Decidability and Enumeration for Automatic Sequences: A Survey

Jeffrey Shallit

School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada
shallit@cs.uwaterloo.ca

Abstract. In this talk I will report on some recent results concerning decidability and enumeration for properties of automatic sequences. This is work with Jean-Paul Allouche, Émilie Charlier, Narad Rampersad, Dane Henshall, Luke Schaeffer, Eric Rowland, Daniel Goč, and Hamoon Mousavi.

1 Introduction

An infinite sequence $\mathbf{a} = (a_n)_{n\geq 0}$ over a finite alphabet is said to be k-*automatic* if there exists a deterministic finite automaton (with outputs associated with the states) such that after completely processing the input n expressed in base k, the automaton reaches some state q with output a_n [17,5]. A typical example of such a sequence is the Thue-Morse sequence

$$\mathbf{t} = t_0 t_1 t_2 \cdots = \texttt{011010011001} \cdots ,$$

which is generated by the automaton in Figure 1. Here the input is n, expressed in base 2.

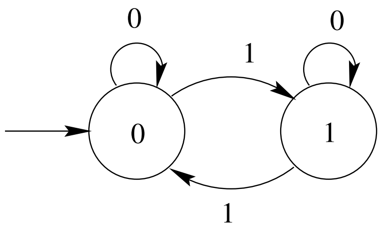

Fig. 1. Finite automaton generating the Thue-Morse sequence $\mathbf{t}$

The Thue-Morse sequence is named after the Norwegian mathematician Axel Thue [45,46,6], who discovered it in 1912, although it also appears if one reads "between the lines" in an 1851 paper of Prouhet [39], and it has since been rediscovered many times (e.g., [34,22]). For more information about $\mathbf{t}$, see the survey [4].

A.A. Bulatov and A.M. Shur (Eds.): CSR 2013, LNCS 7913, pp. 49–63, 2013.

Below we list just a few of the properties of $\mathbf{t}$ that people have studied. By a *factor* we mean a contiguous block of symbols inside another word.

1. $\mathbf{t}$ is not ultimately periodic.
2. $\mathbf{t}$ contains no factor that is an *overlap*, that is, a word of the form $axaxa$, where a is a single letter and x is an arbitrary finite word [45,46,6].
3. $\mathbf{t}$ has infinitely many distinct palindromic factors and infinitely many distinct antipalindromic factors. (A *palindrome* is a word equal to its reverse; an example of a palindrome in Russian is доход ("income"). An *antipalindrome* is a word of the form $x\,\overline{x^R}$, where x^R denotes the reverse of x and $\overline{0} = 1$, $\overline{1} = 0$.)
4. The number $p(n)$ of distinct palindromic factors of length n in $\mathbf{t}$ is given by
$$p(n) = \begin{cases} 0, & \text{if } n \text{ odd and } n \geq 5; \\ 1, & \text{if } n = 0; \\ 2, & \text{if } 1 \leq n \leq 4, \text{ or } n \text{ even and } 3 \cdot 4^k + 2 \leq n \leq 4^{k+1} \text{ for } k \geq 1; \\ 4, & \text{if } n \text{ even and } 4^k + 2 \leq n \leq 3 \cdot 4^k \text{ for } k \geq 1; \end{cases}$$
see [7]. A similar expression exists for the number $p'(n)$ of distinct antipalindromic factors of length n.
5. $\mathbf{t}$ contains infinitely many distinct square factors xx, but for each such factor we have $|x| = 2^n$ or $3 \cdot 2^n$, for $n \geq 1$. Examples of squares in Russian include дядя ("uncle") and кускус ("couscous").
6. $\mathbf{t}$ is *mirror-invariant*: if x is a finite factor of $\mathbf{t}$, then so is its reverse x^R.
7. $\mathbf{t}$ is *recurrent*, that is, every factor that occurs, occurs infinitely often [34].
8. $\mathbf{t}$ is *uniformly recurrent*, that is, for all factors x occurring in $\mathbf{t}$, there is a constant $c(x)$ such that two consecutive occurrences of x are separated by at most $c(x)$ symbols [35, pp. 834 et seq.].
9. $\mathbf{t}$ is *linearly recurrent*, that is, it is uniformly recurrent and furthermore there is a constant C such that $c(x) \leq C|x|$ for all factors x [35, pp. 834 et seq.]. In fact, the optimal bound is given by $c(1) = 3$, $c(2) = 8$, and $c(n) = 9 \cdot 2^e$ for $n \geq 3$, where $e = \lfloor \log_2(n-2) \rfloor$.
10. The lexicographically least sequence in the orbit closure of $\mathbf{t}$ is $\overline{t_1}\ \overline{t_2}\ \overline{t_3}\ \cdots$, which is also 2-automatic [2].
11. The *subword complexity* $\rho(n)$ of $\mathbf{t}$, which is the function counting the number of distinct factors of $\mathbf{t}$, is given by
$$\rho(n) = \begin{cases} 2^n, & \text{if } 0 \leq n \leq 2; \\ 2n + 2^{t+2} - 2, & \text{if } 3 \cdot 2^t \leq n \leq 2^{t+2} + 1; \\ 4n - 2^t - 4, & \text{if } 2^t + 1 \leq n \leq 3 \cdot 2^{t-1}; \end{cases}$$
see [8,32].
12. $\mathbf{t}$ has an unbordered factor of length n if $n \not\equiv 1 \pmod 6$ [19]. (Here by an *unbordered* word y we mean one with no expression in the form $y = uvu$ for words u, v with u nonempty.)

