Acta Cybernetica **14** (1999) 179-192.

Watson-Crick Walks and Roads on DOL Graphs*

Arto Salomaa †

Abstract

Apart from the massive parallelism of DNA strands, the phenomenon known as Watson-Crick complementarity is basic both in the experiments and theory of DNA computing. The parallelism makes exhaustive searches possible, whereas the complementarity is a powerful computational tool. This paper investigates complementarity as a language-theoretic operation: "bad" words obtained through a generative process are replaced by their complementary ones. While this idea is applicable to any generative process, it seems particularly suitable for Lindenmayer systems. DOL systems augmented with a specific complementarity transition, "Watson-Crick DOL systems", are investigated in this paper. Many issues involved are conveniently expressed in terms of certain paths, "Watson- Crick walks", in an associated digraph.

Keywords: DNA computing, Lindenmayer systems, DOL sequences, Watson-Crick complementarity.

1 Introduction

Adleman's celebrated experiment, [1], demonstrated how methods of molecular biology can possibly be applied to problems intractable by ordinary computational methods. Since then the interest in "DNA computing" has been growing rapidly, see the list of references in [6]. The impact of the resulting new notions and ideas to the theory of formal languages is visible from the recent Handbook, [8].

A keynote in theoretical studies about DNA computing is a phenomenon known as *Watson-Crick complementarity.* DNA (deoxyribonucleic acid) consists of polymer chains, referred to as *DNA strands.* A chain is composed of *nucleotides* or *bases.* The four DNA bases are customarily denoted by A (adenine), C (cytosine), G (guanine) and T (thymine). A DNA strand can be viewed as a word over the *DNA alphabet* $\Sigma_{DNA} = \{A, C, G, T\}$. The familiar DNA double helix arises by the boundage of two strands. The Watson-Crick complementarity comes into the

*Dedicated to Ferenc Gécseg on his 60th birthday. My "Super-Brother Feri" has been an invaluable companion on the paths of science and a true friend in everyday life. I owe him much and wish him sunny days on the road ahead.

†Academy of Finland and Turku Centre for Computer Science Lemminkäisenkatu 14 A FIN-20520 Turku, Finland, e-mail: asalomaa@utu.fi.

picture in the formation of such *double strands*. The bases A and T are *complementary*, and so are the bases C and G. Bonding occurs only if the bases in the corresponding positions in the two strands are complementary.

Consider the letter-to-letter endomorphism h_W of Σ^*_{DNA} defined by

$$h_W(A) = T,\ h_W(T) = A,\ h_W(G) = C,\ h_W(C) = G.$$

The morphism h_W will be referred to as the *Watson-Crick morhism*. Thus, a DNA strand X bonds with $h_W(x)$ to form a double strand. (We ignore here the orientation of the strands, indicated customarily by speaking of the 5′- and 3′-ends of a strand. We also would like to point out that we use the nowadays standard term "DNA computing" although, in our estimation, "DNA-based computing" would be more appropriate.) The complementarity of two strands leads (under appropriate conditions) to bondage. By encoding information on the original strands in a clever way, far-reaching conclusions can be made from the mere fact that bondage has occurred. This means that the phenomenon of complementarity provides computing power. The idea of using the fundamental knowledge, concerning how the double strands have possibly come into being, is central in Adleman's experiment, [1]. The idea is also behind the computational universality of many models of DNA computing, [9], [6].

Complementarity can be viewed also as a language-theoretic operation. As such h_W is only a morphism of a special kind. However, the operational complementarity can be considered also as a tool in a developmental model: undesirable conditions in a string *trigger* a transition to the complementary string. Thus, the class of "bad" strings is somehow specified. Whenever a bad string x is about to be produced by a generative process, the string $h_W(x)$ is taken instead of x. If the generative process produces a unique sequence of strings (words), the sequence continues from $h_W(x)$. The class of bad strings has to satisfy the following *soundness condition*: whenever x is bad, the complementary string $h_W(x)$ is not bad. This condition guarantees that no bad strings are produced.

While the operational complementarity can be investigated in connection with any generative process for words, it seems particularly suitable for *Lindenmayer systems*, the systems themselves being developmental models. The simplest L system, namely the DOL system, has been thoroughly investigated, [7]. A DOL system generates a sequence of words. When it is augmented with a trigger for complementarity transitions, as described above, the resulting sequences contain no bad words. The study of such "Watson-Crick DOL systems" was begun in [4] and [5], and will be continued in the present paper. The present paper is largely self-contained. In particular, no knowledge of [4] and [5] is required on the part of the reader. For more information about formal languages, L systems or DNA computing, the references [10], [7] or [6], respectively, may be consulted.

