Nondeterministic automatic complexity of overlap-free and almost square-free words

Kayleigh K. Hyde
Schmid College of Science & Technology
Chapman University
Orange, California, U.S.A.
khyde@chapman.edu

Bjørn Kjos-Hanssen*
Department of Mathematics
University of Hawai‘i at Mānoa
Honolulu, Hawai‘i, U.S.A.
bjoernkh@hawaii.edu

Submitted: Nov 25, 2014; Accepted: Jul 30, 2015; Published: Aug 14, 2015
Mathematics Subject Classifications: 68R15, 68Q30

Abstract

Shallit and Wang studied deterministic automatic complexity of words. They showed that the automatic Hausdorff dimension $I(\mathbf{t})$ of the infinite Thue word satisfies $1/3 \leqslant I(\mathbf{t}) \leqslant 1/2$. We improve that result by showing that $I(\mathbf{t}) = 1/2$. We prove that the nondeterministic automatic complexity $A_N(x)$ of a word x of length n is bounded by $b(n) := \lfloor n/2 \rfloor + 1$. This enables us to define the complexity deficiency $D(x) = b(n) - A_N(x)$. If x is square-free then $D(x) = 0$. If x is almost square-free in the sense of Fraenkel and Simpson, or if x is a overlap-free binary word such as the infinite Thue–Morse word, then $D(x) \leqslant 1$. On the other hand, there is no constant upper bound on D for overlap-free words over a ternary alphabet, nor for cube-free words over a binary alphabet.

The decision problem whether $D(x) \geqslant d$ for given x, d belongs to $\mathrm{NP} \cap \mathrm{E}$.

1 Introduction

The Kolmogorov complexity of a finite word w is, roughly speaking, the length of the shortest description w^* of w in a fixed formal language. The description w^* can be thought of as an optimally compressed version of w. Motivated by the non-computability of Kolmogorov complexity, Shallit and Wang [9] studied a deterministic finite automaton analogue. A more recent approach is due to Calude, Salomaa, and Roblot [1].

Definition 1 (Shallit and Wang [9]). The *automatic complexity* of a finite binary string $x = x_1 \cdots x_n$ is the least number $A_D(x)$ of states of a deterministic finite automaton M such that x is the only string of length n in the language accepted by M.

*This work was partially supported by a grant from the Simons Foundation (#315188 to Bjørn Kjos-Hanssen).

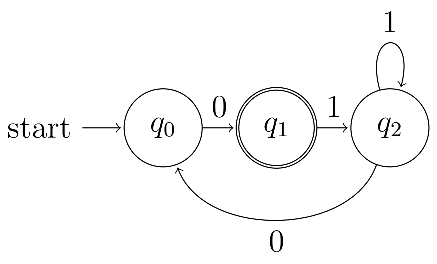


Figure 1: A witnessing automaton for the inequality $A_D(011100) \leqslant 4$. All missing transitions go to a dead state q_3 which is not shown.

This complexity notion has the following two properties:

1. Most of the relevant automata end up having a "dead state" whose sole purpose is to absorb any irrelevant or unacceptable transitions.

2. The complexity of a string can be changed by reversing it. For instance,

$$A_D(011100) = 4 < 5 = A_D(001110). \tag{1}$$

   Equation 1 was verified by a computer program; for the idea and a partial proof see Figure 1. The anonymous referee of this article raised the question, which we have not been able to answer, whether the complexity of a string and its reverse can be arbitrarily far apart.

If we replace deterministic finite automata by nondeterministic ones, these properties disappear. The nondeterministic automatic complexity turns out to have other pleasant properties, such as a sharp linear upper bound.

Technical ideas and results. In this paper we develop some of the properties of nondeterministic automatic complexity. As a corollary we get a strengthening of a result of Shallit and Wang [9] on the complexity of the infinite Thue–Morse word $\mathbf{t}$. Moreover, viewed through an NFA lens we can, in a sense, characterize the complexity of $\mathbf{t}$ exactly. A main technical idea is to extend [9, Theorem 9] which said that not only do squares, cubes and higher powers of a word have low complexity, but a word completely free of such powers must conversely have high complexity. The way we strengthen their results is by considering a variation on square-freeness and cube-freeness, *overlap-freeness.* This notion also goes by the names of *irreducibility* and *strong cube-freeness* in the combinatorial literature. We also take up an idea from [9, Theorem 8] and use it to show that the natural decision problem associated with nondeterministic automatic complexity is in E = DTIME($2^{O(n)}$). This result is a theoretical complement to the practical fact that the nondeterministic automatic complexity can be computed reasonably quickly; to see it in action, for strings of length up to 23 one can view automaton witnesses and check complexity using the following URL format http://math.hawaii.edu/wordpress/bjoern/complexity-of-110101101/

and check one's comprehension by playing a Complexity Guessing Game at

http://math.hawaii.edu/wordpress/bjoern/software/web/complexity-guessing-game/

Let us now define our central notion and get started on developing its properties. Recall that a nondeterministic finite automaton (NFA) is assumed to have no ε-transitions, i.e., it is not an NFA $-\,\varepsilon$.

Definition 2. The nondeterministic automatic complexity $A_N(w)$ of a word w is the minimum number of states of an NFA M accepting w such that there is only one accepting path in M of length $|w|$.

The minimum complexity $A_N(w) = 1$ is only achieved by words of the form a^n where a is a single letter.