Recently I and my co-authors J.-P. Allouche, E. Charlier, D. Goč, D. Henshall, N. Rampersad, E. Rowland, and L. Schaeffer, have developed and implemented a decision procedure by which all these assertions, and many others, can be feasibly verified and/or generated in a purely mechanical fashion. In this talk I will explain how our method works, what has been done so far, and what remains to be done.

2 Logic

By Th($\mathbb{N}, +, 0, 1, <$) I mean the set of all true first-order sentences in the logical theory of the natural numbers with addition. In such a theory, for example, we can express the so-called "Chicken McNuggets" theorem [47, Lesson 5.8, Problem 1] to the effect that 43 is the largest integer that cannot be represented as a non-negative integer linear combination of $6, 9$, and 20, as follows:

$$(\forall n > 43\ \exists x, y, z \geq 0 \text{ such that } n = 6x + 9y + 20z) \land \neg(\exists x, y, z \geq 0 \text{ such that } 43 = 6x + 9y + 20z). \quad (1)$$

Here, of course, "$6x$" is shorthand for the expression "$x + x + x + x + x + x$", and similarly for $9y$ and $20z$.

Thanks to the work of Presburger [37,38] we know that Th($\mathbb{N}, +, 0, 1, <$) is *decidable*: that is, there exists an algorithm that, given a sentence in the theory, will decide its truth.

In fact, there is a relatively simple proof of this fact, based on finite automata, and due to Büchi [11,12], Elgot [21], and Hodgson [27]. More recently it has appeared (without attribution) in the textbook of Sipser [44, §6.2] and progress has been made on its complexity (e.g., [29]). The idea is to represent integers in a integer base $k \geq 2$ using the alphabet $\Sigma_k = \{0, 1, \ldots, k-1\}$. We can then represent n-tuples of integers as words over the alphabet Σ_k^n, padding with leading zeroes, if necessary. Thus, for example, the pair $(21, 7)$ can be represented in base 2 by the word

$$[1,0][0,0][1,1][0,1][1,1].$$

Then the relation $x + y = z$ can be checked by a simple 2-state automaton depicted in Figure 2, where transitions not depicted lead to a nonaccepting "dead state".

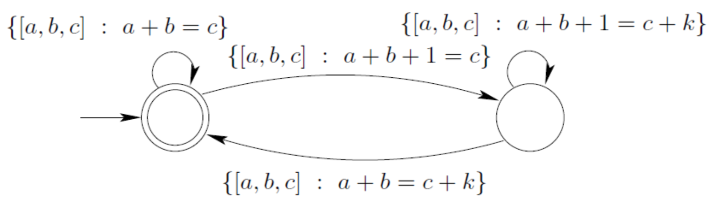

Fig. 2. Checking addition in base k

Relations like $x = y$ and $x < y$ can be checked similarly.

Given a formula with free variables $x_1, x_2, \ldots, x_n$, we construct an automaton accepting the base-k expansion of those n-tuples $(x_1, \ldots, x_n)$ for which the proposition holds. If a formula is of the form $\exists x_1, x_2, \ldots x_n \; p(x_1, \ldots, x_n)$, then we use nondeterminism to "guess" the x_i and check them. If the formula is of the form $\forall p$, we use the equivalence $\forall p \equiv \neg\exists\neg p$; this may require using the subset construction to convert an NFA to a DFA and then flipping the "finality" of states. Finally, the truth of a formula can be checked by using the usual depth-first search techniques to see if any final state is reachable from the start state.

However, even more is true. If we add the function $V_k : \mathbb{N} \to \mathbb{N}$ to our logical theory, where $V_k(x) = k^n$, and k^n is the largest power of k dividing x, it is still decidable by a similar automaton-based technique [10]. By doing so, we gain the capability of deciding many questions about automatic sequences. Thus we have

Theorem 1. *There is an algorithm that, given a predicate phrased using only the universal and existential quantifiers, indexing into a given automatic sequence* **a***, addition, subtraction, logical operations, and comparisons, will decide the truth of that proposition.*

We call such a predicate an *automatic predicate.*

Although the worst-case running time of our algorithm is bounded above by

$$2^{2^{\cdot^{\cdot^{\cdot^{2^{p(N)}}}}}},$$

where the number of 2's in the exponent is equal to the number of quantifiers, p is a polynomial, and N is the number of states needed to describe the underlying automatic sequence, it turns out that in practice, significantly better running times are usually achieved.

