Closed, palindromic, rich, privileged, trapezoidal, and balanced words in automatic sequences

Luke Schaeffer
Massachusetts Institute of Technology
Cambridge, MA 02139
USA
lrschaeffer@gmail.com

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

Submitted: xxx; Accepted: xxx; Published: XX
Mathematics Subject Classifications: 11B85, 68R15, 03B25, 03B35, 11A63.

Abstract

We prove that the property of being closed (resp., palindromic, rich, privileged trapezoidal, balanced) is expressible in first-order logic for automatic (and some related) sequences. It therefore follows that the characteristic function of those n for which an automatic sequence $\mathbf{x}$ has a closed (resp., palindromic, privileged, rich, trapezoidal, balanced) factor of length n is itself automatic. For privileged words this requires a new characterization of the privileged property. We compute the corresponding characteristic functions for various famous sequences, such as the Thue-Morse sequence, the Rudin-Shapiro sequence, the ordinary paperfolding sequence, the period-doubling sequence, and the Fibonacci sequence. Finally, we also show that the function counting the total number of palindromic factors in the prefix of length n of a k-automatic sequence is not k-synchronized.

Keywords: decision procedure, closed word, palindrome, rich word, privileged word, trapezoidal word, balanced word, Thue-Morse sequence, Rudin-Shapiro sequence, period-doubling sequence, paperfolding sequence, Fibonacci word.

1 Introduction

Recently a wide variety of different kinds of words have been studied in the combinatorics on words literature, including the six flavors of the title: closed, palindromic, rich, privileged, trapezoidal, and balanced words. In this paper we show that, for k-automatic sequences $\mathbf{x}$ (and some analogs, such as the so-called "Fibonacci-automatic" sequences [19]), the property of a factor belonging to each class is expressible in first-order logic; more precisely, in the theory $\mathrm{Th}(\mathbb{N}, +, n \to \mathbf{x}[n])$. Previously we did this for unbordered factors [22].

As a consequence of our results, and the pioneering work of Büchi [9] and Hodgson [24], we get that (for example) the characteristic sequence of those lengths for which a factor of that length belongs to each class is k-automatic, and the number of such factors of each length forms a k-regular sequence. (For more about the connection to logic, see the excellent survey [5]. For definitions of k-automatic and k-regular, see, for example, [2].)

Using an implementation of a decision procedure for first-order expressible properties, we can give explicit expressions for the lengths of factors in each class for some famous sequences, such as the Thue-Morse sequence, the Rudin-Shapiro sequence, the period-doubling sequence, and the ordinary paperfolding sequence. For some of the properties, these expressions are surprisingly complicated.

2 Notation and definitions

As usual, if $w = xyz$, we say that x is a prefix of w, that z is a suffix of w, and y is a factor of w. By $|x|_w$ we mean the number of (possibly overlapping) occurrences of w as a factor of x. For example, $|\texttt{confrontation}|_{\texttt{on}} = 3$. By x^R we mean the reversal (sometimes called mirror image) of the word x. Thus, for example, $(\texttt{drawer})^R = \texttt{reward}$. By Σ_k we mean the alphabet $\{0, 1, \ldots, k-1\}$ of cardinality k.

A factor w of x is said to be *right-special* if both wa and wb are factors of x, for two distinct letters a and b.

A word x is a *palindrome* if $x = x^R$. Examples of palindromes in English include `radar` and `redivider`. Droubay, Justin, and Pirillo [18] proved that every word of length n contains at most $n+1$ distinct palindromic factors (including the empty word). A word is called *rich* if it contains exactly this many. For example, the English words `logology` and `Mississippi` are both rich. For example, `Mississippi` has the following distinct nonempty palindromic factors:

`M`, `i`, `s`, `p`, `ss`, `pp`, `sis`, `issi`, `ippi`, `ssiss`, `ississi`.

For more about rich words, see [21, 17, 6, 7].

A nonempty word w is a *border* of a word x if w is both a prefix and a suffix of x. A word x is called *closed* (aka "complete first return") if it is of length $\leqslant 1$, or if it has a border w with $|x|_w = 2$. For example, `abracadabra` is closed because of the border `abra`, while `alfalfa` is closed because of the border `alfa`. The latter example shows that, in the definition, the prefix and suffix are allowed to overlap. For more about closed words, see [3].

