Proceedings of Grammar Systems Week 2004, edited by E. Csuhaj-Varjú and Gy. Vaszil,
MTA SZTAKI, Budapest, pages 212 - 224, 2004.

DNA Involutions and Hairpin Structures

Lila Kari

Department of Computer Science, The University of Western Ontario,
London, ON, Canada, N6A 5B7,
lila@csd.uwo.ca

Stavros Konstantinidis

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

Elena Losseva

Department of Computer Science, The University of Western Ontario,
London, ON, Canada, N6A 5B7,
elena@csd.uwo.ca

Petr Sosík

Department of Computer Science, The University of Western Ontario,
London, ON, Canada, N6A 5B7,
sosik@csd.uwo.ca
and
Institute of Computer Science, Silesian University, Opava, Czech Republic

Gabriel Thierrin

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

Abstract

We formalize the notion of a DNA hairpin secondary structure, examining its mathematical properties. Two related secondary structures are also investigated, taking into account imperfect bonds (bulges, mismatches) and multiple hairpins. We characterize maximal sets of hairpin-forming DNA sequences, as well as hairpin-free ones. We study their algebraic properties and their computational complexity. Related polynomial-time algorithms deciding hairpin-freedom of regular sets are presented. Finally, effective methods for design of long hairpin-free DNA words are given.

1 Introduction

A single strand of deoxyribonucleic acid (DNA) consists of a sugar-phosphate backbone and a sequence of nucleotides attached to it. There are four types of nucleotides denoted by A, C, T, and G. Two single strands can bind to each other if they have opposite polarity (strand's orientation in space) and are pairwise Watson-Crick complementary: A is complementary to T, and C to G. The binding of two strands is also called annealing. The ability of DNA strands to anneal to each other allows for creation of different secondary structures. A DNA hairpin is a particular type of secondary structure investigated in this paper. An example of hairpin structure is shown in Figure 1.

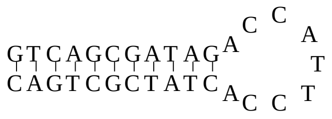

Figure 1: Hairpin loop.

The reader is referred to [9, 17] for an overview of the DNA computing paradigm. Study of DNA secondary structures such as hairpin loops is motivated by finding reliable encodings for DNA computing techniques. These techniques usually rely on a certain set of DNA bonds and secondary structures, while other types of bonds and structures are undesirable. Various approaches to the design of DNA encodings without undesirable bonds and secondary structures are summarized in [15]. For more details we refer the reader e.g. to [3, 4, 13, 14]. Here we apply the formal language approach which has been used in [1, 7, 10, 11, 12] and others.

Hairpin-like secondary structures are of special importance for DNA computing. For instance, they play an important role in insertion/deletion operations with DNA. Hairpins are the main tool used in the Whiplash PCR computing techniques [19]. In [21] hairpins serve as a binary information medium for DNA RAM. Last, but not least, hairpins are basic components of "smart drugs" [2].

The paper is organized as follows. Section 2 introduces basic definitions, Section 3 presents results on hairpins, and the following two sections define important variants on the hairpin definition. The first one takes into account imperfect DNA bonds (mismatches, bulges), the second one is related to hairpin-based nanomachines. We characterize maximal hairpin-free languages via hypercodes and study their algebraic properties. The hairpin-freedom problem and the problem of maximal hairpin-free sets are both shown to be decidable in linear (cubic) time for regular (context-free, respectively) DNA languages. The last section provides methods of constructing long hairpin-free words.

2 Preliminary Definitions

In this paper we will use X to denote a finite alphabet and X^* its corresponding free monoid. The cardinality of the alphabet X is denoted by $|X|$. The empty word is denoted by 1, and $X^+ = X^* - \{1\}$. A *language* is an arbitrary subset of X^*. A *uniform (block) code* is a language all the words of which are of the same length k, for some $k \geq 0$.

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 $\phi(1) = 1$. See [5] 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 $\theta^{-1} = \theta$.

In our examples we shall refer to the DNA alphabet $\Delta = \{A, C, T, G\}$, on which 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 $\mu\gamma$ (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 $\mu\gamma(u) = v$, then the two DNA strands represented by u, v anneal as Watson-Crick complementary sequences.

A nondeterministic finite automaton (*NFA*) is a quintuple $A = (S, X, s_0, F, P)$, 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 A is denoted by $L(A)$. The *size* $|A|$ of the automaton A is the number $|S| + |P|$.