3 Periodicity

An infinite word **a** is *periodic* if it is of the form $x^\omega = xxx\cdots$ for a finite nonempty word x. It is *ultimately periodic* if it is of the form yx^ω for a (possibly empty) finite word y.

Honkala [28] was the first to prove that ultimate periodicity is decidable for automatic sequences. Later, Leroux [31], and, more recently, Marsault and Sakarovitch [33] gave efficient algorithms for the problem.

Using our approach, we can easily see that periodicity is decidable for k-automatic sequences [3]. It suffices to express ultimately periodicity as an automatic predicate:

$$\exists p \geq 1, N \geq 0 \; \forall i \geq N \; \mathbf{a}[i] = \mathbf{a}[i+p].$$

When we run this on the Thue-Morse sequence, we discover (as expected) that **t** is not ultimately periodic.

4 Repetitions

Repetitions in sequences have been studied for over a hundred years. We defined overlaps above in § 1. Other classic repetitions include *squares* (factors of the form xx, where x is nonempty) and *cubes* (factors of the form xxx).

Thue [46] proved that $\mathbf{t}$ contains no overlaps; that is, $\mathbf{t}$ is overlap-free. Using our technique, we can express the property of having an overlap $axaxa$ beginning at position N with $|ax| = p$, as follows: $\mathbf{a}[N..N+p] = \mathbf{a}[N+p..N+2p]$. So the corresponding automatic predicate for $\mathbf{t}$ is

$$\exists p \geq 1, N \geq 0 \ \mathbf{t}[N..N+p] = \mathbf{t}[N+p..N+2p],$$

or, in other words,

$$\exists p \geq 1, N \geq 0 \ \forall i, 0 \leq i \leq p \ \mathbf{t}[N+i] = \mathbf{t}[N+p+i].$$

From now on, we will abbreviate predicates like the one above by writing the first form only.

We programmed up our decision procedure and verified that indeed $\mathbf{t}$ is overlap-free [3].

We can also ask about the lengths and positions of squares in the Thue-Morse sequence. Here we can create an automaton to accept

$$\{(N,p)_2 \ : \ p \geq 1 \text{ and } N \geq 0 \text{ and } \mathbf{t}[N..N+p-1] = \mathbf{t}[N+p..N+2p-1]\}.$$

When we do so, we get the automaton depicted below in Figure 3 (computed by Daniel Goč).

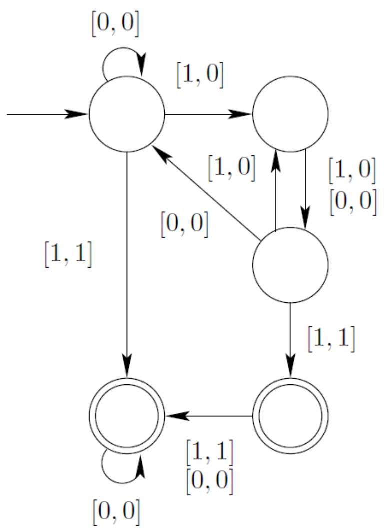

Fig. 3. Positions and lengths of squares in Thue-Morse

From this automaton, we easily recover the results of [36,8] that the only squares xx that occur have $|x| = 2^n$ or $|x| = 3 \cdot 2^n$ for $n \geq 0$, and all those lengths occur. The positions where these squares occur were previously given by Brown et al. [9].

5 Critical Exponent

We can define more general repetitions as follows: a word x is an α-power for $\alpha \geq 1$ if we can write $x = y^e y'$ where $e = \lfloor \alpha \rfloor$ and y' is a prefix of y and $|x| = \alpha |y|$. Thus, for example, `abracadabra` is an $\frac{11}{7}$-power and the Russian word люблю ("love") is a $\frac{5}{3}$-power. The techniques above suffice to check if a k-automatic sequence has α-powers, using the following predicate:

$$\exists N \geq 0, p, q \geq 1 \ \mathbf{a}[N..N+p-q-1] = \mathbf{a}[N+q..N+p-1] \text{ and } p = \alpha q.$$

However, this observation alone does not suffice to compute the so-called *critical exponent* of $\mathbf{a}$, which is the supremum over all rational α such that $\mathbf{a}$ has α-power factors.

It turns out that the critical exponent is also computable for automatic sequences [43,42]. More generally, we can extend the concept of k-automatic sets of natural numbers to k-automatic sets of non-negative rational numbers, as follows. Given a word $x \in (\Sigma_k \times \Sigma_k)^*$, define

$$\text{quo}_k(x) = \frac{[\pi_1(x)]_k}{[\pi_2(x)]_k},$$

where $\pi_i(x)$ is the projection of x onto its i'th coordinate $(i = 1, 2)$, and $[x]_k$ is the integer represented by the word x in base k. This is extended to languages L in the usual way:

$$\text{quo}_k(L) = \{\text{quo}_k(x) \ : \ x \in L\}.$$

Then we have

Theorem 2. *If M is a DFA, then* $\sup \text{quo}_k(L(M))$ *is either rational or infinite, and it is computable.*

We can now apply this theorem to our problem. Let $\mathbf{a}$ be a k-automatic sequence. Using the techniques above, we can compute a DFA M accepting the language

$$L = \{(p, q) \ : \ \exists N \ \mathbf{a}[N..N+p-q-1] = \mathbf{a}[N+q..N+p-1]\},$$

which represents all fractional powers p/q occurring in $\mathbf{a}$. Now, applying Theorem 2, we get the desired result.