A word x is called *privileged* if it is of length $\leqslant 1$, or it has a border w with $|x|_w = 2$ that is itself privileged. Clearly every privileged word is closed, but `mama` is an example of an English word that is closed but not privileged. For more about privileged words, see [26, 28, 29, 20].

A word x is called *trapezoidal* if it has, for each $n \geqslant 0$, at most $n+1$ distinct factors of length n. Since for $n = 1$ the definition requires at most 2 distinct factors, this means that every trapezoidal word can be defined over an alphabet of at most 2 letters. An example in English is the word `deeded`. See, for example, [16, 15, 17, 8].

A word x is called *balanced* if, for all factors y, z of the same length of x and all letters a of the alphabet, the inequality $||y|_a - |z|_a| \leqslant 1$ holds. Otherwise it is *unbalanced*. An example of a balanced word in English is `banana`.

We use the terms "infinite sequence" and "infinite word" as synonyms. In this paper, names of infinite words are given in the **bold** font. All infinite words are indexed starting at position 0. If $\mathbf{x} = x_0x_1x_2\cdots$ is an infinite word, with each x_i a single letter, then by $\mathbf{x}[i..j]$ for $j \geqslant i-1$ we mean the finite word $x_ix_{i+1}\cdots x_j$. By $[i..j]$ we mean the set $\{i, i+1, \ldots, j\}$.

3 Sequences

In this section we define the five sequences we will study. For more information about these sequences, see, for example, [2].

The *Thue-Morse sequence* $\mathbf{t} = t_0t_1t_2\cdots = \texttt{01101001}\cdots$ is defined by the relations $t_0 = 0$, $t_{2n} = t_n$, and $t_{2n+1} = 1 - t_n$ for $n \geqslant 0$. It is also expressible as the fixed point, starting with 0, of the morphism $\mu : 0 \to 01,\ 1 \to 10$.

The *Rudin-Shapiro sequence* $\mathbf{r} = r_0r_1r_2\cdots = \texttt{00010010}\cdots$ is defined by the relations $r_0 = 0$, $r_3 = 1$, $r_{2n} = r_n$, $r_{4n+1} = r_n$, $r_{8n+7} = r_{2n+1}$, $r_{16n+3} = r_{8n+3}$, $r_{16n+11} = r_{4n+3}$ for $n \geqslant 0$. It is also expressible as the image, under the coding $\tau : n \to \lfloor n/2 \rfloor$, of the fixed point, starting with 0, of the morphism $\rho : 0 \to 01,\ 1 \to 02,\ 2 \to 31,\ 3 \to 32$.

The *ordinary paperfolding sequence* $\mathbf{p} = p_0p_1p_2\cdots = \texttt{00100110}\cdots$ is defined by the relations $p_{2n+1} = p_n$, $p_{4n} = 0$, $p_{4n+2} = 1$ for $n \geqslant 0$. It is also expressible as the image, under the coding τ above, of the fixed point, starting with 0, of the morphism $\rho : 0 \to 01$, $1 \to 21$, $2 \to 03$, $3 \to 23$.

The *period-doubling sequence* $\mathbf{d} = d_0d_1d_2\cdots = \texttt{10111010}\cdots$ is defined by the relations $d_{2n} = 1$, $d_{4n+1} = 0$, and $d_{4n+3} = d_n$ for $n \geqslant 0$. It is also expressible as the fixed point, starting with 1, of the morphism $\delta : 1 \to 10,\ 0 \to 11$.

The *Fibonacci sequence* $\mathbf{f} = f_0f_1f_2\cdots = \texttt{01001010}\cdots$ is the fixed point, starting with 0, of the morphism $\varphi : 0 \to 01,\ 1 \to 0$.

4 Common predicates

Before we see how rich words, privileged words, closed words, etc. can be phrased as first-order predicates, let us define a few basic predicates.

