Watson–Crick D0L systems: the power of one transition

Arto Salomaa\textsuperscript{a}, Petr Sosík\textsuperscript{b,}\textsuperscript{*}

\textsuperscript{a}Turku Centre for Computer Science, Lemminkäisenkatu 14A, 20520 Turku, Finland
\textsuperscript{b}Institute of Computer Science, Silesian University, 746 01 Opava, Czech Republic

Received 2 January 2002; received in revised form 1 June 2002; accepted 19 June 2002
Communicated by G. Rozenberg

Abstract

We investigate the class of functions computable by uni-transitional Watson–Crick D0L systems: only one complementarity transition is possible during each derivation. The class is characterized in terms of a certain min-operation applied to $\mathbb{Z}$-rational functions. We also exhibit functions outside the class, and show that the basic decision problems are equivalent or harder than a celebrated open problem. For instance, the latter alternative applies to the growth-bound problem for functions in the class.

© 2002 Elsevier Science B.V. All rights reserved.

Keywords: Lindenmayer system; Iterated morphism; $\mathbb{Z}$-rational sequence; DNA computing; Watson–Crick complementarity

1. Introduction

The phenomenon known as \textit{Watson–Crick complementarity}, in addition of being a basic tool in DNA computing, has also opened surprising views in theoretical studies not directly connected with DNA computing. In particular, a simple language-theoretic mechanism may gain surprising power when augmented with a possibility of replacing a string with its complementary one, under certain circumstances. The purpose behind such a replacement is to remove “bad” words obtained through a generative process. This idea seems particularly suitable for \textit{Lindenmayer systems}. D0L systems augmented with a specific complementarity transition, \textit{Watson–Crick D0L systems}, have turned
out to be a particularly interesting model and have already been extensively studied. A language is generated by a Watson–Crick D0L system as a sequence of words. Consequently, the systems can be applied also to compute functions in a natural way.

In the present paper, attention is focused on uni-transitional systems, where at most one complementarity transition takes place in the generated sequence. In spite of their seeming simplicity, uni-transitional systems represent a vast extension of ordinary D0L systems. This becomes apparent in their capacity of defining functions. We begin with the central issues.

DNA 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 $\Sigma_{\text{DNA}} = \{A, C, G, T\}$. The familiar DNA double helix arises by the bondage of two strands. The Watson–Crick complementarity comes into the 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. Adenine and guanine are purines, whereas cytosine and thymine are pyrimidines. The chemical significance of this fact is irrelevant for us but some notions and notations below come from this terminology.

We will use standard language-theoretic terminology and notation. In particular, $\lambda$ 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 $\Sigma$) in $w$. Language-theoretic issues and general background material can be consulted from [3,13], facts about formal power series and rational sequences from [5,14], and questions concerning Lindenmayer systems from [9,14]. The seminal paper in DNA computing is [1], whereas [8] is a general exposition and [10] underlines the significance of complementarity.

Consider the letter-to-letter endomorphism $h_W$ of $\Sigma_{\text{DNA}}^*$ defined by

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

The morphism $h_W$ will be referred to as the Watson–Crick morphism.

A DNA-like alphabet $\Sigma$ is an alphabet with an even cardinality $2n$, $n \geq 1$, where the letters are enumerated as follows:

$$\Sigma = \{a_1, \ldots, a_n, \overline{a}_1, \ldots, \overline{a}_n\}.$$  

We say that $a_i$ and $\overline{a}_i$ are complementary letters. The letter-to-letter endomorphism $h_W$ over $\Sigma^*$ mapping each letter to the complementary letter is also now called the Watson–Crick morphism. Hence

$$h_W(a_i) = \overline{a}_i, \quad h_W(\overline{a}_i) = a_i, \quad 1 \leq i \leq n.$$  

When we view the original DNA alphabet as a DNA-like alphabet, the association of letters is as follows:

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

In analogy with the DNA alphabet, we call the non-barred letters purines and the barred letters pyrimidines.
The basic notion underlying our subsequent investigations is given in the following definition.

**Definition 1.1.** A D0L system is a triple $G=(\Sigma, g, w_0)$, where $\Sigma$ is an alphabet, $w_0 \in \Sigma^*$ (the axiom) and $g$ is an endomorphism of $\Sigma^*$. (In the sequel $g$ is often defined in terms of productions, indicating the image of each letter.) A D0L 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 $f(i) = |w_i|$. 

The following problem turns out to be very significant for our subsequent discussions.

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

The decidability status of $Z_{\text{pos}}$ 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 [5,9,13,14].

The next section gives the fundamental definitions, and all our results are presented in Sections 3 and 4. As regards the notions defined in the next section, Watson–Crick D0L systems were introduced in [6] and investigated further in [4,7,11], whereas [2,16] represent recent work about them. The notion of a function computed by a Watson–Crick D0L system was introduced and its universality established in [15]. Uni-transitional systems were introduced in [12].

2. Definitions

During a generative process of words, under some circumstances we want to replace a word $w$ by its complement $h_w(w)$. The set of all such “bad” words will be called the trigger. Naturally, we would require that if a word $w$ is “bad”, then $h_w(w)$ is “good”. This condition is called the soundness of the trigger $TR: w \in TR$ implies $h_w(w) \notin TR$. Various kinds of triggers (regular, context-free, context-sensitive) have been studied in [4,6,11]. It turns out that the most interesting is the standard trigger $PYR$ described below. Indeed, it is the only one considered in this paper.