As an application, we considered an old construction of Leech [30] for squarefree words: consider the fixed point $\mathbf{L}$ of the 13-uniform morphism φ given by

$$0 \to 0121021201210$$
$$1 \to 1202102012021$$
$$2 \to 2010210120102$$

Using our method, we proved that the critical exponent of L is actually $\frac{15}{8}$. Furthermore, if x is a $\frac{15}{8}$ power occurring in $\mathbf{L}$, then $|x| = 15 \cdot 13^i$ for some $i \geq 0$. See [23].

6 Mirror Invariance

We can express the property that $\mathbf{a}$ is mirror-invariant as follows:

$$\forall N \geq 0, \ell \geq 1 \; \exists N' \geq 0 \; \mathbf{a}[N..N+\ell-1] = \mathbf{a}[N'..N'+\ell-1]^R,$$

which is the same as

$$\forall N \geq 0, \ell \geq 1 \; \exists N' \geq 0 \; \forall i, \; 0 \leq i < \ell \; \mathbf{a}[N+i] = \mathbf{a}[N'+\ell-i-1],$$

which can be easily checked by our method.

7 Recurrence

We can express the property that $\mathbf{a}$ is recurrent by saying that for each factor, and each integer M there exists a copy of that factor occurring at a position after M in $\mathbf{a}$. This corresponds to the following predicate:

$$\forall N, M \geq 0, \ell \geq 1 \; \exists M' \geq M \; \mathbf{a}[N..N+\ell-1] = \mathbf{a}[M'..M'+\ell-1].$$

An easy argument shows that an infinite word $\mathbf{a}$ is recurrent if and only if each finite factor occurs at least twice. This means that the following simpler predicate suffices:

$$\forall N \geq 0, \ell \geq 1 \; \exists M \neq N \; \mathbf{a}[N..N+\ell-1] = \mathbf{a}[M..M+\ell-1].$$

For uniform recurrence, we need to express the fact that two consecutive occurrences of each factor are separated by no more than C positions. Since there are only finitely many factors of each length, we can take C to be the maximum of the constants corresponding to each factor of that length. Thus we get the following predicate:

$$\forall \ell \geq 1 \, \exists C \geq 1 \, \forall N \geq 0 \, \exists M \text{ with } N < M \leq N+C \; \mathbf{a}[N..N+\ell-1] = \mathbf{a}[M..M+\ell-1].$$

For linear recurrence, we have to work harder, since at first glance knowing if there is a factor at distance $C\ell$ seems to require multiplication, which we cannot perform. Instead, we construct a DFA accepting the language

$$L = \{(n,\ell)_k \; : \; \exists \, i \geq 0 \text{ s. t. } \forall \, j, 0 \leq j < \ell \text{ we have } a[i+j] = a[i+n+j] \text{ and } \nexists \, t, 0 < t < n \text{ s. t. } \forall \, j, 0 \leq j < \ell \text{ we have } a[i+j] = a[i+t+j] \; \}.$$

Note that $(n,\ell)_k \in L$ iff there exists some factor of length ℓ for which the next occurrence is at distance n. Then linear recurrence corresponds to $\mathrm{quo}_k(L) < \infty$, which we can test using Theorem 2.

8 Orbit Closure

The *orbit* of a sequence $\mathbf{a} = a_0 a_1 a_2 \cdots$ is the set of all sequences under the shift, that is, the set $\mathcal{S} = \{a_i a_{i+1} a_{i+2} \cdots \ : \ i \geq 0\}$. The *orbit closure* is the topological closure $\overline{\mathcal{S}}$ under the usual topology. In other words, a sequence $\mathbf{b} = b_0 b_1 b_2 \cdots$ is in $\overline{\mathcal{S}}$ if and only if, for each $j \geq 0$, the prefix $b_0 \cdots b_j$ is a factor of $\mathbf{a}$.

In general, the cardinality of the orbit closure is uncountable. On the other hand, the k-automatic sequences are countable. Hence most sequences in the orbit closure of a k-automatic sequence are not automatic themselves. However, we can use our method to show that two distinguished sequences, the lexicographically least and lexicographically greatest sequences in the orbit closure, are indeed k-automatic.

For example, Currie [18] showed that the lexicographically least sequence in the orbit closure of the Rudin-Shapiro sequence

$$\mathbf{r} = r_0 r_1 r_2 \cdots = \texttt{000100100001110100010010111000} \cdots$$

is $\texttt{0}\mathbf{r}$, thus confirming a conjecture in [3].