Theorem 3 (Hyde [5]). *The nondeterministic automatic complexity $A_N(x)$ of a string x of length n satisfies*

$$A_N(x) \leqslant b(n) := \lfloor n/2 \rfloor + 1.$$

Proof sketch. If x has odd length, it suffices to carefully consider the automaton in Figure 2. If x has even length, a slightly modified automaton can be used. □

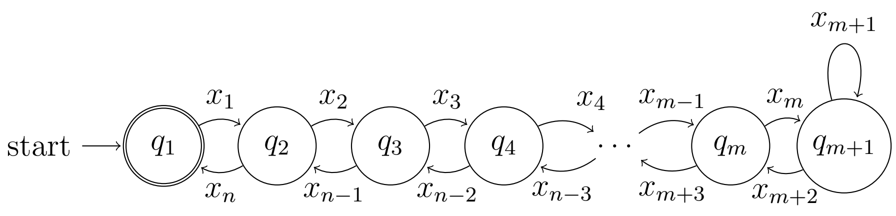


Figure 2: A nondeterministic finite automaton that only accepts one string $x = x_1x_2x_3x_4\cdots x_n$ of length $n = 2m + 1$.

Definition 4. The *complexity deficiency* of a word x of length n is

$$D_n(x) = D(x) = b(n) - A_N(x).$$

The distribution of $A_N(w)$ for w of length $n \leqslant 23$ is given in Table 1. The notion of deficiency is motivated by the experimental observation that about half of all strings have deficiency 0.

2 Time complexity

Definition 5. Let DEFICIENCY be the following decision problem.

Given a binary word w and an integer $d \geqslant 0$, is $D(w) > d$?

$n\backslash k$	1	2	3	4	5	6	7	8	9	10	11	12
23	2	6	20	58	164	430	2540	14252	80962	442278	2160662	5687234
22	2	6	20	58	164	502	2846	16024	94732	451368	2089418	1539164
21	2	6	20	58	176	496	3168	18720	108042	504794	1461670	
20	2	6	20	58	164	430	3814	23328	115896	529148	375710	
19	2	6	20	58	164	582	4996	26542	140668	351250		
18	2	6	20	58	188	598	5692	29990	136024	89566		
17	2	6	20	58	200	514	7102	37042	86128			
16	2	6	20	58	164	752	7738	34320	22476			
15	2	6	20	58	226	908	8530	23018				
14	2	6	20	58	244	1270	9668	5116				
13	2	6	20	64	250	2076	5774					
12	2	6	20	58	282	2090	1638					
11	2	6	20	58	564	1398						
10	2	6	20	64	588	344						
9	2	6	20	78	406							
8	2	6	20	130	98							
7	2	6	22	98								
6	2	6	26	30								
5	2	6	24									
4	2	6	8									
3	2	6										
2	2	2										
1	2											
0	1											

Table 1: The number of strings of length $0 \leqslant n \leqslant 23$ having nondeterministic automatic complexity k.

2.1 NP

Theorem 6 is not surprising; we do not know whether DEFICIENCY is NP-complete.

Theorem 6. DEFICIENCY *is in* NP.

Proof. Shallit and Wang [9, Theorem 2] showed that one can efficiently determine whether a given DFA uniquely accepts w among string of length $|w|$. Hyde [5, Theorem 2.2] extended that result to NFAs, from which the result easily follows. □

2.2 E

Definition 7. Suppose M is an NFA with q states that uniquely accepts a word x of length n. Throughout this paper we may assume that M contains no edges except those traversed on input x. Consider the *almost unlabeled transition diagram* of M, which is a directed graph whose vertices are the states of M and whose edges correspond to transitions. Each edge is labeled with a 0 except for an edge entering the initial state as described below.

We define the *accepting path* P for x to be the sequence of $n+1$ edges traversed in this graph, where we include as first element an edge labeled with the empty string ε that enters the initial state q_0 of M.

We define the *abbreviated accepting path* P' to be the sequence of edges obtained from P by considering each edge in order and deleting it if it has previously been traversed.

Lemma 8. *Let v be a vertex visited by an abbreviated accepting path $P' = (e_0, \dots, e_t)$. Then v is of one of the following five types.*

1. *In-degree 1 (edge e_i), out-degree 1 (edge e_{i+1}).*
2. *In-degree 2 (edges e_i and e_j with $j > i$), out-degree 1 (e_{i+1}).*
3. *In-degree 1 (edge e_i), out-degree 2 (edges e_{i+1} and e_j, $j > i+1$).*
4. *In-degree 2 (edges e_i and e_j with $j > i$), out-degree 2 (e_{i+1} and e_{j+1}).*
5. *In-degree 1 (edge e_t), out-degree 0.*[1]

Proof. The out-degree and in-degree of each vertex encountered along P' are both $\leqslant 2$, since failure of this would imply non-uniqueness of accepting path. Since all the edges of M are included in P, the list includes all the possible in-degree, out-degree combinations. We can define i by the rule that e_i is the first edge in P' entering v. Again, since all the edges of M are included in P, e_{i+1} must be one of the edges contributing to the out-degree of v, if any, and e_j must also be as specified in the types. □

Lemma 8 implies that Definition 9 makes sense.

[1]This type was omitted by Shallit and Wang.

Definition 9. For $0 \leqslant i \leqslant t+1$ and $0 \leqslant n \leqslant t+1$ we let $E(i,n)$ be a string representing the edges $(e_i, \cdots, e_n)$. The meaning of the symbols is as follows: 0 represents an edge. A left bracket [represents a vertex that is the target of a backedge. A right bracket] represents a backedge. The symbol + represents a vertex of out-degree 2. When $i > n$, we set $E(i,n) = \varepsilon$. Next, assuming we have defined $E(j,m)$ for all m and all $j > i$, we can define $E(i,n)$ by considering the type of the vertex reached by the edge e_i. Let $a_i \in \{0, \varepsilon\}$ be the label of e_i.

1. $E(i,n) := a_i E(i+1,n)$.
2. $E(i,n) := a_i[E(i+1,j-1)]E(j+1,n)$.
3. $E(i,n) := a_i + E(i+1,n)$.
4. $E(i,n) := a_i[+E(i+1,j-1)]E(j+1,n)$.
5. $E(i,n) := a_i E(i+1,n)$.