The formal definitions will be given below. An important remark should be made already at this stage. So far we have spoken only of the four-letter DNA alphabet but in our theoretical considerations below the size of the alphabet will

be arbitrary. Indeed, we will consider *DNA-like alphabets*

$$\Sigma_n = \{a_1, \ldots, a_n, \bar{a}_1, \ldots, \bar{a}_n\} \quad (n \geq 2)$$

and refer to the letters a_i and $\bar{a}_i$, $i = 1, \ldots, n$, as *complementary*. The endomorphism h_W of Σ_n^* defined by

$$h_W(a_i) = \bar{a}_i, \; h_W(\bar{a}_i) = a_i, \; i = 1, \ldots, n,$$

is also now referred to as the *Watson-Crick morphism*. When we view the original DNA alphabet in this way, the association of letters is as follows:

$$a_1 = A, \; a_2 = G, \; \bar{a}_1 = T, \; \bar{a}_2 = C.$$

(Observe that this conforms with the two definitions of h_W.) The nucleotides A and G are *purines*, whereas T and C are *pyrimidines*. This terminology is extended to concern DNA-like alphabets: the non-barred letters $a_1, \ldots, a_n$ are called *purines*, and the barred letters $\bar{a}_1, \ldots, \bar{a}_n$ are called *pyrimidines*. The class of bad words, considered most frequently in the sequel, consists of words where the pyrimidines form a majority.

In spite of their formal simplicity, Watson-Crick DOL systems have quite remarkable properties. This observation made already in [4] and [5] will be further substantiated in this paper. In particular, we will be concerned with basic decision problems. The following decision problem turns out to be very significant.

Problem Z_{pos}. Decide whether or not a negative number appears in a given Z-rational sequence of integers.

The decidability status of Z_{pos} is open, although the problem is generally believed to be decidable. The input is of course assumed to be given by some effective means such a linear recurrence with integer coefficients, or a square matrix M with integer entries such that the sequence is read from the upper right corners of the powers M^i, $i = 1, 2, 3, \ldots$. Further discussion about this problem and its different representations can be found in [3] and [7].

Ordinary DOL systems have been widely investigated and their properties are fairly well understood, whereas rather little is known about Watson-Crick DOL systems. It was already observed in [5] that graphs associated to them, as well as certain paths in such graphs, are very useful for studying the systems. Such "Watson-Crick walks and roads" will be investigated in this paper from a more general point of view.

2 Graphs associated to DOL systems

We will use standard language-theoretic notation. In particular, λ is the empty word, $|w|$ is the length of the word w, and $|w|_a$ (resp. $|w|_\Sigma$) is the number of occurrences of a (resp. letters of Σ) in w. The minimal alphabet of a word w is denoted by $alph(w)$.

An equivalence relation $\equiv$ on Σ^* is called a *morphic equivalence* if it preserves all endomorphisms of Σ^*, that is, whenever h is an endomorphism of Σ^* and $x \equiv y$ then also $h(x) \equiv h(y)$. Typical examples of morphic equivalences are:

(i) $alph(x) = alph(y)$,

(ii) x and y are powers of the same primitive root,

(iii) x and y have the same Parikh vector.

Of (i)–(iii), only (i) is of finite index. Note also that the equivalence defined by $|x| = |y|$ is not morphic.

A *DOL system* is a triple $G = (\Sigma, g, w_0)$, where Σ is an alphabet, $w_0 \in \Sigma^*$ (the *axiom*) and g is an endomorphism of Σ^*. (In the sequel g is often defined in terms of *productions*, indicating the image of each letter.) A DOL system defines the *sequence* $S(G)$ of words w_i, $i \geq 0$, where $w_{i+1} = g(w_i)$, for all $i \geq 0$. It defines also the *language* $L(G)$, consisting of all words in $S(G)$, the *length sequence* $|w_i|$, $i \geq 0$, as well as the *growth function* $f(i) = |w_i|$.

Given a DOL system $G = (\Sigma, g, w_0)$ and a morphic equivalence $\equiv$ on Σ^*, the *associated graph* $H(G, \equiv)$ is defined as follows. As a preparation for the sequel, we give this simple definition inductively, denoting the equivalence class of a word w by $[w]$. First the initial node of H, labeled by the equivalence class $[w_0]$, is created. Whenever a node labeled by $[w_i]$ has been created but no node labeled by $[g(w_i)]$ has been created, then the latter node is created and an arrow labeled by 0 is drawn from the former to the latter node. If the node labeled by the equivalence class $[g(w_i)]$ has already been created and denoted by, say, $[w_j]$ then an arrow labeled by 0 is drawn from the node $[w_i]$ to the node $[w_j]$.