Let $\Sigma$ be a DNA-like alphabet. The subset of $\Sigma^*$ consisting of all words with the number of occurrences of pyrimidines strictly greater than that of purines is denoted by $PYR$. The complement of $PYR$ is denoted by $PUR$. Clearly, both $PYR$ and $PUR$ are deterministic context-free nonregular languages. Observe that, as a trigger, $PYR$ satisfies the soundness condition mentioned above. We further use the notations $\Sigma^{\text{PUR}}=\{a_1,\ldots,a_n\}$ and $\Sigma^{\text{PYR}}=\{\overline{a_1},\ldots,\overline{a_n}\}$. 
Definition 2.1. A standard Watson–Crick D0L system is a construct $G = (\Sigma, g, w_0)$, where $\Sigma$ is a DNA-like alphabet, $\Sigma = \{a_1, \ldots, a_n, a_1^{-1}, \ldots, a_n^{-1}\}$, $g : \Sigma^* \to \Sigma^*$ is a morphism and the word $w_0 \in \text{PUR}$. The sequence $S(G)$ defined by $G$ is the sequence $w_0, w_1, w_2, \ldots$, where for $i \geq 0$,

$$w_{i+1} = \begin{cases} g(w_i) & \text{if } g(w_i) \in \text{PUR}, \\ h_W(g(w_i)) & \text{if } g(w_i) \in \text{PYR}. \end{cases}$$

The step to obtain $w_{i+1}$ from $w_i$ is called a derivation step of $G$ and it is denoted by $w_i \Rightarrow_G w_{i+1}$. If $w_{i+1} = h_W(g(w_i))$, then we say that a complementarity transition has taken place. We denote $\Rightarrow^*_G$ the transitive and reflexive closure of $\Rightarrow_G$ as usual. The language generated by $G$ is the set $L(G) = \{ w_i \mid w_0 \Rightarrow^*_G w_i \}$. Since we consider only standard Watson–Crick D0L systems in this paper (that is, PYR constitutes the trigger), we will omit the word “standard” in our subsequent discussions.

We also use constructs of the form $(\Sigma, p)$, called Watson–Crick D0L schemes. They are merely a Watson–Crick D0L systems without an axiom. Given a scheme $G = (\Sigma, p)$ and a string $w_0 \in \text{PUR}$, we denote by $S(G, w_0)$ the derivation sequence of the system $(\Sigma, p, w_0)$. (The interplay between systems and schemes is a common practice for Lindenmayer systems, see [9].)

We now introduce the special type of Watson–Crick D0L systems investigated in this paper.

Definition 2.2. A Watson–Crick D0L system $G = (\Sigma, g, w_0)$ operates in the uni-transitional mode if its sequence $(w_i), i \geq 0$ is defined by

$$w_{i+1} = \begin{cases} h_W(g(w_i)) & \text{if } i \text{ is the smallest index such that } g(w_i) \in \text{PYR}, \\ g(w_i) & \text{otherwise} \end{cases}$$

for all $i \geq 0$. Watson–Crick D0L systems operating in the uni-transitional mode are called uni-transitional systems or, briefly, UT-systems.

Based on the definition of a UT-system, its language and sequence are introduced as in Definition 2.1. The smallest index $k, k \geq 1$, such that $g(w_{k-1}) \in \text{PYR}$ is referred to as the transition point of the system. We also speak of UT-schemes.

The following definition is general in the sense that it is applicable to Watson–Crick D0L schemes, as well as to uni-transitional schemes.

Definition 2.3. Consider a (resp. uni-transitional) Watson–Crick D0L scheme $G = (\Sigma, g)$. A partial recursive function $f$ mapping a subset of the set of nonnegative integers into nonnegative integers is computed by $G$ if the alphabet $\Sigma$ contains the letters $B, b, E, e$ with the productions $E \rightarrow E$ and $e \rightarrow e$ and satisfying the following condition. For all $i \geq 0$, equation $f(i) = j$ holds exactly in case there is a derivation according to $G$

$$Bb^i \Rightarrow^* Ee^j$$

and, moreover, the letters $E$ and $e$ appear in this derivation at the last step only.
A function \(f\) is \textit{Watson–Crick computable} if it is computed by some Watson–Crick DOL scheme \(G\). A function \(f\) is \textit{UT-computable} if it is computed by some UT-scheme \(U\).

We call \(B\) and \(b\), resp. \(E\) and \(e\) the \textit{input}, resp. the \textit{output symbols} of \(\Sigma\). According to the result established in [15], every partial recursive function is Watson–Crick computable.

3. Characterization of UT-computable functions

Consider a function \(f : \mathbb{N} \to \mathbb{N}\) computed by a UT-scheme \(G\) in the sense of Definition 2.3. Observe first that if the string of the form \(Ee^j\), \(j \geq 0\), does not appear in the sequence \(S(G, Bh^i)\) under the conditions stated in Definition 2.3, then the value of \(f(i)\) is undefined. The following technical lemma establishes a connection between the domain of \(f\) and the stability problem of UT-systems. We say that a partial recursive function \(h\) is an 
\textit{extension} of a partial recursive function \(f\) if, whenever \(f(i)\) is defined, then so is \(h(i)\) and \(f(i) = h(i)\). (Thus, \(h(i)\) can be defined although \(f(i)\) is not defined.)