First, we have the two basic predicates $\textsc{In}(i, r, s)$, which is true if and only if $i \in [r..s]$:

$$\textsc{In}(i, r, s) := (i \geqslant r) \ \wedge \ (i \leqslant s),$$

and $\textsc{Subs}(i, j, m, n)$, which is true if and only if $[i..i+m-1] \subseteq [j..j+n-1]$:

$$\textsc{Subs}(i, j, m, n) := (j \leqslant i) \ \wedge \ (i + m \leqslant j + n).$$

Next, we have the predicate

$$\text{FactorEq}(i,j,n) := \forall k\ (k < n) \implies (\mathbf{x}[i+k] = \mathbf{x}[j+k]),$$

which checks whether $\mathbf{x}[i..i+n-1]$ and $\mathbf{x}[j..j+n-1]$ are equal by comparing them at corresponding positions, $\mathbf{x}[i+k]$ and $\mathbf{x}[j+k]$, for $k = 0, \ldots, n-1$. By a similar principle, we can compare $\mathbf{x}[i..i+n-1]$ with $\mathbf{x}[j..j+n-1]^R$, but in this paper we only need the special case $i = j$, i.e., palindromes:

$$\text{Pal}(i,n) := \forall k\ (k < n) \implies (\mathbf{x}[i+k] = \mathbf{x}[i+n-1-k]).$$

From FactorEq, we derive other useful predicates. For instance, the predicate

$$\text{Occurs}(i,j,m,n) := (m \leqslant n)\ \wedge\ (\exists k\ (k+m \leqslant n)\ \wedge\ \text{FactorEq}(i, j+k, m))$$

tests whether $\mathbf{x}[i..i+m-1]$ is a factor of $\mathbf{x}[j..j+n-1]$. We also define

$$\text{Border}(i,m,n) := \text{In}(m,1,n)\ \wedge\ \text{FactorEq}(i, i+n-m, m),$$

which is true if and only if $\mathbf{x}[i..i+m-1]$ is a border of $\mathbf{x}[i..i+n-1]$.

In the next five sections, we obtain our results using the implementation of a decision procedure for the corresponding properties, written by Hamoon Mousavi, and called `Walnut`, to prove theorems by machine computation. The software is available for download at

https://cs.uwaterloo.ca/~shallit/papers.html .

All of the predicates in this paper can easily be translated into Hamoon Mousavi's `Walnut` program. Files for the examples in this paper are available at the same URL as above, so the reader can easily run and verify the results.

5 Closed words

We can create a predicate $\text{Closed}(i,n)$ that asserts that $\mathbf{x}[i..i+n-1]$ is closed as follows:

$$(n \leqslant 1)\ \vee\ (\exists j\ (j < n)\ \wedge\ \text{Border}(i,j,n)\ \wedge \neg\text{Occurs}(i, i+1, j, n-2))$$

Theorem 1. *(a) There is a closed factor of Thue-Morse of every length.*

(b) There is a 15-state automaton accepting the base-2 representation of those n for which there is a closed factor of Rudin-Shapiro of length n.

(c) There is an 11-state automaton accepting the base-2 representation of those n for which there is a closed factor of the paperfolding sequence of length n. It is depicted below in Figure 1.

(d) There is a closed factor of the period-doubling sequence of every length.

(e) There is a closed factor of the Fibonacci sequence of every length.