9 Unbordered Factors

Recall that a word is *bordered* if it can be expressed as uvu for words u, v with u nonempty, and otherwise it is unbordered. Currie and Saari [19] proved that $\mathbf{t}$ has an unbordered factor of length n if $n \not\equiv 1 \pmod 6$. However, these are not the only lengths with an unbordered factor; for example,

$$\texttt{0011010010110100110010110100101}$$

is an unbordered factor of length 31. We can express the property that $\mathbf{t}$ has an unbordered factor of length ℓ as follows:

$$\exists N \geq 0 \ \forall j, 1 \leq j \leq \ell/2 \ \mathbf{t}[N..N+j-1] \neq \mathbf{t}[N+\ell-j..N+\ell-1].$$

Using our technique, we can create a DFA to accept the base-2 representations of all such ℓ. Using our method, we were able to prove [23]

Theorem 3. *There is an unbordered factor of length ℓ in $\mathbf{t}$ if and only iff $(\ell)_2 \notin 1(01^*0)^*10^*1$.*

10 Enumeration

Up to now we have focused on deciding properties of automatic sequences. In many cases, however, we can actually count the number $T(n)$ of length-n factors of an automatic sequence having a particular property P. Here by "count" we mean, give an algorithm A to compute $T(n)$ efficiently, that is, in time bounded by a polynomial in $\log n$. Although *finding* the algorithm A may not be particularly efficient (and indeed, has a "tower-of-2's" running time depending on the predicate to express P), but once we have it, we can compute $T(n)$ quickly.

One example is subword complexity, the number of distinct length-n factors of a sequence. To count these factors, we create a DFA M accepting the language

$$\begin{aligned} &\{(n,\ell)_k \ : \ \mathbf{a}[n..n+\ell-1] \text{ is the first occurrence of the given factor}\} \\ = &\{(n,\ell)_k \ : \ \forall n' < n \ \mathbf{a}[n..n+\ell-1] \neq \mathbf{a}[n'..n'+\ell-1]\}. \end{aligned}$$

Once we have M, the number of ℓ corresponding to a given n is just the subword complexity. It then turns out [16] that this number can be expressed as the product

$$vM_{a_1}\cdots M_{a_i}w$$

for suitable vectors v, w and matrices $M_0, \ldots, M_{k-1}$, where $a_1 \cdots a_i$ is the base-k representation of n, thus giving an efficient algorithm to compute it.

In a similar way, we can handle

- palindrome complexity (the number of distinct length-n palindromic factors) [1];
- the number of words whose reversals are also factors;
- the number of squares of a given length;
- the number of unbordered factors [24];

and so forth.

For this last example, the number $f(n)$ of unbordered factors of length n, we carried out an explicit computation for the Thue-Morse sequence. The resulting computation allowed us to prove that $f(n) \leq n$ for $n \geq 4$ and $f(n) = n$ infinitely often [24].

11 Synchronization

Sometimes even more is true: we can build a DFA to accept the language

$$\{(n, T(n))_k \ : \ n \geq 0\},$$

where $T(n)$ counts some interesting property about an automatic sequence. In this case we say, following Carpi [15,13,14], that the function T is k-synchronized. When a function T is k-synchronized, we have $T(n) = O(n)$ and further, we can compute it in $O(\log n)$ time [25].

Many enumerations about automatic sequences are now known to be k-synchronized. These include

- the separator sequence [15];
- the repetitivity index [13];
- the recurrence function [16];
- the appearance function [16];

– the subword complexity function [25];
– the number of factors of length n that are primitive [25].

Here is a novel example of synchronization. Blondin-Massé et al. studied the *longest palindromic suffix* of a finite word w, defined to be the unique longest word x such that $x = x^R$ and there exists y such that $w = yx$; it is denoted $\mathrm{LPS}(w)$. Given an infinite word $\mathbf{a} = a_0 a_1 \cdots$, they defined the related function $\mathrm{LLPS}_{\mathbf{a}}(n) = |\,\mathrm{LPS}(\mathbf{a}[0..n])|$, which measures the length of the longest palindromic suffix of each prefix.

We can see that $\mathrm{LLPS}_{\mathbf{a}}(n)$ is k-synchronized, as we can build an automaton to accept

$$\{(n,i)_k : \mathbf{a}[n-i+1..n] = \mathbf{a}[n-i+1..n]^R \text{ and } \exists j, 0 \le j \le n-i \;\; \mathbf{a}[j..n] \ne \mathbf{a}[j..n]^R\}.$$

Blondin-Massé et al. also studied a related function, given by

$$H_{\mathbf{a}}(n) = \begin{cases} \mathrm{LLPS}_{\mathbf{a}}(n), & \text{if } \mathrm{LPS}(\mathbf{a}[0..n]) \text{ does not occur in } \mathbf{a}[0..n-1]\ ; \\ 0, & \text{otherwise.} \end{cases}$$