\textbf{Lemma 3.1.} Each UT-computable function \(f\) has an extension \(h\) computed by a UT-scheme \(G = (\Sigma, p)\) such that the system \((\Sigma, p, Bb^i)\) is nonstable iff \(h(i)\) is defined.

\textbf{Proof.} We prove the lemma in two steps, showing (i) the function \(g\) satisfying the “if” part, and (ii) its extension \(h\) satisfying both “if” and “only if” part of the statement.

(i) Let \(f\) be a function computed by a UT-scheme \(G_1 = (\Sigma_1, p_1)\), and denote \(B_1, b_1\), resp. \(E_1, e_1\) the input, resp. the output symbols of \(\Sigma_1\). We show that \(f\) has an extension \(g\) computed by a UT-scheme \(G_2 = (\Sigma_2, p_2)\) with the input, resp. output symbols \(B_2, b_2\), resp. \(E_2, e_2\), such that if \(g(i)\) is defined, then the system \((\Sigma_2, p_2, B_2b^i)\) is nonstable.

Let \(\Sigma_2 = \Omega \cup \{B_2, b_2, E_2, e_2, \hat{B}_2, \hat{b}_2, \hat{E}_2, \hat{e}_2\}\), let \(p_2(a) = p_1(a)\) for each \(a \in \Sigma_1 - \{E_1, e_1\}\). Let

\[
\begin{align*}
p_2(B_2) &= p_1(B_1), & p_2(E_1) &= \hat{E}_2, & p_2(\hat{E}_2) &= E_2, \\
p_2(b_2) &= p_1(b_1), & p_2(e_1) &= \hat{e}_2, & p_2(\hat{e}_2) &= e_2
\end{align*}
\]

and \(p_2(x) = x, x \in \{E_2, e_2, \hat{B}_2, \hat{b}_2\}\). Hence if \(B_1b_1^j \Rightarrow_{G_1} E_1e_1^j\), then \(B_2b_2^j \Rightarrow_{G_2} E_2e_2^j\), and, moreover, a complementation step always occurs in the latter derivation due to (1).

(ii) We show that \(g\) has an extension \(h\) computed by a UT-scheme \(G = (\Sigma, p)\) such that the system \((\Sigma, p, Bb^i)\) is nonstable iff \(h(i)\) is defined.

Denote \(\Sigma_2 = \{a_1, \ldots, a_n, \hat{a}_1, \ldots, \hat{a}_n\}\), \(n > 1\). Let \(\Sigma_2' = \{a'_1, \ldots, a'_n, \hat{a}'_1, \ldots, \hat{a}'_n\}\) be a Watson–Crick alphabet disjoint from \(\Sigma\). Let \(\varphi : \Sigma_2 \to \Sigma_2' \cup \Sigma_2'\) be the coding defined by

\[
\varphi(a_i) = a_i, \quad 1 \leq i \leq n,
\]

\[
\varphi(\hat{a}_i) = \hat{a}'_i, \quad 1 \leq i \leq n.
\]
Let
\[ \Sigma = \Sigma_2 \cup \Sigma'_2 \cup \{ B, b, C, C', E, e, \bar{B}, \bar{b}, \bar{C}, \bar{C}', \bar{E}, \bar{e} \}, \]
the three above subsets of \( \Sigma \) being mutually disjoint. Let \( B, b \), resp. \( E, e \) be the input, resp. the output symbols of \( \Sigma \). Define
\[ p(B) = \bar{C}\bar{C}', p(b) = \varphi(p_2(b_2)), \quad p(C) = C, \quad p(\bar{C}') = \bar{C}' \]
and further \( p(\varphi(a)) = \varphi(p_2(a)) \) for each \( a \in \Sigma_2 \). If we compare the derivation sequence \( S(G;B_b) \) with the sequence \( S(G_2;B_2b_2) \) before the transition point \( t \) is reached (observe that \( t \) is the same for both the sequences), the differences are:

(a) in \( S(G;B_b) \) all the pyrimidines from \( \Sigma_2 \) are replaced by their primed versions from \( \Sigma'_2 \),
(b) the strings in \( S(G;B_b) \) have the prefix \( CC' \).

Assume now that a complementary transition occurred in the sequence \( S(G;B_b) \), and focus on the following part of the sequence. Denote \( p^{2n} \) the morphism obtained as \( 2n \) iterations of \( p \), and \( p_2^{2n} \) denotes the \( 2n \) iterations of \( p_2 \). Observe that \( G \) behaves as an ordinary DOL scheme now and hence for each symbol \( a \in \Sigma, a = \Rightarrow^* e^j \) iff \( p^{2n}(a) = e^j \). Define
\[ p(\bar{a}_i) = e^j, \quad j = |p_2^{2n}(\bar{a}_i)|_{E_2}, \quad 1 \leq i \leq n, \]
\[ p(a'_i) = e^j, \quad j = |p_2^{2n}(a_i)|_{E_2}, \quad 1 \leq i \leq n, \]
\[ p(C') = E, \]
\[ p(\bar{C}) = \lambda, \]
\[ p(a) = a \quad \text{for all other } a \in \Sigma. \]

If \( B_2b_2 \Rightarrow^t_{G_2} w_i \Rightarrow^* E_2E \) for some \( i, j \geq 0 \), then clearly \( Bb \Rightarrow^* \bar{C}C\bar{C}'w' \Rightarrow E e^j \), where \( w' \) is obtained from \( w_i \) by replacing all the pyrimidines by their primed versions. Hence \( h(i) = g(i) \) for each \( i \in \text{dom}(g) \).