Analogously we define a *pushdown automaton (PDA)* and a *deterministic pushdown automaton (DPDA)*. We refer to [6, 20] for detailed definitions and further information on formal language theory.

3 Hairpins

Definition 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 involution $\mu\gamma$, then a hairpin structure with the length of bond greater than or equal to k is a word that is not hp(θ,k)-free.

Definition 2 Denote by *hpf*(θ, k) the set of all hp(θ,k)-free words in X^*. The complement of *hpf*(θ, k) is *hp*$(\theta, k) = X^* -$ *hpf*(θ, k).

Note that *hp*(θ, 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. In particular, if $w \in$ *hp*(θ, k), then w can be written as $w = xvy\theta(v)z$ with $|v| = |\theta(v)| = k$ for some $x, v, y, z \in X^*$. It is also the case that $hp(\theta, k+1) \subseteq hp(\theta, k)$ for all $k > 0$.

Definition 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 *iff* $X^*vX^*\theta(v)X^* \cap L = \emptyset$ for all $|v| \geq k$.

Examples.

(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 a 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 now investigate properties of the languages $hp(\theta, k)$ and $hpf(\theta, k)$. Recall that $hp(\mu, k)$ is the set of all words containing two non-overlapping mirror parts of length at least k.

Lemma 1 *Consider a binary alphabet X. Then $hpf(\mu, k)$ is finite if and only if $k \leq 4$.*

Proof. Denote $X = \{a, b\}$ and consider a word w in the set $hpf(\mu, 4)$. Suppose that w is arbitrarily long. This word is of the form

$$w = a^{r_1}b^{s_1} \cdots a^{r_n}b^{s_n},$$

with $r_1 \geq 0$ and $r_i \geq 1$, for $i > 1$, and $s_n \geq 0$ and $s_j \geq 1$, for $j < n$. As w is in $hpf(\mu, 4)$, one has that $r_i < 8$ and $s_i < 8$, for all i, and n is arbitrarily large.

Consider now indices i_1, i_2, j_1, j_2 such that $1 < i_1, i_2, j_1, j_2 < n$. There can be at most one such index i_1 with $r_{i_1} \geq 3$ – otherwise, both ba^3 and a^3b would occur in w. By symmetry, there is at most one j_1 with $s_{j_1} \geq 3$. Moreover, if there is an index i_2 such that $r_{i_2} = 2$, then either there is no other index with this property, or it is the case that $s_{i_2} = 1$ and $r_{i_2+1} = 2$ and no other index i exists with $r_i = 2$. By symmetry again, there can be an index j_2 such that $s_{j_2} = 2$, and possibly $s_{j_2+1} = 2$, and no other index j exists with $s_j = 2$. Hence, we have that $r_k = 1$ for all k except i_1, i_2 and possibly 1, n and $i_2 + 1$, and $s_l = 1$ for all indices l except j_1, j_2

and possibly 1, n and j_2+1. As w is long enough, it would contain the substring $abababab a$, which contradicts the assumption that $w \in hpf(\mu, 4)$. Hence, w cannot be arbitrarily long and, therefore, the set $hpf(\mu, 4)$ is finite.

Now consider the language $L_5 = (aabbab)^+$. The set of its subwords of length 5 is $Sub_5(L_5) = \{aabba, abbab, bbaba, babaa, abaab, baabb\}$. 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 $1 \le m \le k$, finiteness of $hpf(\mu, k)$ implies also finiteness of $hpf(\mu, m)$. Hence the facts that $hpf(\mu, 4)$ is finite and $hpf(\mu, 5)$ is infinite imply the statement of the lemma. □

Proposition 1 *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 $k \le 4$.

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 $k \ge 1$, hence $hpf(\theta, k)$ is infinite.

Assume now that $\theta = \epsilon$ and let w be any word of length $\ge k|X|^k + k$. Since there are $|X|^k$ distinct words of length k, there are at least two 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* $k \le 4$ by Lemma 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). □

3.1 Properties of hp$(\theta, 1)$-free languages

Recall the definition of an embedding order: $u \le_e v$ if and only if

$$u = u_1 u_2 \cdots u_n, \quad v = v_1 u_1 v_2 u_2 \cdots v_n u_n v_{n+1}$$

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

A language L is called *right $\le_e$-convex* [22] if $u \le_e w$, $u \in L$ implies $w \in L$. The following result is well known: *All languages (over a finite alphabet) that are right$\le_e$-convex are regular.* We show that $hp(\theta, 1)$ is right $\le_e$-convex (and hence regular).