Thus, all arrows (edges) in the (di)graph H are labeled by 0. (This is because H is a special case of the definition in the next section, where two labels are needed for the arrows.) The graph H is infinite if all words in $S(G)$ belong to different equivalence classes of $\equiv$. Starting from the initial node, an i-step *walk* (path) ends at a node labeled by the equivalence class $[w_i]$, where w_i is the ith word in the sequence $S(G)$. If $\equiv$ is of finite index, the digraph H is finite and can be separated into an "initial mess" and a "loop" in the customary fashion. This fact can also be expressed as the following theorem.

Theorem 2.1 *Let G be a DOL system and $\equiv$ a morphic equivalence (with the same alphabet) of finite index. Then the equivalence classes represented by the words in $S(G)$ form an ultimately periodic sequence.*

Proof. The claim follows by the construction of the graph H. Since $\equiv$ is of finite index, some word in the sequence $S(G)$, say $w_i = g(w_{i-1})$, represents an equivalence class already represented by w_j, for some $j < i$. If i has its smallest possible value, the words $w_0, \ldots, w_{j-1}$ represent equivalence classes in the "initial mess" and the words $w_j, \ldots, w_i$ equivalence classes in the "loop" of the ultimately periodic sequence. □

Theorem 2.1 is a general formulation of many known periodicity results concerning DOL sequences. For instance, the alphabets $alph(w_i)$ and prefixes or suffices of fixed lengths form ultimately periodic sequences, [7]. Such results are immediate corollaries of Theorem 2.1.

Observe that the graph H may be finite although $\equiv$ is of infinite index. For instance, assume that $x \equiv y$ iff x and y are powers of the same primitive root and that the DOL system G is determined by the axiom ab and prodution $a \to aba$, $b \to a$. Then the graph $H(G, \equiv)$ consists of only one node because all words in the sequence $S(G)$ are powers of the word ab.

Assume now that the alphabet Σ of the given DOL system $G = (\Sigma, g, w_0)$ actually is a DNA-like alphabet, $\Sigma = \Sigma_n$, and that the Watson-Crick morphism h_W is defined as in Section 1. (Observe that some letters of Σ_n might not occur in $S(G)$.) Then we define the *Watson-Crick graph* $H_W(G, \equiv)$ associated to G and a morphic equivalence $\equiv$ as follows. We are now dealing with two morphisms: g and the composition $h_W g$ (meaning that first g, then h_W is applied). The edge labels 0 and 1, respectively, are associated to these morphisms, respectively.

To construct $H_W(G, \equiv)$, we again first create the *initial node* labeled by $[w_0]$. Assume that a node labeled by $[w]$ has already been created and no node labeled by $[g(w)]$ (resp. $[h_W g(w)]$) has been created, then the latter node is created and an arrow labeled by 0 (resp. by 1) is drawn from the node $[w]$ to the newly created node. If a node labeled by $[g(w)]$ (resp.$[h_W g(w)]$) has been created and denoted by, say, $[w']$ (resp. $[w'']$) then an arrow labeled by 0 (resp. by 1) is drawn from the node $[w]$ to $[w']$ (resp. to $[w'']$).

Thus, H_W is a (possibly infinite) (di)graph, where exactly two arrows emanate from each node. A sufficient but not necessary condition for the finiteness of H_W is that the morphic equivalence $\equiv$ is of finite index. The smallest possible graph H_W consists of a single node $[w_0]$ with two arrows, labeled by 0 and 1, emanating from and going into $[w_0]$. This situation arises, for instance, if $x \equiv y$ is defined by the condition $alph(x) = alph(y)$, and every letter of Σ occurs in every word of $S(G)$. (It is easy to verify that in such a case an application of $h_W g$ never leads to smaller alphabets.) This smallest possible graph $H_W(G, \equiv)$ is referred to as *trivial.*

A *walk* W in the graph $H_W(G, \equiv)$ is any finite path beginning from the initial node. Walks in graphs $H(G, \equiv)$ can be described (equivalently) either by the sequence of nodes or by the sequence of edges because there is only one possibility at each node. In graphs $H_W(G, \equiv)$ there are two possibilities, perhaps both leading to the same node (as is the case, for instance, with the trivial graph). Consequently, walks in H_W must be described by listing the sequence of *edges* visited. In this fashion, we get a one-to-one correspondence between walks in H_W and words over the binary alphabet $\{0, 1\}$.

Many types of questions can be asked concerning the notions introduced in this paper – we will focus on some *decision problems.* Therefore, the following *general observation* is significant. Consider decision problems (for instance, the emptiness problem) involving languages of the form $L \cap K$, where L is a DOL language and K is in one of the classes of the Chomsky hierarchy. Such problems are usually decidable (resp. undecidable) if K ranges over regular (resp. context-sensitive) languages.

If K ranges over context-free languages, the decidability is often hard to settle, although intuitively the problem might seem to be decidable. Such problems are often algorithmically equivalent to the problem Z_{pos}.