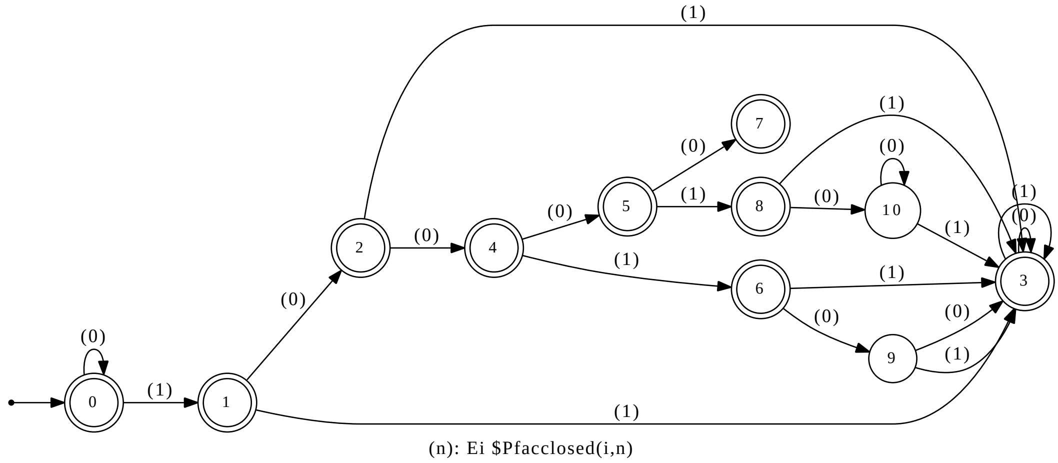

Figure 1: Automaton for lengths of closed factors of the paperfolding sequence

As we have seen above, the Thue-Morse sequence contains a closed factor of every length. We now turn to enumerating $f(n)$, the number of such factors of length n. Here are the first few values of $f(n)$:

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$f(n)$	1	2	2	2	4	4	6	4	8	8	10	8	12	8	8	8

The first step is to create a predicate $\mathrm{UCF}(i,n)$ which is true if $\mathbf{t}[i..i+n-1]$ is a closed factor of $\mathbf{t}$ of length n, and is also the first occurrence of that factor:

$$\mathrm{UCF}(i,n) := \textsc{Closed}(i,n) \ \wedge\ \neg\textsc{Occurs}(i,0,n,i+n-1).$$

The associated DFA then gives us (as in [22]) a linear representation for $f(n)$: vectors v, w and a matrix-valued homomorphism $\mu : \{0,1\} \to \mathbb{N}^{k \times k}$ such that $f(n) = v\mu(x)w^T$ for all x that are valid base-2 representations of n.

They are as follows (with $\mu(i) = M_i$):

$$
M_0 = \begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2
\end{bmatrix}
$$

$$
M_1 = \begin{bmatrix}
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2
\end{bmatrix}
$$

$$
v = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
$$

$$
w = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}
$$

This linear representation can be minimized, using the algorithm in [4], obtaining

$$M_0' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 1 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 & -3 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 4 & -4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 11/4 & -1 \end{bmatrix}$$

$$M_1' = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 2 & -2 & -1 & 4 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & -4 & 0 & 10 & -8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1/2 & 7/2 & -1 \end{bmatrix}$$

$$v' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$w' = \begin{bmatrix} 1 & 2 & 2 & 2 & 4 & 4 & 6 & 4 & 8 & 8 \end{bmatrix}$$

From this, using the technique in [22], we can obtain the following relations

$$\begin{aligned}
f(8n) &= -2f(2n+1) + f(4n) + 2f(4n+1) \\
f(8n+1) &= -2f(2n+1) + 3f(4n+1) \\
f(8n+3) &= -2f(2n+1) + 2f(4n+1) + f(4n+3) \\
f(8n+4) &= 2f(2n+1) - \frac{5}{2}f(4n+1) + f(4n+2) + \frac{1}{2}f(4n+3) + f(8n+2) \\
f(8n+5) &= 2f(4n+3) \\
f(8n+7) &= -4f(2n+1) + 2f(4n+1) - 2f(4n+3) + 2f(8n+6) \\
f(16n+2) &= -6f(2n+1) + \frac{13}{2}f(4n+1) + \frac{1}{2}f(4n+3) \\
f(16n+6) &= -\frac{1}{2}f(4n+1) + f(4n+2) + \frac{3}{2}f(4n+3) + f(8n+2) \\
f(16n+10) &= 2f(4n+3) + f(8n+6) \\
f(32n+14) &= -2f(2n+1) - \frac{7}{2}f(4n+1) + 3f(4n+2) + \frac{7}{2}f(4n+3) + 3f(8n+2) \\
f(32n+30) &= 24f(2n+1) - 6f(4n+1) + 14f(4n+3) - 4f(8n+2) \\
&\quad -12f(8n+6) + 5f(16n+14).
\end{aligned}$$

From these we can verify the following theorem by a tedious induction on n:

Theorem 2. *Let $n \geqslant 8$ and let $k \geqslant -1$ be an integer. Then*

$$f(n) = \begin{cases} 2^{k+4}, & \text{if } 15 \cdot 2^k < n \leqslant 18 \cdot 2^k; \\ 2n - 20 \cdot 2^k - 2, & \text{if } 18 \cdot 2^k < n \leqslant 19 \cdot 2^k; \\ 56 \cdot 2^k - 2n + 2, & \text{if } 19 \cdot 2^k < n \leqslant 20 \cdot 2^k; \\ 4n - 64 \cdot 2^k - 4, & \text{if } 20 \cdot 2^k < n \leqslant 22 \cdot 2^k; \\ 112 \cdot 2^k - 4n + 4, & \text{if } 22 \cdot 2^k < n \leqslant 24 \cdot 2^k; \\ 2^{k+4}, & \text{if } 24 \cdot 2^k < n \leqslant 28 \cdot 2^k; \\ 8n - 208 \cdot 2^k - 8, & \text{if } 28 \cdot 2^k < n \leqslant 30 \cdot 2^k. \end{cases}$$

6 Palindromic words

Palindromes in words have a long history of being studied; for example, see [1].

It is already known that many aspects of palindromes in k-automatic sequences are expressible in first-order logic; see, for example, [13].

In this section, we turn to a variation on palindromic words, the so-called "maximal palindromes". For us, a factor x of an infinite word $\mathbf{w}$ is a *maximal palindrome* if x is a palindrome, while no factor of the form axa for a a single letter occurs in $\mathbf{w}$. This differs slightly from the existing definitions, which deal with the maximality of *occurrences* [25].

The property of being a maximal palindrome is easily expressible in terms of predicates defined above:

$$\textsc{MaxPal}(i,n) := \textsc{Pal}(i,n) \wedge (\forall j\ ((j \geqslant 1) \wedge \textsc{FactorEq}(i,j,n)) \implies \mathbf{x}[j-1] \neq \mathbf{x}[j+n])$$

Using this, and our program, we can easily prove the following result:

Theorem 3. *(a) The Thue-Morse sequence contains maximal palindromes of length $3 \cdot 4^n$ for each $n \geqslant 0$, and no others. These palindromes are of the form $\mu^{2n}(010)$ and $\mu^{2n}(101)$ for $n \geqslant 0$.*

(b) The Rudin-Shapiro sequence contains exactly 8 maximal palindromes. They are

$$0100010, 0001000, 1110111, 1011101, 0010000100, 1101111011,$$
$$1110110111, 10000100100001.$$

(c) The ordinary paperfolding sequence contains exactly 6 maximal palindromes. They are

$$001100, 110011, 011000110, 100111001, 1000110110001, 0111001001110.$$

(d) The period-doubling sequence contains maximal palindromes of lengths $3 \cdot 2^n - 1$ for all $n \geqslant 0$, and no others.

(e) The Fibonacci sequence contains no maximal palindromes at all.

We now turn to a result about counting palindromes in automatic sequences. To state it, we first need to describe representations of integers in base k. By $(n)_k$ we mean the string over the alphabet $\Sigma_k := \{0, 1, \ldots, k-1\}$ representing n in base k, and having no leading zeroes. This is generalized to representing r-tuples of integers by changing the alphabet to Σ_k^r, and padding shorter representations on the left, if necessary, with leading zeroes. Thus, for example, $(6, 3)_2 = [1, 0][1, 1][0, 1]$. By $[w]_k$, for a word w, we mean the value of w when interpreted as an integer in base k.

Next, we need the concept of k-synchronization [12, 10, 11, 23]. We say a function $f(n)$ is *k-synchronized* if there is a finite automaton accepting the language $\{(n, f(n))_k : n \geqslant 0\}$.