Lemma 2 *If $u = u_1 u_2 \in hp(\theta, 1)$ and $w \in X^*$ then $u_1 w u_2 \in hp(\theta, 1)$.*

Proposition 2 *The language $hp(\theta, 1)$ is right $\leq_e$ -convex.*

Proof. Immediate. □

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

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

Hence $S(L)$ is the set of all the words $w \in X^*$ that can be expressed 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 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 *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 the result 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 $H = \{aa, bb\}$.

3.2 Properties of hp(θ, k)-free languages

The previous results, Lemma 2 and Proposition 2, true for the case $k = 1$, cannot in general be extended to the case $k > 1$ as shown in the following example.

Example. Let $X = \{a, b\}$ with morphic $\theta(a) = b$, $\theta(b) = a$. If $u = a^2b^2$, then $u = a^2\theta(a^2)$ and hence $u \in hp(\theta, 2)$. The word $w = abab^2 = a.b.abb$ is obtained from u by the insertion of the word b and it is immediate that $w \notin hp(\theta, 2)$.

Furthermore, the language $hp(\theta, 2)$ is not $\leq_e$-convex, because $u \leq_e w$, $u \in hp(\theta, 2)$ and $w \notin hp(\theta, 2)$. However, the following result can be shown about $hpf(\theta, k)$-free languages, which indicates a possible construction method.

Proposition 4 *If $u = u_1u_2 \in hp(\theta, 2k)$ and $w \in X^+$, then $u_1wu_2 \in hp(\theta, k)$ for any $k \geq 1$.*

Proof. Let $u = u_1u_2 \in hp(\theta, 2k)$. Then u can be written as $u = xvy\theta(v)z$ with $|v| = |\theta(v)| = 2k$. The word u has v and $\theta(v)$ as substrings, which can be changed by the insertion of w into u only if insertion happens either in the middle of v or in the middle of $\theta(v)$. In the first case; $v_1wv_2, \theta(v_1v_2)$ are substrings of u_1wu_2 with $max(|v_1|, |v_2|) \geq k$, hence $u_1wu_2 \in hp(\theta, k)$. The second case is similar. □

Proposition 5 *The languages $hp(\theta,k)$ and $hpf(\theta,k)$, $k \geq 1$, are regular.*

Proof. The language $hp(\theta,k)$ can be written as $hp(\theta,k) = \bigcup_{|w|\geq k} X^*wX^*\theta(w)X^*$. This language is furthermore equal to $L = \bigcup_{|w|=k} X^*wX^*\theta(w)X^*$. Every language $X^*wX^*\theta(w)X^*$with $|w| = k$ is regular, hence L is a union of a finite number of regular languages. Therefore both $hp(\theta,k)$ and its complement $hpf(\theta,k)$ are regular. □

Corollary 1 *Let θ be a morphic or antimorphic involution and $k \geq 1$. The following problem of* hairpin-freedom *is decidable in linear time w.r.t. $|M|$:*

Input: *An NFA M.*

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

Proof. By definition, $L(M)$ is hp(θ,k)-free *iff* $L(M) \subseteq hpf(\theta,k)$ *iff* $L(M) \cap hp(\theta,k) = \emptyset$. This problem is solvable in linear time for regular languages. □

Also the *maximal hairpin-freedom* problem is highly relevant: given a hairpin-free language L_1 and a pool of DNA words L_2, is it possible to add new words from L_2 to L_1 without breaking its hairpin-freedom?

Corollary 2 *Let θ be a morphic or antimorphic involution and $k \geq 1$. The following problem is decidable in linear time w.r.t. $|M_1| \cdot |M_2|$:*

Input: *A DFA M_1 such that $L(M_1)$ is $hp(\theta,k)$-free, 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 $hp(\theta,k)$-free.*

Proof. We want to determine if 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 $\mathcal{O}(|M_1| \cdot |M_2|)$ whether $(hpf(\theta,k) \cap L(M_2)) - L(M_1) = \emptyset$. The size of an automaton accepting $hpf(\theta,k)$ 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 $|K| * l$, 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 adopt often the form of a block code.