3 Watson-Crick DOL systems

Consider again a DNA-like alphabet Σ_n and the Watson-Crick morphism h_W. A *trigger* TR is any recursive subset of Σ_n^* satisfying the following condition: whenever x is in TR, then $h_W(x)$ is in the complement of TR, that is, in $\Sigma_n^* - TR$.

According to our terminology in the Introduction, TR consists of "bad" strings. The restriction imposed on TR is our *soundness condition*: no "bad" strings result if emerging "bad" words are always replaced by their complementary ones. We now come to our central definitions.

A *Watson-Crick DOL system* is a construct

$$G_W = (G,\ TR),$$

where $G = (\Sigma_n,\ g,\ w_0)$ is a DOL system, TR is a trigger and $w_0 \in \Sigma_n^* - TR$. The *sequence* $S(G_W)$, consisting of words w_i, $i = 0,\ 1,\ \ldots$, is defined by the condition

$$w_{i+1} = \begin{cases} h_w(g(w_i)) & \text{if } g(w_i) \in TR, \\ g(w_i) & \text{otherwise,} \end{cases}$$

for all $i \geq 0$. The *language*, *length sequence* and *growth function* of G_W are defined for $S(G_W)$ as for ordinary DOL systems.

The *Watson-Crick graph* $H_W(G_W,\ \equiv)$ associated to a Watson-Crick DOL system $G_W = (G,TR)$ and morphic equivalence $\equiv$ equals, by definition, the Watson-Crick graph $H_W(G, \equiv)$. (Thus, $H_W(G_W,\ \equiv)$ is independent of the trigger.) A *Watson-Crick walk* $WW(G_W,\ \equiv)$ associated to G_W and $\equiv$ is the walk in $H_W(G_W,\ \equiv)$ determined by the binary word $u_1 \ldots u_k$ such that, for $1 \leq i \leq k$, $u_i = 0$ (resp. $u_i = 1$) if $w_i = g(w_{i-1})$ (resp. $w_i = h_W(g(w_{i-1}))$) in $S(G_W)$.

Thus, the binary word $u_1 \ldots u_k$, determining the sequence of *edges* visited, is actually independent of the equivalence $\equiv$. If we are only interested in the sequence of edges, we may speak of the Watson-Crick walk of G_W, without specifying the equivalence. The latter becomes important if we are interested in the sequence of *nodes* visited. Observe that, viewed as a sequence of edges, the Watson-Crick walk in the trivial graph can be quite complicated. This is exemplified in Theorem 3.3 below. The next theorem follows directly from the definitions.

Theorem 3.1 *Viewed as binary words, all Watson-Crick walks* $WW(G_W,\ \equiv)$ *are prefixes of the same infinite (binary) word* $WW(G_W)$. *Thus, each Watson-Crick walk of* G_W *is completely determined by its length.* □

The infinite word $WW(G_W)$ is called the *Watson-Crick road* of G_W. Two Watson-Crick DOL systems are called *road equivalent* if they have the same Watson-Crick road. A Watson-Crick DOL system G_W is called *stable* if its Watson-Crick road equals 0^ω (that is, the infinite word consisting of 0s).

Thus, the Watson-Crick road completely characterizes the complementarity transitions: letters 1 occur in positions such that a transition takes place at the corresponding step. A system being stable means that no complementarity transitions occur, that is, the sequence is obtained as in an ordinary DOL system. The *stability problem* is basic in the study of Watson-Crick DOL systems. In general, the problem is undecidable. We have the following more specific results.

Theorem 3.2 *The stability problem is decidable for Watson-Crick DOL systems with regular trigger but undecidable for systems with context-sensitive triggers.*

Proof. A Watson-Crick DOL system $G_W = (G, TR)$ is stable iff the intersection $L(G) \cap TR$ is empty, where $L(G)$ is the language of G, viewed as an ordinary DOL system. But the emptiness of such an intersection is decidable (resp. undecidable) for regular (resp. context-sensitive) triggers TR, see [7] (resp. [2]). □

We now show that Watson-Crick roads can be arbitrarily complex, even if attention is restricted to systems whose Watson-Crick graph is trivial. Let φ be a recursive function mapping the set of positive integers into $\{0, 1\}$. We denote by u_φ the infinite binary word whose ith letter equals 1 exactly in case $\varphi(i) = 1$, for all $i \geq 1$.