Moreover, if the system \( (\Sigma, p,Bb) \) is nonstable, on the one hand, then the string \( E e^j \), \( j \geq 0 \) is generated and hence \( h(i) \) is defined. On the other hand, if \( (\Sigma, p,Bb) \) is stable, then \( E \) never appears in the sequence \( S(G;B_b) \) and hence \( h(i) \) is undefined.

The following two theorems give the characterization of the class of UT-computable functions in terms of rational functions. We call a function \( f \), whose domain contains the set of nonnegative integers, to be \( \mathbb{Q} \)-rational, resp. \( \mathbb{Z} \)-rational, resp. \( \mathbb{N} \)-rational, if \( f(0), f(1), f(2), \ldots \) is a \( \mathbb{Q} \)-rational, resp. \( \mathbb{Z} \)-rational, resp. \( \mathbb{N} \)-rational sequence.

**Theorem 3.2.** Each function of the form
\[ f(x) = g_1(t) + xg_2(t), \quad t = \min\{ y \in \mathbb{N} | h_1(y) + xh_2(y) < 0 \}, \]
\(g_1\) and \(g_2\) being \(\mathbb{N}\)-rational functions and \(h_1\), \(h_2\) being \(\mathbb{Z}\)-rational functions, is UT-computable.

**Proof.** It is well known (see [9]) that every \(\mathbb{Z}\)-rational sequence can be expressed as the difference of two DOL growth functions. Let \(h_{11}, h_{12}, h_{21}\) and \(h_{22}\) be DOL growth functions such that

\[
\begin{align*}
    h_{11}(y) &= h_{11}(y) - h_{12}(y), \quad y \geq 0, \\
    h_{21}(y) &= h_{21}(y) - h_{22}(y), \quad y \geq 0.
\end{align*}
\]

The functions \(h_{11}, h_{12}, h_{21}, h_{22}\) can be effectively constructed, see e.g. [14] for details. Denote

\[
\begin{align*}
    (\Sigma_{h_{11}}, p_{h_{11}}, w_{h_{11}}) & \quad \text{the DOL system with the growth function } h_{11}, \\
    (\Sigma_{h_{12}}, p_{h_{12}}, w_{h_{12}}) & \quad \text{the DOL system with the growth function } h_{12}, \\
    (\Sigma_{h_{21}}, p_{h_{21}}, w_{h_{21}}) & \quad \text{the DOL system with the growth function } h_{21}, \\
    (\Sigma_{h_{22}}, p_{h_{22}}, w_{h_{22}}) & \quad \text{the DOL system with the growth function } h_{22}, \\
    (\Sigma_{g_{1}}, p_{g_{1}}, w_{g_{1}}, \varphi_{1}) & \quad \text{the HDOL system with the growth function } g_{1}, \\
    (\Sigma_{g_{2}}, p_{g_{2}}, w_{g_{2}}, \varphi_{2}) & \quad \text{the HDOL system with the growth function } g_{2},
\end{align*}
\]

\(\varphi_{1}\) and \(\varphi_{2}\) being morphisms. Let further \(\bar{\Sigma}'_{g_{1}}\) and \(\bar{\Sigma}'_{g_{2}}\) be alphabets such that there exist bijections \(\phi_{1} : \Sigma_{g_{1}} \rightarrow \bar{\Sigma}'_{g_{1}}\) and \(\phi_{2} : \Sigma_{g_{2}} \rightarrow \bar{\Sigma}'_{g_{2}}\). Consider all the above alphabets to be pairwise disjoint.

Let \(\Sigma = \Sigma^{\text{PUR}} \cup \Sigma^{\text{PYR}}\) be a DNA-like alphabet. We adopt the convention that if \(\Sigma_{x} \subseteq \Sigma^{\text{PUR}}\) is a set of purines, then we denote \(\bar{\Sigma}_{x} \subseteq \Sigma^{\text{PYR}}\) the set of the corresponding pyrimidines, and vice versa, \(x\) being an arbitrary of the above subscripts \(h_{11}, h_{12}, \ldots, g_{2}\).