Lemma 10. *The abbreviated accepting path can be reconstructed from* $E(0,t)$.

We do not include the proof of Lemma 10; instead, Figure 3 gives an example of an automaton and the computation of $E(0,t)$.

Lemma 11.
$$|E(a,b)| \leqslant 2(b-a+1).$$

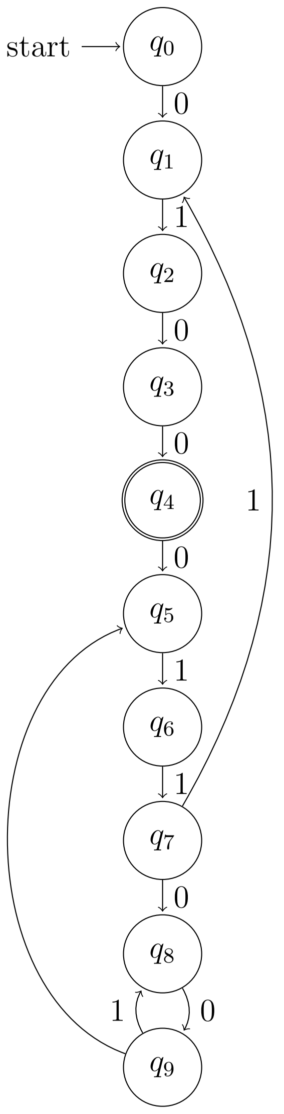
Proof of Lemma 11. The four rules are

1. $E(i,n) = a_i E(i+1,n)$
2. $E(i,n) = a_i[E(i+1,j-1)]_{a_j} E(j+1,n)$
3. $E(i,n) = a_i + E(i+1,n)$
4. $E(i,n) = a_i[+E(i+1,j-1)]_{a_j} E(j+1,n)$

So either
$$|E(i,n)| \leqslant 2 + |E(i+1,n)|$$
or
$$|E(i,n)| \leqslant 4 + |E(i+1,j-1)| + |E(j+1,n)|.$$
So if by induction hypothesis $|E(a,b)| \leqslant 2(b-a+1)$ then
$$|E(i,n)| \leqslant 2 + 2(n-i-1+1) = 2(n-i+1)$$
or
$$|E(i,n)| \leqslant 4 + 2(j-1-i-1+1) + 2(n-j-1+1) = 2(n-i+1).$$
□

$E(i,n)$	Computation
$E(0,12)$	$\varepsilon E(1,12) = E(1,12)$
$E(1,12)$	$a_1[E(2,11)]_{a_{12}}E(13,12)$
$E(13,12)$	ε
$E(2,11)$	$a_2E(3,11)$
$E(3,11)$	$a_3E(4,11)$
$E(4,11)$	$a_4E(5,11)$
$E(5,11)$	$a_5[E(6,10)]_{a_{11}}E(12,11)$
$E(6,10)$	$a_6E(7,10)$
$E(7,10)$	$a_7 + E(8,10)$
$E(8,10)$	$a_8[E(9,9)]_{a_{10}}E(11,10)$
$E(9,9)$	$a_9 + E(10,9) = a_9+$
$E(8,10)$	$a_8[a_9+]_{a_{10}}$
$E(7,10)$	$a_7 + a_8[a_9+]_{a_{10}}$
$E(6,10)$	$a_6a_7 + a_8[a_9+]_{a_{10}}$
$E(5,11)$	$a_5[a_6a_7 + a_8[a_9+]_{a_{10}}]_{a_{11}}$

(a) The + marks the place of a loopback.



(b) Complexity witness for the string 0100011001010101111100, one of the 2,655,140 simple strings of length $n = 22$.

Figure 3: The code is $E(0,12) = a_1[a_2a_3a_4a_5[a_6a_7 + a_8[a_9+]_{a_{10}}]_{a_{11}}]_{a_{12}}$ where $(a_1,\ldots,a_{12}) = (0,1,0,0,0,1,1,0,0,1,1,1)$. In reduced form, $E(0,12) = 0[0000[00 + 0[0+]]]$.

Theorem 12. DEFICIENCY *is in E.*

Proof. Let w be a word of a length n, and let $d \geqslant 0$. To determine whether $D(w) > d$, we must determine whether there exists an NFA M with at most $\lfloor \frac{n}{2} \rfloor - d$ states which accepts w, and accepts no other word of length n. Since there are *prima facie* more than single-exponentially many automata to consider, we consider instead codes $E(0,t)$ as in Definition 9. By Lemma 10 we can recover the abbreviated accepting path P' and hence M from such a code. The number of edges t is bounded by the string length n, so by Lemma 11

$$|E(0,t)| \leqslant 2(t+1) \leqslant 2(n+1);$$

since there are four symbols this gives

$$4^{2(n+1)} = O(16^n)$$

codes to consider. Finally, to check whether a given M accepts uniquely takes only polynomially many steps, as in Theorem 6. □

Remark 13. The bound 16^n counts many automata that are not uniquely accepting; the actual number may be closer to 3^n based on computational evidence.

3 Powers and complexity

In this section we shall exhibit infinite words all of whose prefixes have complexity deficiency bounded by 1. We say that such a word has a hereditary deficiency bound of 1.

3.1 Square-free words

Lemma 14. *Let x and y be strings over an arbitrary alphabet with $xy = yx$. Then there is a string z and integers k and ℓ such that $x = z^k$ and $y = z^\ell$.*

Lemma 14 is proved in Shallit [8, Theorem 2.3.3] and is originally due to Lyndon and Schuetzenberger [6].

Definition 15. A word x is a *factor* in a word y if $y = uxv$ for some words u and v. In this case we also say that *y contains x.*

We will use the following simple strengthening from DFAs to NFAs of a fact used in [9, Theorem 9].