Theorem 3.3 *For every recursive function φ, a Watson-Crick DOL system whose Watson-Crick road equals u_φ can be effectively constructed. Moreover, the items involved can always be chosen in such a way that the morphic equivalence is defined by the relation $alph(x) = alph(y)$ and that the associated Watson-Crick graph is trivial.*

Proof. Given φ, we construct a Watson-Crick DOL system $G_W = (G, TR)$ as follows. The alphabet of the DOL system G is $\Sigma_2 = \{a_1, a_2, \bar{a}_1, \bar{a}_2\}$. We prefer to write Σ_2 as the original DNA alphabet $\{A, T, C, G\}$ in the way indicated in Section 1 (trusting that the slight notational ambiguity causes no confusion). The axiom of the system is $ACGT$, and the morphism g is defined by the rules

$$A \to A, \quad C \to C^2, \quad G \to G^2, \quad T \to T.$$

The morphic equivalence is defined by the condition: $x \equiv y$ iff $alph(x) = alph(y)$. Then (independently of TR which we have not yet defined) the Watson-Crick graph $H_W(G_W, \equiv)$ is trivial. This follows because each word in the sequence $S(G_W)$ equals either $w_1(i) = AC^{2^i}G^{2^i}T$ or $w_2(i) = TG^{2^i}C^{2^i}A$, for some i.

We now define the trigger by

$$TR = \{w_1(i), w_2(i) | i \in \varphi^{-1}(1)\}.$$

Clearly, TR is recursive. It is also easy to verify that this construction satisfies the theorem. □

A very natural trigger (and the only one considered in [5]) is the set of words, where the pyramidines (barred letters) are in strict majority. Thus, we denote $\Sigma_{pyr} = \{\bar{a}_1, \ldots, \bar{a}_n\}$ and consider the language

$$PYR = \{w \in \Sigma_n^* \mid |w|_{\Sigma_{pyr}} > \frac{|w|}{2}\}.$$

Clearly, PYR satisfies the soundness condition. Watson-Crick DOL systems (G, PYR) will be referred to as *standard*. Consequently, for a standard system G_W, in every word of $S(G_W)$ there are at least as many purines as pyrimidines.

Observe that PYR and its complement are context-free nonregular languages. Thus, considering Theorem 3.2, we are closer to the borderline between decidability and undecidability. Indeed, the following result was established in [5].

Theorem 3.4 *The stability problem for standard Watson-Crick DOL systems is algorithmically equivalent to the problem* Z_{pos}. □

An infinite binary word is referred to as *ultimately periodic* if it is of the form uv^ω, where $u \in \{0, 1\}^*$ and $v \in \{0, 1\}^+$. The trigger used in the general result Theorem 3.3 is very complicated. However, all ultimately periodic roads can be realized with simpler triggers.

Theorem 3.5 *Every ultimately periodic Watson-Crick road can be expressed in the form* $WW(G_W)$ *where* G_W *is standard (resp.* G_W *has a finite trigger).*

Proof. Assume uv^ω is the given word, where

$$u = b_1 \ldots b_k,\ v = b_{k+1} \ldots b_l,\ k \geq 0,\ l \geq k+1,\ b_j \in \{0, 1\}.$$

We construct a Watson-Crick DOL system $G_W = (G, TR)$. The alphabet of G equals $\{a_0, \ldots, a_l, \bar{a}_0, \ldots, \bar{a}_l\}$, the axiom is a_0 and the morphism is defined by the productions

$a_j \to a_{j+1}$ or $a_j \to \bar{a}_{j+1}$, depending whether $b_{j+1} = 0$ or $b_{j+1} = 1$, $0 \leq j \leq l-1$;
$a_l \to a_{k+1}$ or $a_l \to \bar{a}_{k+1}$, depending whether $b_{k+1} = 0$ or $b_{k+1} = 1$;
$\bar{a}_j \to \bar{a}_j,\ 0 \leq j \leq l$.

If we now choose $TR = PYR$ or $TR = \{\bar{a}_1, \ldots, \bar{a}_l\}$, the resulting system G_W has the Watson-Crick road uv^ω. □

For any Watson-Crick DOL system G_W and any morphic equivalence $\equiv$, every node in the graph $H_W(G_W, \equiv)$ is *reachable* in the sense that there is a walk ending with that node. This follows by the construction of H_W. However, the Watson-Crick road of G_W does not necessarily pass through all nodes of G_W.By the *reachability problem* we understand the problem of deciding, given G_W, $\equiv$ and a node N in H_W, whether or not the Watson-Crick road of G_W passes through N.

In this context we assume that the morphic equivalence is defined by the condition $alph(x) = alph(g)$, implying that H_W is finite. Examples can be given of cases, where N actually is reachable but the shortest prefix of the Watson-Crick road of G_W ending with N is very long (for instance, in terms of the number of nodes of H_W). This is natural, in view of the following theorem.

Theorem 3.6 *The reachability problem is undecidable. For standard Watson-Crick DOL systems G_W, any algorithm for solving the reachability problem can be converted into an algorithm for solving the problem Z_{pos}.*

Proof. The second sentence has been established in [5]. To prove the first sentence, we show that the decidability of the problem would imply the decidability of the emptiness problem for languages $L(G) \cap K$, where K is context-sensitive and G is a DOL system. However, the latter problem is known to be undecidable.