The above results can be extended also for the case of context-free languages.

Corollary 3 *Let θ be a morphic or antimorphic involution and $k \geq 1$. The following problem is decidable in cubic time w.r.t. $|M_1| \cdot |M_2|$:*

Input: *A DPDA M_1 such that $L(M_1)$ is $hp(\theta,k)$-free, 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 $hp(\theta,k)$-free.*

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 $M_1 = (Q_1, X, \Gamma, q_1, Z_0, A_1, \delta_1)$, and let $M_2' = (Q_2, X, q_2, A_2, \delta_2)$ be a NFA accepting the language $hpf(\theta, k) \cap L(M_2)$. Consider the PDA $M = (Q, X, \Gamma, q_0, Z_0, A, \delta)$, 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) $\delta((p,q), a, Z) = \{((p', q'), \alpha) | (p', \alpha) \in \delta_1(p, a, Z) \text{ and } \delta_2(q, a) = q'\}$

(2) $\delta((p,q), \lambda, Z) = \{((p', q), \alpha) | (p', \alpha) \in \delta_1(p, \lambda, Z)\}$

$A = \{(p,q) | p \notin A_1 \text{ and } q \in A_2\}$ Then $L(M) = (hpf(\theta, k) \cap L(M_2)) - L(M_1)$, and the size of M is $\mathcal{O}(|M_1| \cdot |M_2|)$. Let G be a CFG such that $L(G) = L(M)$. Note that the construction of G takes cubic time w.r.t. $|M|$ (see Theorem 7.31 of [6]). Finally, it is possible to decide in linear time w.r.t. $|G|$ (see Section 7.4.3 of [6]) whether $L(G) = \emptyset$ or not. □

We can similarly extend the hairpin-freedom result in Corollary 1. In fact, the result is true for PDAs in general, including nondeterministic ones. The proof is remained to the reader.

Corollary 4 *Let θ be a fixed morphic or antimorphic involution and $k \geq 1$. The following problem is decidable in cubic time w.r.t. $|M|$:*

Input: *A PDA M.*

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

4 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 [1] and [12]. Motivated by this observation, we consider now a generalization of hairpins.

Definition 4 Let θ be an involution of X^* and let k be a positive integer. A word $u = wy \in X^*$ is θ-k-scattered-hairpin-free or simply shp(θ, k)-free if $t \leq_e w$, $\theta(t) \leq_e y$ implies $|t| < k$.

The set of words which are not shp(θ, k)-free characterizes DNA structures such as shown on Fig. 2. The depicted hairpin incorporates a bulge which is caused by a mismatched nucleotides within a double helix.

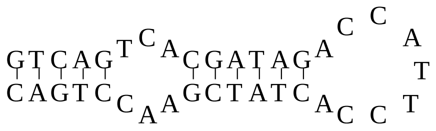

Figure 2: Bulge.

Definition 5 We denote by $shpf(\theta, k)$ the set of all shp(θ, k)-free words in X^*, and by $shp(\theta, k)$ its complement $X^* - shpf(\theta, k)$.

Example. Any word in $\{A, C\}^*$ is shp(θ, k)-free for every $k \geq 1$.

Definition 6 A language L is called θ-k-scattered-hairpin-free or simply shp(θ,k)-free if $L \subseteq shpf(\theta, k)$.

Lemma 3 $shp(\theta, k) = S\Big(\bigcup_{w \in X^k} w\theta(w)\Big)$.

Based on the above immediate result, analogous statements as in Section 3.1 hold also for scattered hairpins. Proofs are straightforward and left to the reader.

Proposition 6

(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 $shp(\theta, k) = S(H)$.

Corollary 5

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

(ii) The maximal scattered-hairpin-freedom *problem is decidable in linear time for regular languages and in cubic time for deterministic context-free languages.*

5 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 3. Second, simple hairpins are not always undesirable from the point of view of DNA computing. There exist nanomachines using simple hairpins, see e.g. [21]. Hairpins play also important role in some DNA computing techniques as we mentioned in Section 1. Hence it may be desirable to design DNA strands forming simple hairpins but avoiding more complex structures. Therefore we consider another extension of the results from Section 3.1.