Theorem 16. *If an NFA M uniquely accepts w of length n, and visits a state p at least $k+1$ times, where $k \geqslant 2$, during its computation on input w, then w contains a kth power.*

Proof. Let $w = w_0 w_1 \cdots w_k w_{k+1}$ where

- w_0 is the portion of w read before the first visit to the state p,

- w_i is the portion of w read between visits number i and $i+1$ to the state p for $1 \leqslant i \leqslant k$, and

- w_{k+1} is the portion of w read after the last visit to the state p.

Thus $|w_i| \geqslant 1$ for each $1 \leqslant i \leqslant k$, but it is possible to have $|w_0| = 0$ ($|w_{k+1}| = 0$) since the initial (final) state of M's on input w computation may be p.

For any permutation π on $1, \ldots, k$, M accepts $w_0 w_{\pi(1)} \cdots w_{\pi(k)} w_{k+1}$. Let $1 \leqslant j \leqslant k$ be such that w_j has minimal length and let

$$\hat{w}_j = w_1 \cdots w_{j-1} w_{j+1} \cdots w_k.$$

Then M also accepts

$$w_0 w_j \hat{w}_j w_{k+1} \quad \text{and} \quad w_0 \hat{w}_j w_j w_{k+1}.$$

By uniqueness,

$$w_0 w_j \hat{w}_j w_{k+1} = w = w_0 \hat{w}_j w_j w_{k+1}$$

and so

$$w_j \hat{w}_j = \hat{w}_j w_j.$$

By Lemma 14, w_j and $\hat{w}_j$ are both powers of a string z. Since $|\hat{w}_j| \geqslant (k-1)|w_j|$, $w_j \hat{w}_j$ is at least a kth power of z, so w contains a kth power of z. □

Theorem 17 (Extended Pigeonhole Principle)**.** *If $aq + d$ pigeons are placed in q pigeonholes where $d > 0$, then it cannot be the case that all pigeonholes have at most a pigeons; in fact, either*

- *there is a pigeonhole with at least $a + d$ pigeons; or*

- *there is a pigeonhole with at least $a + d - 1$ pigeons, and another with $a + 1$ pigeons; or*

- *there is a pigeonhole with at least $a + d - 2$ pigeons, and another with $a + 2$ pigeons; or*

- *there is a pigeonhole with at least $a + d - 2$ pigeons, and two others with $a + 1$ pigeons; or*

- *all pigeonholes have at most $a + d - 3$ pigeons (which is impossible if $a + d - 3 \leqslant a$ and $d > 0$).*

Proof. Consider the maximum number of pigeons in a pigeonhole m. If $m \geqslant a + d$ we are in Case 1. If $m = a + d - 1$, we consider all the other pigeons and pigeonholes; there are then $q - 1$ pigeonholes and $aq + d - (a + d - 1) = a(q-1) + 1$ pigeons. By the plain Pigeonhole Principle, there is a pigeonhole with at least $a + 1$ pigeons. If $m = a + d - 2$, we repeat the argument, consider the maximum number of pigeons in a pigeonhole other than a given one with the maximum number of pigeons. □

We next strengthen a particular case of [9, Theorem 9] to NFAs.

Theorem 18. *A square-free word has deficiency 0.*

Proof. Suppose w is a word of length $n = 2k$ or $n = 2k + 1$, of deficiency d. Then there is a witnessing automaton M with $q = k + 1 - d$ states. Since $n + 1 \geqslant 2k + 1 = 2(k + 1 - d) + 2d - 1 = 2q + (2d - 1)$, by the Extended Pigeonhole Principle (Theorem 17), there is a state p which is visited $2 + (2d - 1) = 3$ times $t_1 < t_2 < t_3$, during the $n+1$ times of the computation of M on input w (and is not visited at any other times in the interval $[t_1, t_3]$). By Theorem 16, w contains a square. □

Corollary 19. *There exists an infinite word of hereditary deficiency 0.*

Proof. There is an infinite square-free word over the alphabet $\{0, 1, 2\}$ as shown by Thue [11]. The result follows from Theorem 18. □

3.2 Cube-free and overlap-free words

Definition 20. For a word u, let $\text{first}(u)$ and $\text{last}(u)$ denote the first and last letters of u, respectively. An *overlap* is a word of the form $uu\,\text{first}(u)$ (or equivalently, $\text{last}(u)uu$). A word w is *overlap-free* if it does not contain any overlaps.

Definition 21 (Thue-Morse morphism)**.** Let $\{0, 1\}^*$ denote the set of all finite binary words. A *morphism* is a function $\nu : \{0, 1\}^* \to \{0, 1\}^*$ which is a homomorphism with respect to concatenation, in the sense that

$$\nu(xy) = \nu(x)\nu(y)$$

for all $x, y \in \{0, 1\}^*$. The *Thue-Morse morphism* is the unique morphism μ satisfying

$$\mu(0) = 01, \qquad \mu(1) = 10.$$

Theorem 22 (Shelton and Soni [10])**.** *Let $a \geqslant 0$. The words $\mu^a(00)$ and $\mu^a(001001)$ are overlap-free squares of lengths 2^{a+1}, $3 \cdot 2^{a+1}$, respectively*[2]*.*

Example 23 (Examples of Theorem 22.)**.** The following overlap-free squares exemplify the first few possible lengths, 2, 4, 6, 8 and 12:

$$00, \quad \mu(00) = 0101, \quad 001001, \quad \mu^2(00) = 01100110,$$

$$\mu(001001) = 010110010110.$$

Theorem 22 is used in the proof of the following result.

Theorem 24 (Shelton and Soni [10])**.** *Let ℓ be a positive integer. The following are equivalent.*

[2] There is a minor typo in Shelton and Soni's paper (line 10 of page 98), equivalent to writing $\mu^a(001)$ instead of $\mu^a(001001)$.