Given K and G over the alphabet $\Sigma = \{a_1, \ldots, a_n\}$, we construct a Watson-Crick DOL system $G_W = (G', K)$. (Without loss of generality, we assume that the language K does not contain λ.) The DOL system G' is almost the same as G; we only extend the alphabet Σ to the DNA-like alphabet Σ_n and add the productions $\bar{a}_i \to \bar{a}_i$, $1 \leq i \leq n$. Clearly, K satisfies the soundness condition for triggers because, for $x \in K$, $h_W(x)$ consists of pyramidines and K contains no such words.

Two possibilities arise. If $L(G) \cap K = \phi$ then $S(G_W) = S(G)$ because no complementarity transition takes place. If $L(G) \cap K \neq \phi$ and w_i is the first word in $S(G)$ belonging to K (we may assume that w_i is not the axiom), then $S(G_W)$ coincides with $S(G)$ up to w_{i-1}, after which $S(G_W)$ consists of only repetitions of $\bar{w}_i$. Because of our agreement concerning the morphic equivalence, the latter alternative occurs exactly in case the Watson-Crick road of G_W passes through some node in H_W labeled by pyramidines. This is a question we can settle if we can decide the reachability problem. □

We conclude this section with an example of a standard Watson-Crick DOL system G_W, due originally to [4]. The alphabet of G_W is Σ_3, the axiom is $a_1 a_2 \bar{a}_3$, and the productions are

$$a_1 \to a_1,\ a_2 \to a_2,\ a_3 \to a_3,\ \bar{a}_1 \to \bar{a}_1\bar{a}_2,\ \bar{a}_2 \to \bar{a}_2,\ \bar{a}_3 \to \bar{a}_3^3.$$

The graph $H_W(G_W, \equiv)$ where $\equiv$ is again defined as in Theorem 3.6, consists of two nodes: the initial node N_1 labeled by $\{a_1, a_2, \bar{a}_3\}$, and the node N_2 labeled by $\{\bar{a}_1, \bar{a}_2, a_3\}$. The arrows labeled by 0 preserve both nodes, whereas the arrows labeled by 1 interchange them. The Watson-Crick road of G_W begins with the word $10110^5110^{17}11$. In general, there is an exponentially growing sequence of 0s between words 11. Explicitly, after the first position the bit 1 occurs exactly in positions $3^{i+1} + i$ and $3^{i+1} + i + 1$, for all $i \geq 0$. It is interesting to note that only three of the four arrows of H_W are used on the Watson-Crick road; the arrow from N_1 to itself is never used. The example shows the validity of the following theorem.

Theorem 3.7 *The Watson-Crick road of a standard system is not necessarily ultimately periodic.* □

Theorem 3.7 should be contrasted to the many results about context-free language showing ultimate periodicity; recall that the trigger in a standard system is context-free.

The growth function of the system G_W fluctuates between a linear function (due to the productions $\bar{a}_1 \to \bar{a}_1\bar{a}_2$, $\bar{a}_2 \to \bar{a}_2$) and an exponential function (due to $\bar{a}_3 \to \bar{a}_3^3$). Such a fluctuation is not possible for DOL growth functions. Indeed, it is shown in [4] that the growth function of G_W is not Z-rational. The system G_W is the smallest standard system with these properties (strange growth function and nonperiodicity) we have been able to find. It would be interesting to have similar examples with the original DNA alphabet or, equivalently, Σ_2.

4 Equivalence problems

The decidability of various *equivalence problems* constitutes a central chapter in the history of L systems, see [7]. We use here the standard terminology. Thus, the *sequence* (resp. *growth*) *equivalence problem* for DOL systems consists of deciding of two given DOL systems whether or not they generate the same sequence (resp. growth function). For DOL systems, the decidability of the growth equivalence problem was settled first. It was also shown quite early, that the decidability of the sequence equivalence implies the decidability of the language equivalence and vice versa, whereas the decidability itself remained as a celebrated open problem, until it was finally settled in the late 70s, see [7].

Clearly, the *sequence*, *language* and *growth equivalence problems* can be formulated for Watson-Crick DOL systems exactly as for ordinary DOL systems. In addition, we have the very natural *road equivalence problem* for Watson-Crick DOL systems: given two systems, decide whether or not they define the same Watson-Crick road. Thus, the road equivalence of two Watson-Crick DOL systems means only that the complementarity transitions occur in the two sequences at the same steps; the two sequences themselves can be very different. For instance, two stable systems are always road equivalent.