Let

\[
\begin{align*}
    \Sigma^{\text{PUR}} &= \Sigma_{h_{11}} \cup \Sigma_{h_{12}} \cup \Sigma_{h_{21}} \cup \Sigma_{h_{22}} \cup \Sigma_{g_{1}} \cup \Sigma_{g_{2}} \cup \{B, b, C, E, e\}, \\
    \Sigma^{\text{PYR}} &= \Sigma_{h_{11}} \cup \Sigma_{h_{12}} \cup \Sigma_{h_{21}} \cup \Sigma_{h_{22}} \cup \Sigma_{g_{1}} \cup \Sigma_{g_{2}} \cup \{\bar{B}, \bar{b}, \bar{C}, \bar{E}, \bar{e}\},
\end{align*}
\]

none of these subsets of \(\Sigma\) having common letters. Let \(U = (\Sigma, p)\) be a UT-scheme, where

\[
p \supset (p_{h_{11}} \cup p_{h_{12}} \cup p_{h_{21}} \cup p_{h_{22}} \cup p_{g_{1}} \cup p_{g_{2}}).
\]

Let, moreover, \(p(B) = w_{h_{11}} \bar{w}_{h_{12}} \bar{E} w_{g_{1}} \varphi_{1}(w_{g_{1}}) C\) and \(p(b) = w_{h_{21}} \bar{w}_{h_{22}} w_{g_{2}} \varphi_{2}(w_{g_{2}})\). Hence the string \(p(Bb^{+})\) contains the axioms of all the systems in (4), together with the strings \(\varphi_{1}(w_{g_{1}})\), \(\varphi_{2}(w_{g_{2}})\), as well as the letters \(\bar{E}\) and \(C\). Let further

\[
\begin{align*}
p(C) &= C, \\
p(\bar{E}) &= \bar{E}, \\
p(\varphi_{1}(a)) &= \varphi_{1}(p_{g_{1}}(a)), \quad a \in \Sigma_{g_{1}}, \\
p(\varphi_{2}(a)) &= \varphi_{2}(p_{g_{2}}(a)), \quad a \in \Sigma_{g_{2}}.
\end{align*}
\]

Remind now that the elements of \(\Sigma_{h_{11}}, \Sigma_{h_{21}}, \Sigma_{g_{1}}\) and \(\Sigma_{g_{2}}\) are all purines, and the elements of \(\bar{\Sigma}_{h_{12}}, \bar{\Sigma}_{h_{22}}, \bar{\Sigma}'_{g_{1}}\) and \(\bar{\Sigma}'_{g_{2}}\) are all pyrimidines. Hence we can write for the
members \( s_k \) of the ordinary D0L sequence \( S(\Sigma, p, Bb^*) \) due to (5) and (6),
\[
|s_k|_{\text{PUR}} = h_1(k) + g_1'(k) + xh_2(k) + xg_2'(k) + 1,
|s_k|_{\text{PYR}} = h_2(k) + g_1'(k) + xh_2(k) + xg_2'(k) + 1,
\]
g_1' and g_2' being the growth functions of the D0L systems \((\Sigma_1, p, g_1, w_1)\) and \((\Sigma_2, p, g_2, w_2)\), respectively. Therefore,
\[
|s_k|_{\text{PUR}} - |s_k|_{\text{PYR}} = h_1(k) - h_2(k) + x(h_2(k) - h_2(0)),
\]
and, by (3), we can observe that the transition point of the UT-system \((\Sigma, p, Bb^*)\) is
\[
t(x) = \min\{k \in \mathbb{N} \mid h_1(k) + xh_2(k) < 0\} + 1.
\]

Let us focus now on the behavior of the system at its transition point. Let
\[
p(a) = \lambda, \quad a \in \bar{\Sigma}_1 \cup \Sigma_1 \cup \bar{\Sigma}_2 \cup \Sigma_2 \cup \bar{\Sigma}_2', \cup \Sigma_2',
p(\bar{a}) = e_{\lambda opt}(a), \quad \bar{a} \in \Sigma_1,
p(\bar{a}) = e_{\lambda opt}(a), \quad \bar{a} \in \Sigma_2,
\]
and \( p(a) = a \) for all other \( a \in \Sigma \) such that \( p(a) \) has not been defined yet. Hence due to (7) and (9)
\[
p(s_{t(x)}) = E e^z \quad \text{where } z = g_1(t(x) - 1) + xg_2(t(x) - 1)
\]
and together with (8) we can conclude that
\[
p(Bb^*) \Rightarrow_{U}^* E e^{f(x)}
\]
and hence \( f(x) \) is UT-computable. \( \square \)

**Theorem 3.3.** Each UT-computable function \( g \) has an extension \( f \) which can be expressed in the form (2).

**Proof.** Due to Lemma 3.1 there is an extension \( f \) of \( g \) computed by a uni-transitional scheme \( U = (\Sigma, p) \) such that \( f(x) \) is defined iff the UT-system \((\Sigma, p, Bb^*)\) is nonstable. Let \(|\Sigma| = 2n\) for some \( n \geq 1\). Denote \( (s_k) \) the ordinary D0L sequence \( S(U, Bb^*) \), \( x \in \text{dom}(f) \). Then the sequences \((|s_k|_{\text{PUR}})\) and \((|s_k|_{\text{PYR}})\) are \( \mathbb{N} \)-rational, and hence we can write down
\[
|s_k|_{\text{PUR}} - |s_k|_{\text{PYR}} = h_1(k) + xh_2(k), \quad k \geq 0,
\]
h_1 and h_2 being \( \mathbb{Z} \)-rational functions. The transition point of the system \((\Sigma, p, Bb^*)\) is
\[
t = \min\{k \in \mathbb{N} \mid h_1(k) + xh_2(k) < 0\}.
\]
Once the sequence of the system \((\Sigma, p, Bb^*)\) has passed its transition point, the string \( s_t \) is rewritten to \( E e^{f(x)} \) due to Definition 2.3 and Lemma 3.1. It takes at most \( 2n \)
derivation steps since in each D0L sequence over the alphabet $\Sigma$ with the cardinality $2n$ each letter appears either in the first $2n$ words or never.