1. *There exists an overlap-free binary word y and a word x such that y contains xx and $\ell = |xx|$.*

2. $\ell \in \{2^a : a \geqslant 1\} \cup \{3 \cdot 2^a : a \geqslant 1\}$.

Lemma 25. *If a cube www contains another cube xxx then either $|x| = |w|$, or $xx\,\mathrm{first}(x)$ is contained in the first two consecutive occurrences of w, or $\mathrm{last}(x)xx$ is contained in the last two occurrences of w.*

Proof. We prove the contrapositive. Suppose $xx\,\mathrm{first}(x)$ is not contained in the first two consecutive occurrences of w, and $\mathrm{last}(x)xx$ is not contained in the last two occurrences of w. Then the middle $\mathrm{last}(x)x\,\mathrm{first}(x)$ of the factor xxx has $\mathrm{last}(w)w\,\mathrm{first}(w)$ as a factor, and hence $|x| \geqslant |w|$. □

Theorem 26. *The deficiency of cube-free binary words is unbounded.*

Proof. Given k, we shall find a cube-free word x with $D(x) \geqslant k$. Pick a number n such that $2^n \geqslant 2k + 1$. Let $w := \mu^n(0)$, which is a word of length $\ell := 2^n$. By Theorem 22, ww is overlap-free. Let $x = ww\hat{w}$ where $\hat{w}$ is the proper prefix of w of length $|w| - 1$. By Lemma 25, x is cube-free. The complexity of x is at most $|w|$ as we can just make one loop of length w, with code (Theorem 12)

$$[w_1 \cdots w_{\ell-1}]_{w_\ell}.$$

And so

$$\begin{aligned} D(x) &\geqslant \left\lfloor \frac{|x|}{2} \right\rfloor + 1 - |w| \geqslant \frac{|x|}{2} - |w| \\ &= \frac{3|w| - 1}{2} - |w| = \frac{|w|}{2} - \frac{1}{2} \geqslant k. \end{aligned}$$ □

3.3 Overlap-free words

Theorem 27 (Thue [11])**.** *The infinite Thue–Morse word*

$$\mathbf{t} = t_0 t_1 \cdots = 0110\,1001\,1001\,0110 \cdots$$

given by

$$b = \sum b_i 2^i, \quad b_i \in \{0, 1\} \quad \Longrightarrow \quad t_b = \sum b_i \mod 2,$$

is overlap-free.

Lemma 28. *Fix j and k and let t_x denote the xth bit of the Thue–Morse word. The function*

$$f(u) = t_{x(u)-1} \quad \text{where} \quad x(u) = 3^{k-j}(3u + 2)$$

is eventually nonconstant.

Proof. Gelfond [4] showed that $\mathbf{t}$ has no infinite arithmetic progressions (see also Morgenbesser, Shallit, Stoll [7]). □

Lemma 29. *For each $k \geqslant 1$ there is a sequence $x_{1,k}, \ldots, x_{k,k}$ of positive integers such that*

$$\sum_{i=1}^{k} a_i x_{i,k} = 2\sum_{i=1}^{k} x_{i,k} \quad \Longrightarrow \quad a_1 = \cdots = a_k = 2.$$

Let t_j denote bit j of the infinite Thue–Morse word. Then we can ensure that

1. *$x_{i,k} + 1 < x_{i+1,k}$ and*
2. *$t_{x_{i,k}} \neq t_{x_{i+1,k}}$ for each $1 \leqslant i < k$.*

Proof. Let

$$x_{1,1} = 1.$$

Given $x_{1,k-1}, \ldots, x_{k-1,k-1}$, we let $x_{i,k} = 3x_{i,k-1}$ for $i < k$ and $x_{k,k} = 3u_{k-1} + 2$ for a sufficienctly large number u_{k-1}. Reducing the equation

$$\sum_{i=1}^{k} a_i x_{i,k} = 2\sum_{i=1}^{k} x_{i,k}$$

modulo 3, we see that $a_k \equiv 2 \pmod 3$. If $a_k \geqslant 5$ then

$$\sum_i a_i x_{i,k} \geqslant 5x_{k,k} = 15u_{k-1} + 10$$

$$> 6\sum_{i<k} x_{i,k-1} + 6u_{k-1} + 4 = 2\sum_{i<k} x_{i,k} + 2(3u_{k-1} + 2) = 2\sum_{i\leqslant k} x_{i,k};$$

provided

$$3u_{k-1} + 2 > 2\sum_{i<k} x_{i,k-1}$$

so we conclude $a_k = 2$. Then we can cancel a_k, divide by three and reduce to the induction hypothesis.

Thus our numbers are

$$x_{1,2} = 3, \quad x_{2,2} = 3u_1 + 2,$$

$$x_{1,3} = 3^2, \quad x_{2,3} = 3(3u_1 + 2), \quad x_{3,3} = 3u_2 + 2$$

and in general

$$x_{j,k} = 3^{k-j}(3u_{j-1} + 2)$$

To ensure (1) we just take u_{j-1} sufficiently big. To ensure (2), we apply Lemma 28. □

Theorem 30. *The complexity deficiency of overlap-free words over an alphabet of size three is unbounded.*