Definition 7 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 a *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:

$$u = x_1 v_1 y_1 \theta(v_1) z_1 x_2 v_2 y_2 \theta(v_2) z_2 \cdots x_j v_j y_j \theta(v_j) z_j$$

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

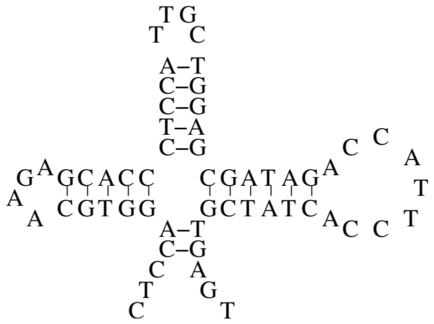

Figure 3: Secondary structure of a DNA strand with a hairpin frame.

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. 3.

A word u is said a *hp(fr,j)-word* if it contains at least one hp-frame of degree j. Observe that there may be more ways how to find 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 8 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^*.

Example. The word in Fig. 3 is in $hp(\theta, fr, 3)$, where $\theta = \mu\gamma$.

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 4 $hp(\theta, fr, j) = hp(\theta, 1)^j = \left(\bigcup_{a \in X} X^* a X^* \theta(a) X^* \right)^j.$

Proposition 7

(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 6

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

(ii) The maximal hp(fr,j)-freedom *problem is decidable in linear time for regular languages and in cubic time for deterministic context-free languages.*

6 Construction of Long Hairpin-Free Words

The following results hold for the case $\theta = \epsilon$. 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 [8] 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. See [8, 18] for background information on codes.

Proposition 8 *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 Fa is a uniform solid code of length $k : Fa \subseteq X^k$. 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 by $k-1-m$ matrix G over X_1 such that $F = \{w * [I_m|G] : w \in X_1^m\}$, where I_m is the identity m by m matrix and $*$ is the multiplication operation between a 1 by m vector and an m by 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 $x = wG$.

Proposition 9 *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 Fa.*

Proof. It is sufficient to show that $H(Fa) = \lfloor k/2 \rfloor$ and there are no different words in Fa with a common prefix of length $\lfloor k/2 \rfloor$. Obviously $H(Fa) = H(F) = \lfloor k/2 \rfloor$. 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 Fa 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 $n \leq \text{card}(F)$, for every choice of n different words in Fa. 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 Grant Agency of Czech Republic, Grant 201/02/P079 to P.S.

References

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

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

[3] R. Deaton, R. Murphy, M. Garzon, D.R. Franceschetti, S.E. Stevens. Good encodings for DNA-based solutions to combinatorial problems. *DNA-based computers II*, in AMS DIMACS Series, vol.44, L.F.Landweber, E.Baum Eds., 1998, 247–258.

[4] J. Chen, R. Deaton, M. Garzon, J.W. Kim, D. Wood, H. Bi, D. Carpenter, Y.-Z. Wang. Characterization of non-crosshybridizing DNA oligonucleotides manufactured *in vitro*. In [16], 132–141.

[5] T. Harju, J. Karhumäki. Morphisms. In [20], 439–510.

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

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

[8] H. Jürgensen, S. Konstantinidis. Codes. In [20], 511–607.

[9] L. Kari. DNA computing: arrival of biological mathematics. *The Mathematical Intelligencer*, vol.19, nr.2, Spring 1997, 9–22.

[10] L. Kari, S. Konstantinidis, E. Losseva, G. Wozniak. Sticky-free and overhang-free DNA languages. *Acta Informatica* 40, 2003, 119–157.

[11] L. Kari, S. Konstantinidis, P. Sosík. Preventing undesirable bonds between DNA codewords. In [16], 375–384.

[12] L. Kari, S. Konstantinidis, P. Sosík. Bond-free languages: formalizations, maximality and construction methods. In [16], 16–25.

[13] S. Kobayashi. Testing Structure Freeness of Regular Sets of Biomolecular Sequences. In [16], 395–404.

[14] A. Marathe, A. Condon, R. Corn. On combinatorial DNA word design. *DNA based Computers V*, DIMACS Series, E.Winfree, D.Gifford Eds., AMS Press, 2000, 75–89.

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

[16] G. Mauri, C. Ferretti., Eds., *DNA 10, Tenth International Meeting on DNA Computing.* Preliminary proceedings, University of Milano-Bicocca, 2004.

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

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

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

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

[21] N. Takahashi, A. Kameda, M. Yamamoto, A. Ohuchi, Aqueous computing with DNA hairpin-based RAM. In [16], 50–59.

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