The following is a useful lemma:

Lemma 4. *If $(f(n))_{n\geqslant 0}$ is a k-synchronized sequence, and f is unbounded, then there exists a constant $c > 0$ such that $f(n) \geqslant cn$ infinitely often.*

Proof. Since f is unbounded, there exists $n > 0$ such that $f(n) > k^N$, where N is the number of states in the minimal automaton accepting L^R, where $L = \{(n, f(n))_k : n \geqslant 0\}$. Apply the pumping lemma to the string $z = (n, f(n))_k^R$. It says that we can write $z = uvw$, where $|uv| \leqslant n$ and w has nonzero elements in both components. Then, letting $(n_i, f(n_i)) = [(uv^i w)^R]_k$ we see that this subsequence has the desired property. □

Theorem 5. *The function counting the number of distinct palindromes in the prefix of length n of a k-automatic sequence is not necessarily k-synchronized.*

Proof. Our proof is based on two infinite words, $\mathbf{a} = (a_i)_{i\geqslant 0}$ and $\mathbf{b} = (b_i)_{i\geqslant 0}$.

The word $\mathbf{a}$ is defined as follows:

$$a_i = \begin{cases} (k \bmod 2) + 1, & \text{if there exists } k \text{ such that } 4^{k+1} - 4^k \leqslant i \leqslant 4^{k+1} + 4^k; \\ 0, & \text{otherwise.} \end{cases}$$

The word $\mathbf{b}$ is defined as follows:

$$b_i = \begin{cases} (k \bmod 2) + 1, & \text{if there exists } k \text{ such that } 4^{k+1} - 4^k < i < 4^{k+1} + 4^k; \\ 0, & \text{otherwise.} \end{cases}$$

We leave the easy proof that $\mathbf{a}$ and $\mathbf{b}$ are 4-automatic to the reader.

We now compare the palindromes in $\mathbf{a}$ to those in $\mathbf{b}$. From the definition, every palindrome in either sequence is clearly in

$$0^* + 1^* + 2^* + 0^*1^*0^* + 0^*2^*0^*.$$

Since $\mathbf{a}$ has longer blocks of 1's and 2s than $\mathbf{b}$ does, there may be some palindromes of the form 1^i or 2^i that occur in a prefix of $\mathbf{a}$, but not the corresponding prefix of $\mathbf{b}$. Conversely, $\mathbf{b}$ may contain palindromes of the form 0^i that do not occur in the corresponding prefix of $\mathbf{a}$. The net difference, if any, is at most a constant.

Call an occurrence of a factor in a word *novel* if it is the first occurrence in the word. The palindromes not of the form a^i, where $a \in \{0, 1, 2\}$, are of the form $0^i1^j0^i$ or $0^i2^j0^i$, and must be centered at a position that is a power of 4. It is not hard to see that if $\mathbf{a}[i..i+n-1]$ is a novel palindrome occurrence of this form in $\mathbf{a}$, then $\mathbf{b}[\mathbf{i}..\mathbf{i}+\mathbf{n}-\mathbf{1}]$ is also a novel palindrome occurrence of this form.

On the other hand, for each $k \geqslant 1$, there are two palindromes that occur in $\mathbf{b}$ but not $\mathbf{a}$. The first is of the form 01^j0 or 02^j0, since the corresponding factor of $\mathbf{a}$ is either $1 \cdots 1$ or $2 \cdots 2$, and hence has been previously accounted for (as a palindrome of the form 1^* or 2^*). Second, there is a factor of the form $0^*1^*0^*$ or $0^*2^*0^*$ in $\mathbf{b}$ which appears as $20^*1^*0^*$ or $10^*2^*0^*$ in $\mathbf{a}$ because the neighbouring block of 1's or 2's is slightly wider in $\mathbf{a}$ and therefore slightly closer. We conclude that the length-n prefix of $\mathbf{b}$ has $2\log_4 n + O(1)$ more palindromes than the length-n prefix of $\mathbf{a}$.