Proof. Let $d \geqslant 1$. We will show that there is a word w of deficiency $D(w) \geqslant d$. Let $k = 2d-1$. For each $1 \leqslant i \leqslant k$ let $x_i = x_{k+1-i,k}$ where the $x_{j,k}$ are as in Lemma 29. Note that since $x_{i,k}+1 < x_{i+1,k}$, we have $x_i > x_{i+1}+1$. Let

$$\begin{aligned} w &= \left(2\prod_{i=1}^{x_1-1} t_i\right)^2 t_{x_1} \left(2\prod_{i=1}^{x_2-1} t_i\right)^2 t_{x_2} \left(2\prod_{i=1}^{x_3-1} t_i\right)^2 \cdots t_{x_{k-1}} \left(2\prod_{i=1}^{x_k-1} t_i\right)^2 \\ &= \lambda_1 t_{x_1} \lambda_2 \cdots t_{x_{k-1}} \lambda_k \end{aligned}$$

where $\lambda_i = (2\tau_i)^2$, $\tau_i = \prod_{j=1}^{x_i-1} t_j$, and where t_i is the ith bit of the infinite Thue–Morse word on $\{0,1\}$, which is overlap-free (Theorem 27). Let M be the NFA with code (Theorem 12)

$$[+0^{x_1-1}]0[+0^{x_2-1}]0\cdots 0 * [+0^{x_k-1}],$$

where $*$ indicates the accept state. Let $X = \sum_{i=1}^{k} x_i$. Then M has $k-1+X$ edges but only $q = X$ states; and w has length

$$n = k - 1 + 2X = 2(d-1) + 2X,$$

giving $n/2 + 1 = d + X$.

Suppose v is a word accepted by M. Then M on input v goes through each loop of length x_i some number of times $a_i \geqslant 0$, where

$$k - 1 + \sum_{i=1}^{k} a_i x_i = |v|.$$

If additionally $|v| = |w|$, then by Lemma 29 we have $a_1 = a_2 = \cdots = a_k$, and hence $v = w$. Thus

$$D(w) \geqslant \lfloor n/2 + 1 \rfloor - q = d + X - X = d.$$

Below we prove that w is overlap-free. □

Proof that the word w in Theorem 30 is overlap-free. Suppose a word uu is contained in w.

Proof that the number of 2s in uu is either 0 or 2. Let $o_1, \ldots, o_{2a}$ denote the occurrences of 2s in uu and suppose $a \geqslant 1$. Let $\delta_i = o_{i+1} - o_i$. Then the sequence $(\delta_1, \cdots, \delta_a)$ is an interval in the sequence

$$(x_1 - 1, x_1, x_2 - 1, x_2, \ldots, x_{k-1} - 1, x_{k-1}, x_k - 1).$$

Since $x_i > x_{i+1} + 1$, in particular $|x_i - x_{i+1}| > 1$ and so this sequence is injective, i.e., no two entries are the same. But $(o_1, \cdots, o_a) = (o_{a+1} - |u|, \cdots, o_{2a} - |u|)$. So $\delta_{a+1} = o_{a+2} - o_{a+1} = o_2 - o_1 = \delta_1$ which implies $a = 1$.

So either Case 1 or Case 2 below obtains. **Case 1: The number of 2s in uu is zero.** Then certainly $uu\,\mathrm{first}(u)$ is not contained in w, since the infinite Thue–Morse word is overlap-free. **Case 2: The number of 2s in uu is two.** Then we have one of the following two cases.

1. uu is contained in a word of the form

   $$t_1 \cdots t_{x_i} \quad 2 \quad t_1 \cdots t_{x_{i+1}-1} \quad 2 \quad t_1 \cdots t_{x_{i+1}}.$$

   We guard against that by making sure that

   - $t_{x_i} \neq t_{x_{i+1}-1}$ (Lemma 29) and
   - $2 \neq t_{x_{i+1}}$ (the Thue–Morse word uses only the letters 0 and 1)

2. uu is contained in a word of the form

   $$t_1 \cdots t_{x_i-1} \quad 2 \quad t_1 \cdots t_{x_i} \quad 2 \quad t_1 \cdots t_{x_{i+1}-1}.$$

   Since uu contains exactly two 2s and the t_j are not 2s, it follows that $uu = a2b2c$ where a, b, c are words over the binary alphabet $\{0,1\}$. Then $u = a2b_1 = b_2 2c$ where $b = b_1 b_2$, so $a = b_2$, $c = b_1$ and so actually $u = a2c$ and $t_1 \cdots t_{x_i} = b = ca$. Here then $|ca| = x_i$. If $|a| \leqslant 2$ then consequently

   $$x_i - 2 \leqslant |c| \leqslant x_{i+1} - 1,$$

   which contradicts $x_{i+1} < x_i - 1$. If $|a| \geqslant 2$ then we appeal to Lemma 31.

□

Lemma 31. *$t_{x_i-2}t_{x_i-1}\,2\,t_1 \cdots t_{x_i}2$ cannot be a factor of a square having only two 2s.*

Proof. The Thue–Morse word is a concatenation of disjoint occurrences of the words 01 and 10. Each of these two words are of the form $z\overline{z}$ where $\overline{z} = 1 - z$. The idea now is that if x_i is odd then say it ends in a lone 0 and 2, 02; then adding the next control bit will give something ending in 012, preventing a square.

More precisely, since $t_1 \cdots t_{x_i-1}2$ having odd or even length ends in say $z\overline{z}2$ or $z\overline{z}a2$ respectively, and then $t_1 \cdots t_{x_i-1}t_{x_i}2$ ends in $z\overline{z}b2$ or $z\overline{z}a\overline{a}2$, respectively; either way $t_1 \cdots t_{x_i-1}2$ and $t_1 \cdots t_{x_i-1}t_{x_i}2$ are incompatible. □

Definition 2 yields the following lemma.

Lemma 32. *Let $(q_0, q_1, \cdots)$ be the sequence of states visited by an NFA M given an input word w. For any t, t_1, t_2, and r_i, s_i with*

$$(p_1, r_1, \ldots, r_{t-2}, p_2) = (q_{t_1}, \ldots, q_{t_1+t})$$

and

$$(p_1, s_1, \ldots, s_{t-2}, p_2) = (q_{t_2}, \ldots, q_{t_2+t}),$$

we have $r_i = s_i$ for each i.

Note that in Lemma 32, it may very well be that $t_1 \neq t_2$.