Consider the morphism $\varphi = p^{2n}$, then $\varphi(s_i) = Ee^{f(x)}$. Let further $\psi$ be a morphism such that $\psi(E) = \lambda$ and $\psi(e) = e$. Denoting $g_1$, resp. $g_2$ the $\mathbb{N}$-rational growth functions of the HD0L systems $(\Sigma, p, B, \psi \circ \varphi)$, resp. $(\Sigma, p, b, \psi \circ \varphi)$, we can write
\[
f(x) = |\psi(\varphi(s_i))| = g_1(t) + xg_2(t).
\]

In some special cases function (2) adopts a simpler form.

**Corollary 3.4.** Let $g_1, g_2$ be $\mathbb{N}$-rational functions and $h : \mathbb{R}^+ \to \mathbb{R}$ be a strictly monotonic $\mathbb{Q}$-rational function. Then the function $f(x) = g_1(t) + xg_2(t)$, where
\[
t = \begin{cases} 
\lceil h^{-1}(x) \rceil & \text{if } h \text{ is increasing}, \\
\lfloor h^{-1}(x) \rfloor & \text{if } h \text{ is decreasing}, 
\end{cases}
\]
x $\in \mathbb{N} \cap \text{rng}(h)$, is UT-computable.

**Proof.** Due to Lemma II.6.5. in [14] there is an integer $q > 0$ such that the function $s(y) = h(y)q^{k+1}$, $y \geq 0$, is $\mathbb{Z}$-rational. Denoting $h_1(y) = -s(y) - 1$ and $h_2(y) = q^{k+1}$ ($q^{k+1}$ is a D0L growth function and hence it is of course $\mathbb{Z}$-rational), we can convert the right-hand side of (2) to the form
\[
t = \min \left\{ y \in \mathbb{N} \mid x \leq \frac{s(y)}{q^{k+1}} = h(y) \right\}.
\]

Consider that $h$ is increasing, then for $x \in \mathbb{N} \cap \text{rng}(h)$,
\[
t = \min \{ y \in \mathbb{N} \mid y \geq h^{-1}(x) \}
\]
and hence $t = \lceil h^{-1}(x) \rceil$. The situation for a decreasing $h$ is analogous.  

**Corollary 3.5.** Each $\mathbb{N}$-rational function is UT-computable. Each function of the form $f(g^{-1})$, where $f$ is $\mathbb{N}$-rational and $g$ is $\mathbb{Z}$-rational and strictly monotonic, is UT-computable.

It is rather unlikely that the converse of Corollary 3.4 or 3.5 would be possible. We proved Corollary 3.4 using the fact that each $\mathbb{Q}$-rational function can be expressed as a quotient of a $\mathbb{Z}$-rational and an $\mathbb{N}$-rational function. As the following lemma shows, the converse does not hold.

**Lemma 3.6.** Let $(r_k)$ be a $\mathbb{Z}$-rational sequence and $(s_k)$ be an $\mathbb{N}$-rational sequence, $s_k > 0$ for every $k \geq 0$. Then the sequence $(t_k)$, where $t_k = r_k/s_k$, is $\mathbb{Q}$-rational if and only if there is an integer $q > 0$ such that $q^{k+1}t_k$ is an integer for every $k \geq 0$.

**Proof.** (i) Let $(t_k)$ be a $\mathbb{Q}$-rational sequence, then due to Lemma II.6.5. in [14] there is an integer $q > 0$ such that the sequence $(t_kq^{k+1})$ is $\mathbb{Z}$-rational and hence $q^{k+1}r_k/s_k$, $k \geq 0$, is an integer.
(ii) Let \( q^{k+1}r_k/s_k, k \geq 0 \), be an integer. Denote \( (r_k') \) the \( \mathbb{Z} \)-rational sequence \( r_k' = r_kq^{k+1} \). Then \( r_k'/s_k \) is an integer and due to Exercise II.10.4 in [14] the sequence \( (r_k'/s_k) \) is \( \mathbb{Z} \)-rational. Then the sequence \( (t_k) \), \( t_k = (r_k'/s_k)/q^{k+1} \), is \( \mathbb{Q} \)-rational. □

**Note.** There are D0L growth sequences \( (r_k) \) and \( (s_k) \), namely \( r_k = 1, s_k = k + 1, k \geq 0 \), such that the sequence \( (r_k/s_k) \) is not even \( \mathbb{C} \)-rational.

As a consequence of Theorem 3.2, we can establish the following relation between the \( \mathbb{Z}_{\text{pos}} \) problem and the domain of UT-computable functions.

**Theorem 3.7.** For any constant \( c \in \mathbb{N} \), the following problem is algorithmically equivalent to the problem \( \mathbb{Z}_{\text{pos}} \): given a UT-computable function \( f \) (by some effective means such as a UT-scheme or the form (2)), decide whether or not \( c \in \text{dom}(f) \).