This sequence $H_{\mathbf{a}}$ is also k-synchronized, as we can express it as

$$\begin{aligned} \{(n,i)_k :\ & \mathbf{a}[n-i+1..n] = \mathbf{a}[n-i+1..n]^R \text{ and} \\ & \exists j, 0 \le j \le n-i \quad \mathbf{a}[j..n] \ne \mathbf{a}[j..n]^R \text{ and} \\ & \forall \ell, 0 \le \ell \le n-i \quad \mathbf{a}[\ell..\ell+i-1] \ne \mathbf{a}[n-i+1..n]\} \\ \cup\ \{(n,0)_k :\ & \mathbf{a}[n-i+1..n] = \mathbf{a}[n-i+1..n]^R \text{ and} \\ & \exists j, 0 \le j \le n-i \quad \mathbf{a}[j..n] \ne \mathbf{a}[j..n]^R \text{ and} \\ & \exists \ell, 0 \le \ell \le n-i \quad \mathbf{a}[\ell..\ell+i-1] = \mathbf{a}[n-i+1..n]\}. \end{aligned}$$

12 Paperfolding

Up to now we have only applied our decision procedure to a single automatic sequence. Sometimes, however, it is desirable to talk about the properties of a *family* of such sequences. A famous example of such a family is the set of *paperfolding sequences*. Given a sequence of *unfolding instructions* $\mathbf{f} = f_0 f_1 f_2 \cdots$ over the alphabet $\{0,1\}$, the paperfolding sequence $\mathbf{P_f} = p_1 p_2 p_3 \cdots$ is defined as the limit of the finite sequences given by

$$x_0 = f_0$$
$$x_{n+1} = x_i \; f_i \; \overline{x_i^R},$$

where, as before, $\overline{0} = 1$ and $\overline{1} = 0$. The *regular paperfolding sequence*

`001001100011011`$\cdots$

corresponds to the unfolding instructions `000`$\cdots$.

It turns out that many properties of these sequence are also decidable using our method. The key observation is due to Luke Schaeffer: a known formula to compute the n'th term of a paperfolding sequence [20] can be implemented by the following automaton of 5 states (depicted below in Figure 4) that takes, as input, a prefix of a sequence of unfolding instructions in parallel with the base-2 expansion of n (starting with the least significant digit), and computes the n'th term of the corresponding paperfolding sequence.

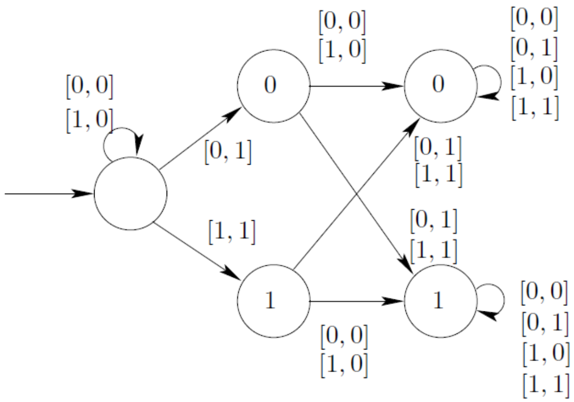

Fig. 4. Automaton for the paperfolding sequences

This makes it possible to prove many of the known results about paperfolding sequences, and some new ones, in a purely mechanical fashion. We just mention one new result, answering a question of Narad Rampersad [26]:

Theorem 4. *If* $\mathbf{f} = f_0 f_1 f_2 \cdots$ *and* $\mathbf{g} = g_0 g_1 g_2 \cdots$ *are two different sequences of unfolding instructions, and the smallest index where they differ is* $f_i = g_i$, *then* $P_{\mathbf{f}}$ *and* $P_{\mathbf{g}}$ *have no factors of length* $\geq 14 \cdot 2^i$ *in common.*

13 Implementation

As mentioned previously, the extraordinary upper bound on the running time of the decision procedure means that care has to be taken during the implementation. Dane Henshall and my master's student Daniel Goč independently wrote code that takes a description of an automatic sequence and a predicate as input and translates the predicate to the appropriate automaton. In Goč's algorithm, DFA minimization is done using Brzozowski's algorithm, which often seems to outperform the usual methods. With these implementations we have been able to find new machine proofs of many old results and also some new ones.

14 Inexpressible Predicates

It is natural to wonder if other kinds of properties of automatic sequences are solvable using our method. One natural candidate that seems difficult is testing abelian squarefreeness. We say that a nonempty word x is an *abelian square* if it of the form ww' with $|w| = |w'|$ and w' a permutation of w. (An example in English is the word `reappear`, and three examples in Russian are крекер ("cracker") отлетело ("(it) flew away") and ротатора ("(of) rotator", genitive case).)

Recently my student Luke Schaeffer has shown that the predicate for abelian squarefreeness is indeed inexpressible, in general [41]. To do so, he considers the regular paperfolding sequence

$$\mathbf{f} = f_1 f_2 f_3 \cdots = \texttt{0010011000110110001001110011011} \cdots ,$$

which is 2-automatic, and then the language

$$L = \{(n,i)_2 \ : \ \mathbf{f}[i..i+n-1] \text{ is a permutation of } \mathbf{f}[i+n..i+2n-1]\}.$$

If abelian squarefreeness were expressible, then L would be regular, but he shows it is not [41].