Theorem 33. *Overlap-free binary words have deficiency bound 1.*

Proof. Suppose w is a word satisfying $D(w) \geqslant 2$ and consider the sequence of states visited in a witnessing computation. As in the proof of Theorem 41, either there is a state that is visited four times, and hence there is a cube in w, or there are three *state cubes* (states that are visited three times each), and hence there are three squares in w. By Theorem 24, a overlap-free binary word can only contain squares of length 2^a, $3 \cdot 2^a$, and hence can only contain powers u^i where $|u|$ is of the form 2^a, $3 \cdot 2^a$, and $i \leqslant 2$.

In particular, the length of one of the squares in the three state cubes must divide the length of another. So if these two state cubes are disjoint then the shorter one repeated can replace one occurrence of the longer one, contradicting Lemma 32.

So suppose we have two state cubes, at states p_1 and p_2, that overlap. At p_1 then we read consecutive words ab that are powers $a = u^i$, $b = u^j$ of a word u, and since there are no cubes in w it must be that $i = j = 1$ and so actually $a = b$. And at p_2 we have words c, d that are powers of a word v and again the exponents are 1 and $c = d$.

The overlap means that in one of the two excursions of the same length starting and ending at p_1, we visit p_2. By uniqueness of the accepting path we then visit p_2 in both of these excursions. If we suppose the state cubes are chosen to be of minimal length then we only visit p_2 once in each excursion. If we write $a = rs$ where r is the word read when going from p_1 to p_2, and s is the word going from p_2 to p_1, then $c = sr$ and w contains $rsrsr$. In particular, w contains an overlap. □

Remark 34. In computability theory, the effective Hausdorff dimension dim and effective packing dimension Dim of a single infinite binary sequence $\mathbf{u}$ are defined, and related to Kolmogorov complexity C. It is shown (see [2, Theorem 13.3.4 and Corollary 13.11.12]) that

$$\dim(\mathbf{u}) = \liminf_n \frac{C(u_1 \cdots u_n)}{n}, \text{ and } \quad \mathrm{Dim}(\mathbf{u}) = \limsup_n \frac{C(u_1 \cdots u_n)}{n}.$$

These results, together with the idea that automatic complexity is a miniaturization of Kolmogorov complexity, constitute our motivation for making Definitions 35 and 37 below.

Definition 35. For an infinite word $\mathbf{u}$ define the *deterministic automatic Hausdorff dimension* of $\mathbf{u}$ by

$$I(\mathbf{u}) = \liminf_{u \text{ prefix of } \mathbf{u}} \frac{A_D(u)}{|u|}.$$

and the *deterministic automatic packing dimension* of $\mathbf{u}$ by

$$S(\mathbf{u}) = \limsup_{u \text{ prefix of } \mathbf{u}} \frac{A_D(u)}{|u|}.$$

The connection between effective dimension and automatic dimension is not merely by analogy, as Theorem 36 shows.

Theorem 36. *If* $\mathbf{x}$ *is an infinite word with* $\dim(\mathbf{x}) > 0$*, then* $I(\mathbf{x}) > 0$.

Proof. This follows from the Kolmogorov complexity calculation in [9, Theorem 9]. □

For nondeterministic complexity, in light of Theorem 3 it is natural to make the following definition.

Definition 37. Define the *nondeterministic automatic Hausdorff dimension* of $\mathbf{u}$ by

$$I_N(\mathbf{u}) = 2 \cdot \liminf_{u \text{ prefix of } \mathbf{u}} \frac{A_N(u)}{|u|}$$

and define S_N analogously.

Theorem 38 (Shallit and Wang's Theorem 18)**.** $\frac{1}{3} \leqslant I(\mathbf{t}) \leqslant S(\mathbf{t}) \leqslant \frac{2}{3}$.

We are now ready to strengthen Theorem 38.

Theorem 39. *$I(\mathbf{t}) = \frac{1}{2}$, and $I_N(\mathbf{t}) = S_N(\mathbf{t}) = 1$.*

Proof. The inequality $I(\mathbf{t}) \geqslant \frac{1}{2}$ and the fact that $I_N(\mathbf{t}) = S_N(\mathbf{t}) = 1$ follow from the observation that the proof of Theorem 33 applies equally for deterministic complexity. The inequality $I(\mathbf{t}) \leqslant \frac{1}{2}$ was already implicit in the proof of [9, Theorem 18]. Let $T(m) = t_0 \cdots t_{m-1}$. In the table they give, with $m = 2^{2n+1}$, we read off the inequality $A_D(T(m)) \leqslant m + 3 - 2^{2n} = \frac{m}{2} + 3$. □

3.4 Almost square-free words

Definition 40 (Fraenkel and Simpson [3])**.** A word whose square factors all belong to the set $\{00, 11, 0101\}$ is called *almost square-free.*

Theorem 41. *A word that is almost square-free has a deficiency bound of 1.*

Proof. It is easy to verify for words of length at most 3. Suppose now w has length at least 4. Suppose w is a word of a length $n \in \{2k, 2k+1\}$ where $k \geqslant 2$, with deficiency at least 2. Then there are $q = k - 1 \geqslant 1$ states occupied at $n + 1$ times. So $n + 1 \in \{2k+1, 2k+2\} = \{2q+3, 2q+4\}$ times. There are at least $2q+3$ times and only q states, so by the Extended Pigeonhole Principle (Theorem 17), we are in one of the following Cases 1–3.