Thus, road equivalence does not imply sequence, language or growth equivalence. On the other hand, sequence equivalence (which implies language and growth equivalence) does not imply road equivalence. For instance, consider two standard Watson-Crick DOL systems G_1 and G_2 over the DNA alphabet. The axiom of both systems is AG. The productions in G_1 are

$$A \to T, \quad G \to C, \quad C \to C^2, \quad T \to T^2,$$

whereas in G_2 they are

$$A \to A, \quad G \to G, \quad C \to C^3, \quad T \to T^3.$$

Then $S(G_1) = S(G_2)$ but G_1 and G_2 are not road equivalent. In fact, the Watson-Crick roads of G_1 and G_2 are 1^ω and 0^ω, respectively. (Observe that it is irrelevant in both systems how we choose the productions for C and T.)

It is also possible that two systems are sequence equivalent when viewed as ordinary DOL systems but not as Watson-Crick DOL systems, and vice versa. We have seen that the above systems G_1 and G_2 are sequence equivalent when viewed as standard (implying that $TR = PYR$) Watson-Crick DOL systems. They are clearly not sequence equivalent when viewed as ordinary DOL systems: after the axiom the two sequences differ at every step, and even the word lengths differ from the third word on. On the other hand, consider two systems G_3 and G_4 with the axiom A, where the productions in G_3 are

$$A \to CTC, \quad C \to CTC, \quad T \to \lambda, \quad G \to G^3,$$

and those in G_4 are

$$A \to CTC, \quad C \to C, \quad T \to TCCT, \quad G \to G^4.$$

Viewed as ordinary DOL systems, G_3 and G_4 are sequence equivalent. Indeed, $S(G_3) = S(G_4)$ begins with A, followed by the words $(CTC)^{2^i}$, $i \geq 0$. If G_3 is viewed as a standard Watson-Crick DOL system, the sequence $S(G_3)$ consists of the words

$$A, \quad GAG, \quad G^3CTCG^3, \quad G^{3^{i+1}}(CTC)^{2^i}G^{3^{i+1}}, \quad i \geq 1.$$

If G_4 is viewed similarly, the sequence $S(G_4)$ consists of the words

$$A, \quad GAG, \quad G^4CTCG^4, \quad G^{4^{i+1}}(CTC)^{2^i}G^{4^{i+1}}, \quad i \geq 1.$$

Consequently, G_3 and G_4 are not sequence equivalent. Observe, however, that G_3 and G_4 are road equivalent: both define the Watson- Crick road 10^ω.

The above examples serve the purpose of illustrating the great variety of problems of new types brought forward by the different notions of equivalence. Indeed, some challenging decision problems in this area remain open. According to the general observation made at the end of Section 2, it is to be expected that equivalence problems involving arbitrary (resp. regular) triggers are undecidable (resp. decidable). The case of arbitrary (context-sensitive) triggers will be dealt with in Theorem 4.1, whereas we hope to return to regular triggers in a forthcoming paper. Equivalence problems involving context-free triggers (as is the case with standard systems) are very challenging. Intuitively, the problems seem to be decidable. But, as shown in Theorem 4.2, they are at least as hard as the problem Z_{pos}.

Theorem 4.1 *The road, growth, sequence and language equivalence problems are all undecidable for Watson-Crick DOL systems with context-sensitive triggers.*

Proof. We use again the undecidability of the emptiness of the intersection $L(G) \cap K$, where G is an ordinary DOL system and K is a context-sensitive language. Indeed, the argument is similar to the one used in the proof of Theorem 3.6.

Assume that we are given a DOL system G and a context-sensitive language K over the alphabet Σ. Without loss of generality, we assume that $L(G)$ is infinite and that the axiom of G is not in K. We can also find a word $u \in \Sigma^+$ not in $L(G)$.

We now construct two Watson-Crick DOL systems G' and G'' by taking the axiom of G, extending the alphabet Σ to a DNA-like alphabet Σ_n by adding to Σ the barred version $\bar{a}$ of each letter a and, finally, by adding to G all productions $\bar{a} \to \bar{a}$, where $a \in \Sigma$. The systems $G'$and G'' are identical except that G' has the trigger $\{u\}$, whereas K is the trigger of G''.

It is now easy to verify, exactly as in the proof of Theorem 3.6, that $G'$and G'' are road, growth, sequence or language equivalent exactly in case $L(G) \cap K = \phi$. Indeed, the Watson-Crick road of G' equals 0^ω, and the growth function, sequence and language of G' coincide with that of G. Each of these statements holds for G'' exactly in case G'' is stable, that is, $L(G) \cap K = \phi$. (Our assumption concerning the infinity of $L(G)$ is needed to justify this conclusion for growth functions.) □

Theorem 4.2 *Any algorithm for solving the road, growth, sequence or language equivalence problem for standard Watson-Crick DOL systems can be converted into an algorithm for solving the problem Z_{pos}.*

Proof. By Theorem 3.4, it suffices to show that an algorithm for solving any of the four equivalence problems for standard Watson-Crick DOL systems can be converted into an algorithm for solving the stability of standard Watson-Crick DOL systems.

