Holland*, 1972, 481–492.

Appendix

Proof of Lemma 4.1:

1. If $i \geq 8$ then w would contain a subword of the form a^4ua^4; a contradiction. If the run is internal and $i \geq 6$ then w would contain a subword of the form ba^3ua^3b; a contradiction again.

2. First note that if w contained the three runs then it would also contain the subwords a^3b and ba^3, which is impossible. Similarly, if w contains the two runs a^j and a^i, they can appear in w only as in $\cdots a^jba^i \cdots$. Also, not both runs can be internal as in $\cdots ba^j \cdots a^ib \cdots$, otherwise w would contain ba^3 and a^3b. The fact that at least one of i and j is 3 is evident.

3. If w contained three internal runs a^2 then the subword ba^2b would occur twice. For the same reason, if w has two runs a^2 they must occur as specified in the statement.

4. Follows by symmetry.

□

Proof of Lemma 4.2:
Suppose there are four indices i with $x_i = y_i = 1$. Then w can be written in the form

$$au_0bau_1bau_2bau_3bau_4b,$$

such that $u_0 \in \{1\} \cup X^*a$ and $u_4 \in \{1\} \cup bX^*$ and $u_i \in \{1\} \cup bX^*a$ for $i = 1, 2, 3$. Note that $abab$ occurs both in au_0bau_1b and au_3bau_4b. This implies that both u_1 and u_3 are nonempty (else, also $baba$ would occur in w). Hence, u_1 and u_3 can be written in the form $bu_1'a$ and $bu_3'a$, respectively, and this implies that u_2 must be nonempty (else the subword $baba$ would follow $abab$ in w). So w is of the form

$$au_0ba(bu_1'a)ba(bu_2'a)ba(bu_3'a)bau_4b.$$

Again, both u_2' and u_3' must be nonempty and, moreover, they do not start with a. Hence, each of u_2' and u_3' starts with b and, therefore, w contains two internal runs b^r and b^s with $r, s \geq 2$. By Lemma 4.1 and the positions of the two runs in w, there can be no other internal run b^t, with $t \geq 2$, to the left of $bu_2'a$. This implies that u_1' cannot start with b and that $baba$ occurs as a subword in $ba(bu_1'a)$, which contradicts the fact that $abab$ occurs in u_2bau_3. □

Proof of Lemma 4.3:
We shall prove the statement for $c = a$. By Lemma 4.1, w starts with a^jba^i, or ends with a^iba^j, such that $i, j \geq 3$. We shall prove the statement for the former case, as the other case would follow by symmetry. We distinguish two cases depending on whether w contains an internal run a^2 – there can be no two internal runs a^2 in w, else a^2ba and aba^2 would occur in w.

Case 1: No internal run a^2. Then w is of the form

$$a^jba^i(b^{x_1}a) \cdots (b^{x_{k-1}}a)b^{x_k}u,$$

where $k \geq 1$ and u is in $\{1, a, a^2\}$. If $k = 1$ the statement is evident. We assume that $k \geq 2$. We note that there can be no three different runs b^y with $y \geq 2$ (otherwise w would contain b^2ab and bab^2 as

subwords). By Lemma 4.1, the length of $a^j ba^i$ is at most 11. By Lemma 4.2, w contains up to 3 pairs of runs (ba). Then $|w| \leq 11 + 3 \times 2 + |b^2 a| + |b^5 a| + |b^{x_k} u| \leq 29$, or $|w| \leq 11 + 3 \times 2 + |b^3 a| + |b^7| \leq 28$. Hence, $|w| \leq 29$.

Case 2: There is an internal run a^2. Then w is of the form

$$a^j ba^i (b^{x_1} a) \cdots (b^{x_{r-1}} a)(b^{x_r} a^2)(b^{x_{r+1}} a) \cdots (b^{x_{k-1}} a) b^{x_k} u,$$

such that $1 \leq r < k$ and $x_r \geq 2$ (else $a^2 ba$ and aba^2 would occur in w). Hence, $b^2 a^2$ occurs as a subword of w. We note that $r \geq 2$ implies $x_1 = 1$ (else $a^2 b^2$ would occur in w) and bab^2 occurs to the left of the internal run a^2. Also, $k > r + 1$ implies $x_{r+1} \geq 2$ (else $a^2 ba$ would occur to the right of aba^2) and $b^2 ab$ occurs to the right of the internal run a^2. Hence, it follows that $r = 1$ or $k = r + 1$.

First consider the subcase of $r = 1$. If $k = r + 1$ then $|w| \leq 22$. So we continue assuming $k > r + 1$. Then $x_r \leq 3$ (otherwise, $a^2 b^2$ would occur in w as a separate subword from $b^2 a^2$). Moreover, Lemma 4.1 implies that one of x_r and x_{r+1} is 2 and the other is at least 3. If $x_r = 3$ and $x_{r+1} = 2$ then there can be no other run b^y with $y \geq 3$ and $|w| \leq 11 + 5 + 2|b^2 a| + 3|ba| + |b^{x_k} u| \leq 31$. If $x_r = 2$ and $x_{r+1} \geq 3$ then there can be no other internal run b^2 in w. If there is another run b^y with $y \geq 3$ then $k = r + 2$ and $u = 1$ and $|w| \leq 11 + 4 + |b^{x_{k-1}} a b^{x_k}| \leq 26$. Else, $|w| \leq 11 + |b^{x_r} a^2| + |b^{x_{r+1}} a| + 3|ba| + |b^{x_k} u| \leq 30$.