15 Open Questions

There are still many interesting questions that are unresolved. For example, it is known that, given a DFA M, we can decide if $\text{quo}_k(L(M)) \subseteq \mathbb{N}$ [40]. However, the following related problems are still open:

Open Question 1. *Are any of the following problems recursively solvable? Given a DFA M accepting $L \subseteq (\Sigma_k \times \Sigma_k)^*$,*

(a) Does there exist $x \in L$ such that $\text{quo}_k(x) \in \mathbb{N}$?
(b) Do there exist infinitely many $x \in L$ such that $\text{quo}_k(x) \in \mathbb{N}$?
(c) Is there an infinite subset $S \subseteq \mathbb{N}$ such that $S \subseteq \text{quo}_k(L)$?

Similarly, if L is represented by a pushdown automaton instead of a DFA, we can ask.

Open Question 2. *Is $\sup \text{quo}_k(L)$ computable for context-free languages L?*

References

1. Allouche, J.P., Baake, M., Cassaigne, J., Damanik, D.: Palindrome complexity. Theoret. Comput. Sci. 292, 9–31 (2003)
2. Allouche, J.P., Currie, J., Shallit, J.: Extremal infinite overlap-free binary words. European J. Combinatorics 5, R27 (1998), http://www.combinatorics.org/ojs/index.php/eljc/article/view/v5i1r27

3. Allouche, J.P., Rampersad, N., Shallit, J.: Periodicity, repetitions, and orbits of an automatic sequence. Theoret. Comput. Sci. 410, 2795–2803 (2009)
4. Allouche, J.P., Shallit, J.: The ubiquitous Prouhet-Thue-Morse sequence. In: Ding, C., Helleseth, T., Niederreiter, H. (eds.) Proceedings of Sequences and Their Applications, SETA 1998, pp. 1–16. Springer (1999)
5. Allouche, J.P., Shallit, J.: Automatic Sequences: Theory, Applications, Generalizations. Cambridge University Press (2003)
6. Berstel, J.: Axel Thue's work on repetitions in words. In: Leroux, P., Reutenauer, C. (eds.) Séries Formelles et Combinatoire Algébrique, Publications du LaCim, Université du Québec à Montréal, vol. 11, pp. 65–80 (1992)
7. Blondin Massé, A., Brlek, S., Garon, A., Labbé, S.: Combinatorial properties of f-palindromes in the Thue-Morse sequence. Pure Math. Appl. 19(2-3), 39–52 (2008), http://www.mat.unisi.it/newsito/puma/public_html/19_2_3/4.pdf
8. Brlek, S.: Enumeration of factors in the Thue-Morse word. Disc. Appl. Math. 24, 83–96 (1989)
9. Brown, S., Rampersad, N., Shallit, J., Vasiga, T.: Squares and overlaps in the Thue-Morse sequence and some variants. RAIRO Inform. Théor. App. 40, 473–484 (2006)
10. Bruyère, V., Hansel, G., Michaux, C., Villemaire, R.: Logic and p-recognizable sets of integers. Bull. Belgian Math. Soc. 1, 191–238 (1994); Corrigendum, Bull. Belg. Math. Soc. 1, 577 (1994)
11. Büchi, J.R.: Weak secord-order arithmetic and finite automata. Zeitschrift für mathematische Logik und Grundlagen der Mathematik 6, 66–92 (1960); reprinted in Mac Lane, S., Siefkes, D. (eds.): The Collected Works of J. Richard Büchi, pp. 398–424. Springer (1990)
12. Büchi, J.R.: On a decision method in restricted second order arithmetic. In: Logic, Methodology and Philosophy of Science (Proc. 1960 Internat. Congr.), pp. 1–11. Stanford University Press (1962)
13. Carpi, A., D'Alonzo, V.: On the repetitivity index of infinite words. Internat. J. Algebra Comput. 19, 145–158 (2009)
14. Carpi, A., D'Alonzo, V.: On factors of synchronized sequences. Theoret. Comput. Sci. 411, 3932–3937 (2010)
15. Carpi, A., Maggi, C.: On synchronized sequences and their separators. RAIRO Inform. Théor. App. 35, 513–524 (2001)
16. Charlier, E., Rampersad, N., Shallit, J.: Enumeration and decidable properties of automatic sequences. Internat. J. Found. Comp. Sci. 23, 1035–1066 (2012)
17. Cobham, A.: Uniform tag sequences. Math. Systems Theory 6, 164–192 (1972)
18. Currie, J.: Lexicographically least words in the orbit closure of the Rudin-Shapiro word. Theoret. Comput. Sci. 41, 4742–4746 (2011)
19. Currie, J.D., Saari, K.: Least periods of factors of infinite words. RAIRO Inform. Théor. App. 43, 165–178 (2009)
20. Dekking, F.M., Mendès France, M., van der Poorten, A.J.: Folds! Math. Intelligencer 4, 130–138, 173–181, 190–195 (1982); erratum 5, 5 (1983)
21. Elgot, C.C.: Decision problems of finite automata design and related arithmetics. Trans. Amer. Math. Soc. 98, 21–51 (1961)
22. Euwe, M.: Mengentheoretische Betrachtungen über das Schachspiel. Proc. Konin. Akad. Wetenschappen, Amsterdam 32, 633–642 (1929)
23. Goč, D., Henshall, D., Shallit, J.: Automatic theorem-proving in combinatorics on words. In: Moreira, N., Reis, R. (eds.) CIAA 2012. LNCS, vol. 7381, pp. 180–191. Springer, Heidelberg (2012)