**Proof.** (i) Given a \( \mathbb{Z} \)-rational sequence \( (s_k) \), we can construct the UT-computable function \( f \) of the form (2), where

\[
\begin{align*}
    h_1(k) &= s_k, & h_2(k) &= 0, & g_1(k) &= k, & g_2(k) &= 0, & k \geq 0.
\end{align*}
\]

Consider the UT-scheme \( U = (\Sigma, p) \) used in the proof of Theorem 3.2. Due to (8), the UT-system \( (\Sigma, p, Bb^+) \) has a transition point \( t \) iff \( t \) is the minimal integer such that \( s_t < 0 \). Moreover, the scheme \( U \) satisfies the conditions of Lemma 3.1. Hence if \( s_t < 0 \) for some \( t \in \mathbb{N} \), then \( \text{dom}(f) = \mathbb{N} \), else \( \text{dom}(f) = \emptyset \).

(ii) Remind first that the stability problem for standard UT-systems has been shown to be algorithmically equivalent to the problem \( \mathbb{Z}_{\text{pos}} \) in [12].

Let \( f \) be a UT-computable function and \( U = (\Sigma, p) \) a UT-scheme computing \( f \), where \( |\Sigma| = 2n \) for some \( n \geq 1 \). If, on the one hand, we knew that the system \( (\Sigma, p, Bb^+) \) is stable, then it would be enough to search the first \( 2n \) members of the sequence \( S(U, Bb^+) \) to decide whether or not the string of the form \( Ee^x \) appears in the sequence.

If, on the other hand, we knew that the system \( (\Sigma, p, Bb^+) \) is nonstable, then there would exist a transition point \( t \) of the system. Then it would be enough to search for the string \( Ee^x \) within the first \( t+2n \) members of the sequence \( S(U, Bb^+) \). In both cases, we would be able to decide whether \( f(c) \) is defined or not. □

**Corollary 3.8.** An algorithm solving any of the following problems can be converted to an algorithm solving \( \mathbb{Z}_{\text{pos}} \): given a UT-computable function \( f \), decide whether or not:

(i) \( \text{dom}(f) = \emptyset \),
(ii) \( \text{dom}(f) = \mathbb{N} \), i.e. whether or not \( f \) is total,
(iii) \( \text{dom}(f) \) is finite.

### 4. Growth of UT-computable functions

In this section we study properties of UT-computable functions, namely the possible limitations of their growth. We show first that each UT-computable function with an infinite range grows at least as fast as a logarithmic function.
Theorem 4.1. Let \( f \) be a UT-computable function with an infinite range. Then there are (effectively constructible) constants \( c_1 \geq 0, c_2 > 0 \) such that for each \( x \in \text{dom}(f) \),
\[
\text{either } f(x) \leq c_1, \text{ or } f(x) \geq c_2 \log_2(x + 1) - 1.
\]

Proof. Let \( G = (\Sigma, g) \) be a UT-scheme computing \( f \), where \( |\Sigma| = 2n, n \geq 1 \). Denote by \( t(x) \) the transition point of the system \((\Sigma, g, B^t)\). Denoting further (as in the proofs above) \( d(x,k) = |s_k|_{2^n, r} - |s_k|_{2^n, l} \), we can write
\[
t(x) = \min\{k \geq 1 \mid d(x,k) < 0\}.
\]

Part (i) of the proof of Lemma 3.1 guarantees that we can assume the existence of such a \( k \) for an arbitrary \( x \in \text{dom}(f) \). Consider the morphism \( g = g_{2^n} \), i.e., \( 2^n \) iterations of \( g \). Then \( |\varphi(s_{t(x)})| = f(x) + 1 \) for each \( x \geq 2 \) due to Definition 2.3 and Lemma 3.1. Now consider the ordinary HD0L systems \((\Sigma, g, B^r)\), resp. \((\Sigma, g, b, \varphi)\), and their growth functions \( q \), resp. \( r \). It follows that
\[
|\varphi(s_k)| = q(k) + xr(k), \quad 0 \leq k \leq t(x).
\]
Hence
\[
f(x) + 1 = q(t(x)) + xr(t(x)), \quad x \in \text{dom}(f).
\]