Now consider the subcase of $k = r + 1$ and $r \geq 2$. Then

$$w = a^j ba^i (b^{x_1} a) \cdots (b^{x_{r-1}} a)(b^{x_r} a^2) b^{x_k} u$$

Recall that $x_1 = 1$. First assume that u is not empty. Then $x_k \geq 2$ (else w would contain aba^2 and $a^2 ba$) and one of x_r and x_k is 2 and the other is at least 3. If $x_k = 2$ there is no other run b^y in w with $y \geq 2$, and $|w| \leq 11 + 3|ba| + (5 + 2) + (2 + |u|) \leq 28$. If $x_r = 2$ there is at most another internal run b^2. Also, here $u = a^2$ implies $x_k \leq 3$ (else, w would contain $a^2 b^2$ and $b^2 a^2$ as subwords). Hence, $|w| \leq 30$. Now assume that u is empty. If there is no index $t < r$ with $x_t \geq 2$ then $|w| \leq 11 + 3|ba| + (x_r + 2) + x_k \leq 28$. If there is an index $t < r$ such that $x_t \geq 2$ then the largest such t would be equal to $r - 1$ (else, $abab$ would occur to the left of $b^{x_t} a$ and $baba$ would occur to the right of $b^{x_t} a$). Also there can be no other index $s < t$ with $x_s \geq 2$, otherwise bab^2 would occur to the left of $b^{x_{r-1}}$ and $b^2 ab$ would occur in $b^{x_{r-1}} a b^{x_r}$). Hence, w is of the form

$$a^j ba^i (ba) \cdots (ba)(b^{x_{r-1}} a)(b^{x_r} a^2) b^{x_k}$$

with at most 3 (ba)'s in w. If one of x_{r-1} and x_r is at least 3 then the other would be 2 and $x_k \leq 2$. This implies $|w| \leq 29$. If $x_{r-1} = x_r = 2$ then $x_k \leq 7$ and $|w| \leq 31$. □

Proof of Lemma 4.4:
By Lemma 4.1, there can be no other internal run b^t with $t \geq 2$. We distinguish two cases depending on whether $b^2 ab^2$ occurs to the left or to the right of b^y. In both cases, we make use of the quantity $h = y_0 + y_{n+1} + t$, where t is the length of the only internal run a^t, with $t \geq 3$, if it exists, or $t = 0$ otherwise. Note that if y_0 or y_{n+1} is greater than 2 then $t = 0$ – recall there can be no two runs a^i and a^j with $i, j \geq 3$. Also, if the run a^t exists then $t \leq 5$ and $h \leq 2 + 2 + 5 = 9$. Similarly, if $t = 0$ then $h \leq (7 + 2) + 0 = 9$.

Case 1: the word w is of the form

$$(a^{y_0})(ba^{y_1} \cdots ba^{y_{l-1}})(b^2 a)(b^2 a^{y_{l+1}})(ba^{y_{l+2}} \cdots ba^{y_{k-1}})(b^{x_k} a^{y_k})(ba^{y_{k+1}} \cdots ba^{y_n})(b^{x_{n+1}} a^{y_{n+1}}),$$

where $x_k \geq 3$ and $k > l+1$, and $k \leq n$, or $k = n+1$ and $y_{n+1} > 0$. Also, $y_{k-1} \geq 2$, else bab^2 would occur to the right of b^2ab^2. First we consider the subcase of $k \leq n$. Here $y_k \geq 2$ (else, b^2ab would occur to the right of b^2ab^2) and $x_k = 3$ (else, $a^{y_{k-1}}b^{x_k}a^{y_k}$ contains $a^2b^2 \cdots b^2a^2$). Moreover one of y_{k-1} and y_k is 2 and the other is at least 3. If $y_{n+1} = 0$ then $h \leq 7$ and depending on whether the run a^t, with $t \geq 3$, is equal to a^{y_k} we have that $|w| \leq (h + x_k) + 3|ba| + 2|b^2a| + 2|ba^2| + x_{n+1}$ or $|w| \leq (h+1) + 3|ba| + 2|b^2a| + (x_k + 2) + |ba^2| + x_{n+1}$. Hence, $|w| \leq 30$. If $y_{n+1} > 0$ then $x_{n+1} = 1$ (else there would be another internal run b^s with $s \geq 2$). Again, by considering whether the run a^t, with $t \geq 3$, is equal to a^{y_k} one verifies that $|w| \leq 31$.