24. Goč, D., Mousavi, H., Shallit, J.: On the number of unbordered factors. In: Dediu, A.-H., Martín-Vide, C., Truthe, B. (eds.) LATA 2013. LNCS, vol. 7810, pp. 299–310. Springer, Heidelberg (2013)
25. Goč, D., Schaeffer, L., Shallit, J.: The subword complexity of k-automatic sequences is k-synchronized (2012) (submitted), preprint available at http://arxiv.org/abs/1206.5352
26. Goč, D., Schaeffer, L., Shallit, J.: A new approach to the paperfolding sequences (manuscript in preparation, 2013)
27. Hodgson, B.: Décidabilité par automate fini. Ann. Sci. Math. Québec 7, 39–57 (1983)
28. Honkala, J.: A decision method for the recognizability of sets defined by number systems. RAIRO Inform. Théor. App. 20, 395–403 (1986)
29. Klaedtke, F.: Bounds on the automata size for Presburger arithmetic. ACM Trans. Comput. Logic 9(2), article 11 (March 2008), http://doi.acm.org/10.1145/1342991.1342995
30. Leech, J.: A problem on strings of beads. Math. Gazette 41, 277–278 (1957)
31. Leroux, J.: A polynomial time Presburger criterion and synthesis for number decision diagrams. In: 20th IEEE Symposium on Logic in Computer Science (LICS 2005), pp. 147–156. IEEE Press (2005)
32. Luca, A.D., Varricchio, S.: Some combinatorial properties of the Thue-Morse sequence and a problem in semigroups. Theoret. Comput. Sci. 63, 333–348 (1989)
33. Marsault, V., Sakarovitch, J.: Ultimate periodicity of b-recognisable sets: a quasi-linear procedure (2013), preprint available at http://arxiv.org/abs/1301.2691
34. Morse, M.: Recurrent geodesics on a surface of negative curvature. Trans. Amer. Math. Soc. 22, 84–100 (1921)
35. Morse, M., Hedlund, G.A.: Symbolic dynamics. Amer. J. Math. 60, 815–866 (1938)
36. Pansiot, J.J.: The Morse sequence and iterated morphisms. Inform. Process. Lett. 12, 68–70 (1981)
37. Presburger, M.: Über die Volständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In: Sparawozdanie z I Kongresu Matematyków Krajów Slowianskich, Warsaw, pp. 92–101 (1929)
38. Presburger, M.: On the completeness of a certain system of arithmetic of whole numbers in which addition occurs as the only operation. Hist. Phil. Logic 12, 225–233 (1991)
39. Prouhet, E.: Mémoire sur quelques relations entre les puissances des nombres. C. R. Acad. Sci. Paris 33, 225 (1851)
40. Rowland, E., Shallit, J.: k-automatic sets of rational numbers. In: Dediu, A.-H., Martín-Vide, C. (eds.) LATA 2012. LNCS, vol. 7183, pp. 490–501. Springer, Heidelberg (2012)
41. Schaeffer, L.: Abelian powers in automatic sequences are not always automatic. Talk at CanadaDAM 2013 Conference, St. John's, Newfoundland (June 2013)
42. Schaeffer, L., Shallit, J.: The critical exponent is computable for automatic sequences. Int. J. Found. Comput. Sci. (2012) (to appear)
43. Shallit, J.: The critical exponent is computable for automatic sequences. In: Ambroz, P., Holub, S., Másaková, Z. (eds.) Proceedings 8th International Conference Words 2011. Elect. Proc. Theor. Comput. Sci, vol. 63, pp. 231–239 (2011)
44. Sipser, M.: Introduction to the Theory of Computation. Cengage Learning, 3rd edn. (2013)

45. Thue, A.: Über unendliche Zeichenreihen. Norske vid. Selsk. Skr. Mat. Nat. Kl. 7, 1–22 (1906); reprinted in Nagell, T. (ed.): Selected Mathematical Papers of Axel Thue, Universitetsforlaget, Oslo, pp. 139–158 (1977)
46. Thue, A.: Über die gegenseitige Lage gleicher Teile gewisser Zeichenreihen. Norske vid. Selsk. Skr. Mat. Nat. Kl 1, 1–67 (1912); reprinted in Nagell, T. (ed.): Selected Mathematical Papers of Axel Thue, Universitetsforlaget, Oslo, pp. 413–478 (1977)
47. Wah, A., Picciotto, H.: Algebra: Themes, Tools, Concepts. Creative Publications, Mountain View (1994), `http://www.mathedpage.org/attc/attc.html`