- Case 1. There is at least one state that is visited at least 5 times. Then by Theorem 16, ***w* contains a fourth power**.
- Case 2. There is at least one state p_1 that is visited at least 4 times and another state $p_2 \neq p_1$ that is visited at least 3 times. Then by Theorem 16, there is a cube xxx and a square yy in w. Since ***w* has no squares of length > 4**, we must have $|xx| \leqslant 4$ and $|yy| \leqslant 4$ and hence $1 \leqslant |x| \leqslant 2$ and $1 \leqslant |y| \leqslant 2$. We next consider possible lengths of x and y.
  - Subcase $|x| = 2$. Say $x = ab$ where $|a| = |b| = 1$. If $a \neq b$ then $xxx \in \{101010, 010101\}$ so 1010 occurs in w, but ***w* does not contain 1010**; if $a = b$ then 0000 or 1111 occurs in w, contra assumption.

  - Subcase $|x| = 1$, $|y| = 2$: In this case, the xxx and yy occurrences must be disjoint, because the states in a yy occurrence are $p_2p_3p_2p_3p_2$ for some p_3 which must be disjoint from $p_1p_1p_1p_1$ when $p_1 \neq p_2$. But then we can replace these by $p_2p_3p_2p_3p_2p_3p_2$ and p_1p_1, respectively, giving two distinct state sequences leading to acceptance, contradicting Lemma 32.
  - Subcase $|x| = 1$, $|y| = 1$: In this case again the occurrences of xxx and yy must be disjoint, since $p_1 \neq p_2$. We can replace p_1^4 and p_2^3 by p_1 and p_2^6, respectively, again contradicting Lemma 32.

- Case 3. There are at least 3 states p_1, p_2, p_3 (all distinct) that are each visited at least 3 times. Then by Theorem 16, there are three squares u_iu_i at three distinct states p_i, $1 \leqslant i \leqslant 3$. By assumption $|u_iu_i| \leqslant 4$ so $|u_i| \leqslant 2$.
  - Subcase 3.1. $|u_i| = |u_j| = 1$ for two values $1 \leqslant i < j \leqslant 3$. Then the argument is entirely analogous to that in Case 2.
  - Subcase 3.2 $|u_j| = |u_k| = 2$ for two values $1 \leqslant j < k \leqslant 3$.
    * Subsubcase 3.2.1. If disjoint, we can replace u_j^2 by u_k^2 to get u_k^4, again a **fourth power**, by the argument of Subcase 3.1.
    * Subsubcase 3.2.2. If nondisjoint with full overlap then

$$p_ja_1p_ja_2p_j$$

and

$$p_kb_1p_kb_2p_k$$

become

$$p_jp_kp_jp_kp_jp_k$$

and immediately we get **10101 or 01010 or a fourth power in $\boldsymbol{w}$**;
    * Subsubcase 3.2.3. If partial overlap only then $p_ja_1p_ja_2p_j$ and $p_kb_1p_kb_2p_k$ become, by Lemma 32, $p_jap_jap_j$ and $p_kbp_kbp_k$ and then

$$p_jap_jp_kp_jp_kbp_k$$

By Lemma 32 again, this must be

$$p_jp_kp_jp_kp_jp_kp_jp_k = (p_jp_k)^4$$

and so the read word must be of the form $abababa$, giving **an occurrence of 1010 (if $\boldsymbol{a \neq b}$) or of a 7th power (if $\boldsymbol{a = b}$) in $\boldsymbol{w}$**.

Thus all cases are covered and the Theorem is proved. □

Corollary 42. *There is an infinite binary word having hereditary deficiency bound of 1.*

Proof. We have two distinct proofs. On the one hand, Fraenkel and Simpson [3] show there is an infinite almost square-free binary word, and the result follows from Theorem 41. On the other hand, the infinite Thue–Morse word is overlap-free (Theorem 27) and the result follows from Theorem 33. □

Conjecture 43. There is an infinite binary word having hereditary deficiency 0.

Remark 44. We obtained some numerical evidence for Conjecture 43. For instance, we found that there are 108 binary words of length 18 having hereditary deficiency 0.

References

[1] Cristian S. Calude, Kai Salomaa, and Tania K. Roblot. Finite state complexity. *Theoret. Comput. Sci.*, 412(41):5668–5677, 2011.

[2] Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic randomness and complexity.* Theory and Applications of Computability. Springer, New York, 2010.

[3] Aviezri S. Fraenkel and R. Jamie Simpson. How many squares must a binary sequence contain? *Electron. J. Combin.*, 2:Research Paper 2, approx. 9 pp. (electronic), 1995.

[4] A. O. Gelfond. Sur les nombres qui ont des propriétés additives et multiplicatives données. *Acta Arith.*, 13:259–265, 1967/1968.

[5] Kayleigh Hyde. Nondeterministic finite state complexity. Master's thesis, University of Hawaii at Manoa, U.S.A., 2013. http://math.hawaii.edu/home/theses/MA_2013_Hyde.pdf.

[6] R. C. Lyndon and M. P. Schützenberger. The equation $a^M = b^N c^P$ in a free group. *Michigan Math. J.*, 9:289–298, 1962.

[7] Johannes F. Morgenbesser, Jeffrey Shallit, and Thomas Stoll. Thue-Morse at multiples of an integer. *J. Number Theory*, 131(8):1498–1512, 2011.

[8] Jeffrey Shallit. *A Second Course in Formal Languages and Automata Theory.* Cambridge University Press, New York, NY, USA, 1 edition, 2008.

[9] Jeffrey Shallit and Ming-Wei Wang. Automatic complexity of strings. *J. Autom. Lang. Comb.*, 6(4):537–554, 2001. 2nd Workshop on Descriptional Complexity of Automata, Grammars and Related Structures (London, ON, 2000).

[10] R. O. Shelton and R. P. Soni. Chains and fixing blocks in irreducible binary sequences. *Discrete Math.*, 54(1):93–99, 1985.

[11] A. Thue. Über die gegenseitige Lage gleicher Teile gewisser Zeichenreihen. *Norske Vid. Skrifter I Mat.-Nat. Kl., Christiania*, 1:1–67, 1912.