Now we consider the subcase of $k = n+1$ and $y_{n+1} > 0$. Then w is of the form

$$(a^{y_0})(ba^{y_1} \cdots ba^{y_{l-1}})(b^2a)(b^2a^{y_{l+1}})(ba^{y_{l+2}} \cdots ba^{y_{k-1}})(b^{x_k}a^{y_k}).$$

Here we note that $k - 1 > l + 1$ implies that $y_{l+1} = 1$ (else b^2a^2 would occur to the left of a^2b^2), and similarly $x_k \geq 4$ implies that $y_k = 1$. If there is up to one internal run a^2 then one verifies that $|w| \leq 29$. If there are two internal runs a^2 in w then their position depends on the value of y_{k-1}. If $y_{k-1} = 2$ then also $y_{k-2} = 2$. Moreover, $y_{l+1} = 1$ because $k - 1 > l + 1$, and there can be no internal run a^t with $t \geq 3$ (else aba^2 would occur to the left of a^2ba^2). Then $|w| \leq h + 3|ba| + 2|b^2a| + 2|ba^2| + x_k \leq 31$. If $y_{k-1} \geq 3$ then $k - 1 = l + 1$ (else aba^2 would occur to the right of a^2ba). Here one verifies that $|w| \leq 30$ if $x_k \geq 4$, and $|w| \leq 29$ if $x_k = 3$.

Case 2: the word w is of the form

$$(a^{y_0})(ba^{y_1} \cdots ba^{y_{k-1}})(b^{x_k}a^{y_k})(ba^{y_{k+1}} \cdots ba^{y_{l-1}})(b^2a)(b^2a^{y_{l+1}})(ba^{y_{l+2}} \cdots ba^{y_n})(b^{x_{n+1}}a^{y_{n+1}}),$$

where $x_k \geq 3$ and $k < l$, and $l + 1 \leq n$, or $l + 1 = n + 1$ and $y_{n+1} > 0$. Also, $y_k \geq 2$, else b^2ab would occur to the left of bab^2. First we consider the subcase of $k \geq 2$. Then $y_{k-1} \geq 2$ and $x_k = 3$ (else a^2b^2 and b^2a^2 would occur in w). Moreover, $|w| \leq \max\{2|ba^2| + h + x_k, |ba^2| + (x_k + |a^2|) + h + 1\} + 3|ba| + 2|b^2a| + x_{n+1}$ which implies $|w| \leq 21 + h + x_{n+1}$. If $y_{n+1} = 0$ then $h \leq 7$ and $|w| \leq 30$. If $y_{n+1} > 0$ and $n + 1 > l + 1$ then $x_{n+1} = 1$ and $|w| \leq 31$. If $y_{n+1} > 0$ and $n + 1 = l + 1$ then $y_{n+1} = 1$, $x_{n+1} = 2$ and $|w| \leq 31$.

Now we consider the subcase of $k = 1$. Then w is of the form

$$(a^{y_0})(b^{x_k}a^{y_k})(ba^{y_{k+1}} \cdots ba^{y_{l-1}})(b^2a)(b^2a^{y_{l+1}})(ba^{y_{l+2}} \cdots ba^{y_n})(b^{x_{n+1}}a^{y_{n+1}}).$$

Here $x_k \geq 4$ implies $y_0 = 1$. If $y_{n+1} = 0$ then $h + x_k \leq 11$ and $|w| \leq 31$. If $y_{n+1} \geq 1$ and $l + 1 = n + 1$ then $x_{n+1} = 2$ and $|w| \leq 30$. If $y_{n+1} \geq 1$ and $l + 1 < n + 1$ then $x_{n+1} = 1$. One verifies that again w is of length at most 31 by considering the following three possibilities: (i) $l > k + 1$, which implies $y_{n+1} = 1$; (ii) $l = k + 1$ and $y_k = 2$; and (iii) $l = k + 1$ and $y_k \geq 3$, which implies that $y_{l+1} = 1$ and $y_k = 3$. □

Proof of Lemma 4.5:
By Lemma 4.1, $y_0 + x_{n+1} + y_{n+1} \leq 7 + \max\{7 + 0, 5 + 2\} = 14$ and w contains at most two internal runs a^2 and at most two internal runs b^2. If it is not the case that there are two internal runs a^2 and two internal runs b^2, then $|w| \leq 14 + 3|ba| + \max\{2|b^2a| + |ba^2|, |b^2a| + 2|ba^2|\} = 29$. If there are two internal runs a^2 and two internal runs b^2 then a^2ba^2 would occur to the right of b^2ab^2 (else aba^2 would occur to the right of $a^{y_0}ba$). Hence

$$w = a^{y_0}u_1(b^2ab^2)u(a^2ba^2)u_2b^{x_{n+1}}a^{y_{n+1}},$$

where $u_1, u_2 \in (ba)^*$ and $u \in (ab)^*$. Hence, $|w| \leq 24 + |u_1| + |u| + |u_2|$. Now note that $u \neq 1$ implies $u_1 = u_2 = 1$ (else $baba$ occurs in u and $abab$ in u_1 or u_2), and $u_1 \neq 1$ implies that $u = 1$ and also implies that, either $u_2 = 1$, or $u_1 = u_2 = ba$. In any case it follows that $|w| \leq 30$. □