Thus, we have to decide whether or not a given Watson-Crick DOL system $G_W = (G, PYR)$ is stable, where $G = (\Sigma_n, g, w_0)$ is a DOL system. Here Σ_n is a DNA-like alphabet and, thus, consists of $2n$ letters. We now extend Σ_n to a DNA-like alphabet $\Sigma_{2n} = \Sigma_n \cup \bar{\Sigma}_n$, where $\bar{\Sigma}_n$ consists of barred versions of letters of Σ_n. (The new bars should not be confused with those appearing in letters of Σ_n.) In connection with Σ_{2n}, the letters of $\bar{\Sigma}_n$ are considered pyrimidines and the set $PYR \subseteq \Sigma_{2n}^*$ is defined accordingly. Consider also the extension g' of g to Σ_{2n}^*, where $g'(a) = a$ for all $a \in \bar{\Sigma}_n$, and define the standard Watson-Crick DOL system $G'_W = (G', PYR)$, where $G' = (\Sigma_{2n}, g', w_0)$ and PYR is defined in connection with Σ_{2n}. Clearly, G'_W is stable and defines the Watson-Crick road 0^ω. Consequently, G_W is stable exactly in case G_W and G'_W are road, sequence or language equivalent. This means that an algorithm for deciding one of these three equivalence problems decides also the stability problem.

The same conclusion cannot be made directly as regards the growth equivalence problem: it is conceivable that G_W and G'_W are growth equivalent although complementarity transitions take place in G_W. However, the proof of Theorem 3.4 in [5] is easily modified to exclude this possibility. In this proof, the Z-rational sequence (given for the problem Z_{pos}) was expressed as the difference of two DOL length sequences, generated by systems both having n letters. When the systems are run simultaneously and the letters of the original systems are viewed as purines and pyrimidines, the combined system is stable exactly in case the Z-rational sequence assumes never a negative value. We now consider two new letters a_{n+1} and $\bar{a}_{n+2}$, as well as their complementary ones $\bar{a}_{n+1}$ and a_{n+2}. The axiom of the combined system is catenated with the word $a_{n+1}^i \bar{a}_{n+2}^i$, where the new letters have the productions $a_{n+1} \to a_{n+1}^j$ and $\bar{a}_{n+2} \to \bar{a}_{n+2}^j$. Here i and j are large enough

for the new letters to dominate the growth function of the combined system. (The new letters contribute the same number of purines and pyrimidines and, thus, do not affect membership in PYR.) The choice $\bar{a}_{n+1} \to \bar{a}_{n+1}^{2j}$, $a_{n+2} \to a_{n+2}^{2j}$ now guarantees that a complementarity transition changes the growth function. □

We do not know whether Theorem 4.2 holds in the reverse order, that is, whether an algorithm for solving the problem Z_{pos} can be converted into an algorithm for solving the equivalence problems.

5 Conclusion

In Watson-Crick DOL systems one investigates a classical topic, Lindenmayer systems, from the new angle provided by the idea of complementarity in DNA computing. In this paper we have focused our attention to the fundamental information brought forward by the associated Watson-Crick road. Many problems in this area are very challenging, especially because of their interconnection with some celebrated open problems. It would be a fascinating project to apply DNA computing itself towards the solution of these problems.

References

[1] Adleman, L. Molecular computation of solutions to combinatorial problems. *Science* 266, 1994, 1021–1024.

[2] Harrison, J. Morphic congruences and DOL languages. *Theoretical Computer Science* 134, 1994, 537-544.

[3] Kuich, W. and Salomaa, A. *Semirings, Automata, Languages.* Springer-Verlag, Berlin, Heidelberg, New York, 1986.

[4] Mihalache, V. and Salomaa, A. Lindenmayer and DNA: Watson-Crick DOL systems. *EATCS Bulletin* 62, 1997, 160–175.

[5] Mihalache, V. and Salomaa, A. Language-theoretic aspects of DNA complementarity. Submitted for publication.

[6] Paun, G., Rozenberg, G. and Salomaa, A. *DNA Computing - New Computing Paradigms.* Springer-Verlag, Berlin, Heidelberg, New York, 1998.

[7] Rozenberg, G. and Salomaa, A. *The Mathematical Theory of L Systems.* Academic Press, New York, 1980.

[8] Rozenberg, G. and Salomaa, A. (eds) *Handbook of Formal Languages*, Vol. 1–3. Springer-Verlag, Berlin, Heidelberg, New York, 1997.

[9] Salomaa, A. Turing, Watson-Crick and Lindenmayer. Aspects of DNA complementarity. In Calude, C., Casti, J. and Dinneen, M. (eds) *Unconventional Models of Computation.* Springer-Verlag, Singapore, 1998, 94–107.

[10] Salomaa, A. *Formal Languages.* Academic Press, New York, 1973.