Observe that \( q \) and \( r \) are \( \mathbb{N} \)-rational functions and hence can be decomposed into D0L functions (see [14, Lemma III.7.4]). It is also known (see [9]) that each D0L growth function either has a finite range or grows at least linearly. Denote
\[
\mathcal{P}_1 = \text{rng}(f_1) \cup \text{rng}(f_2) \cup \cdots \cup \text{rng}(f_{j_1}),
\]
\[
\mathcal{P}_2 = \text{rng}(g_1) \cup \text{rng}(g_2) \cup \cdots \cup \text{rng}(g_{j_2}),
\]
where \( f_1, \ldots, f_{j_1} \) are all the D0L growth functions with a finite range merging to \( q \) or \( r \), and \( g_1, \ldots, g_{j_2} \) are all the other D0L growth functions merging to \( q \) or \( r \). Then there are positive constants \( d_1, \ldots, d_{j_2} \) such that
\[
g_i(k) \geq d_i k \quad \text{for all } k \geq 0, \quad 1 \leq i \leq j_2.
\]
Denote
\[
c = \begin{cases} 
\min\{d_1, \ldots, d_{j_2}\} & \text{if } j_2 > 0, \\
1 & \text{otherwise},
\end{cases}
\]
\[
p_{\text{min}} = \begin{cases} 
\text{minimal nonzero element of } \mathcal{P}_1 & \text{if one exists}, \\
1 & \text{otherwise},
\end{cases}
\]
\[
p_{\text{max}} = \begin{cases} 
\text{maximal element of } \mathcal{P}_1 & \text{if } \mathcal{P}_1 \neq \emptyset, \\
0 & \text{otherwise}.
\end{cases}
\]
Denote further \( m = 2na_{\text{max}} \), where \( a_{\text{max}} \) is the maximal element of the growth matrix associated to \( G \). (We can assume that \( a_{\text{max}} \geq 1 \); otherwise \( \text{dom}(f) = \emptyset \).) Now we are ready to choose the values \( c_1 \) and \( c_2 \). Let
\[
c_1 = p_{\text{max}},
\]
\[
c_2 = 2^{2n} \log_2(m) - 1.
\]
Then we distinguish the following cases:

- (i) If $r(T) = 0$, then $f$ has a finite range since $f(x) = q(T) - 1$ for all $x \geq x_0$.
- (ii) If $r(T) > 0$, then:
  - (a) $r(T) \in P_1$, then $r(T) \geq p_{\text{min}}$ and due to (17),
    \[ f(x) \geq xr(T) - 1 \geq xp_{\text{min}} - 1 \geq c_2 \log_2(x^2 + 1) - 1. \]
  - (b) $r(T) \in P_2$, then due to (15) $r(T) \geq cT \geq c$ since $T \geq 1$, and thus
    \[ f(x) \geq xr(T) - 1 \geq xc - 1 \geq xc_2 - 1 \geq c_2 \log_2(x^2 + 1) - 1. \]

Consider now the remaining possibility

\[ t(x) \geq \log_2(x + 1), \quad x \in \text{dom}(f). \]  \hspace{1cm} (19)

Then we distinguish the following cases:

- (iii) Both $q(t(x)) \in P_1$ and $r(t(x)) \in P_1$. Then,
  - (a) $r(t(x)) = 0$ implies $f(x) = q(t(x)) - 1 \leq c_1$,
  - (b) $r(t(x)) > 0$ implies $r(t(x)) \geq p_{\text{min}}$, and due to (17) we have
    \[ f(x) \geq xr(t(x)) - 1 \geq c_2 x - 1 \geq c_2 \log_2(x^2 + 1) - 1. \]

- (iv) If $q(t(x)) \in P_2$, then, independently of $r(t(x))$ due to (14), (17) and (19),
  \[ f(x) \geq q(t(x)) - 1 \geq ct(x) - 1 \geq c \log_2(x + 1) - 1 \geq c_2 \log_2(x + 1) - 1. \]

\textbf{Corollary 4.2.} The function $f(x) = \lfloor \log_2 \log_2 x \rfloor$ is not UT-computable.

We established a lower bound for the growth of UT-computable functions with an infinite range in the above theorem. The following result shows that finding an effective
upper bound for the growth of UT-computable functions would imply the decidability of the $\mathbb{Z}_{\text{pos}}$ problem.

**Growth bound problem.** Given a UT-scheme computing the function $f$, find a total recursive function $g : \mathbb{N} \to \mathbb{N}$ and an integer $x_0 \geq 0$ such that for each $x \in \text{dom}(f)$,

$$x \geq x_0 \implies f(x) \leq g(x). \quad (20)$$

**Theorem 4.3.** An algorithm solving the growth bound problem can be converted to an algorithm solving $\mathbb{Z}_{\text{pos}}$.

**Proof.** Given a $\mathbb{Z}$-rational sequence $(s_k)$, construct the UT-computable function $f$ as in (2) and (10). It follows from (2) that if $s_k \geq 0$ for every $k \in \mathbb{N}$, then $\text{dom}(f) = \emptyset$, else $f$ is a constant function such that

$$f(x) = t, \quad t = \min \{k \in \mathbb{N} \mid s_k < 0\} \quad \text{for every } x \geq \mathbb{N}.$$

Here $t$ is the transition point of the UT-system computing $f(x)$ due to (8). Assume that we can find a total function $g$ and a number $x_0$ satisfying (20). If $s_k < 0$ for some $k \geq 0$, then $x_0 \in \text{dom}(f)$, and $f(x_0) \leq g(x_0)$ and, hence, $t \leq g(x_0)$. We can conclude that a negative value appears in the sequence $(s_k)$ iff such a value exists among the first $g(x_0)$ elements of the sequence. \qed

**Acknowledgements**

Petr Sosík is grateful to Juhani Karhumäki for allowing his research stay at the University of Turku, and to Alicia Kelemenová and Jozef Kelemen for continuous support of his work. Research of Petr Sosík was partially supported by the Grant Agency of Czech Republic, Grant No. 201/02/P079, and by the Project No. 55/2001 of the Support Programme of Lifetime Education of the Ministry of Education, Czech Republic.

**References**