Now suppose, contrary to what we want to prove, that the number of palindromes in the prefix of length n of a k-automatic sequence is k-synchronized. In particular, the sequence $\mathbf{a}$ (resp., $\mathbf{b}$) is 4-automatic, so the number of palindromes in $\mathbf{a}[0..n-1]$ (resp., $\mathbf{b}[0..n-1]$) is 4-synchronized. Now, using a result of Carpi and Maggi [12, Prop. 2.1], the number of palindromes in $\mathbf{b}[1..n]$ minus the number of palindromes in $\mathbf{a}[1..n]$ is 4-synchronized. But from above this difference is $2\log_4 n+O(1)$, which by Lemma 4 cannot be 4-synchronized. This is a contradiction. $\square$

7 Rich words

As we have seen, a word x is rich if and only if it has $|x|+1$ distinct palindromic subwords. As stated, it does not seem easy to phrase this in first-order logic. Luckily, there is an alternative characterization of rich words, which can be found in [18, Prop. 3]: a word is rich if every prefix p of w has a palindromic suffix s that occurs only once in p. This property can be stated as follows:

$$\begin{aligned}&\textsc{Rich}(i,n) := \forall m\ \textsc{In}(m,1,n) \implies\\&\qquad (\exists j\ \textsc{Subs}(j,i,1,m)\ \wedge \textsc{Pal}(j,i+m-j)\ \wedge\ \neg\textsc{Occurs}(j,i,i+m-j,m-1)).\end{aligned}$$

Finally, we can express the property that $\mathbf{x}$ has a rich factor of length n as follows:

$$\exists i\ \textsc{Rich}(i,n).$$

Theorem 6. *(a) The Thue-Morse sequence contains exactly* 161 *distinct rich factors, the longest being of length* 16.

(b) The Rudin-Shapiro sequence contains exactly 975 *distinct rich factors, the longest being of length* 30.

(c) The ordinary paperfolding sequence contains exactly 494 *distinct rich factors, the longest being of length* 23.

(d) The period-doubling sequence has a rich factor of every length. In fact, every factor of the period-doubling sequence is rich.

(e) Every factor of the Fibonacci sequence is rich.

Of course, (e) was already well known, in much greater generality: Droubay, Justin, and Pirillo [18] proved that every factor of every episturmian word is rich.

8 Privileged words

The recursive definition for privileged words given above in Section 2 is not obviously expressible in first-order logic. However, we can prove a new, alternative characterization of these words, as follows:

Let us say a word w has property P if for all n, $1 \leqslant n \leqslant |w|$, there exists a word x such that $1 \leqslant |x| \leqslant n$, and x occurs exactly once in the first n symbols of w, as a prefix, and x also occurs exactly once in the last n symbols of w, as a suffix.

Lemma 7. *If w is a bordered word with property P, then every border also has property P.*

Proof. Let z be a border of w. Given any $1 \leqslant n \leqslant |z|$, property P for w says that there exists a border x of w such that $1 \leqslant |x| \leqslant n$, and x occurs exactly once in the first (resp., last) n symbols in w. Then observe that the first (resp., last) n symbols of w are precisely the first (resp., last) n symbols of z. Since x is also a border of z, it follows that z has property P. □

Theorem 8. *A word w is privileged if and only if it has property P.*

Proof. If w is privileged, then, by definition, there is a sequence of privileged words $w = w_0, w_1, ..., w_{k-1}, w_k$ such that $|w_k| = 1$ and for all i, w_{i+1} is a prefix and suffix of w_i and occurs nowhere else in w_i. Given an integer n, let x be the largest w_i such that $|w_i| \leqslant n$. Either $i = 0$ because $n = |w|$ and everything works out, or $|w_{i-1}| > n$. Then w_i is a prefix of w_{i-1} (and therefore a prefix of w), and there is no other occurrence of w_i in w_{i-1} (which includes the first n symbols of w). Similarly, w_i is a suffix of w, but does not occur again in the last n symbols of w.

For the other direction, we assume the word has property P and use induction on the length of w. If $|w| = 1$ then the word is privileged immediately. Otherwise, take $n = |w| - 1$ and find the corresponding x promised by property P. Then x is both a prefix and a suffix of w, so it has property P. It is also shorter than w, so by induction, x is privileged. Then x is a privileged prefix and suffix of w which does not occur anywhere else in w (by property P), so w is privileged. □

This property can be represented as a predicate in two different ways. First, let us write a predicate that is true if and only if the prefix $\mathbf{x}[i..i+m-1]$ occurs exactly once in $\mathbf{x}[i..i+n-1]$:

$$\textsc{UniquePref}(i, m, n) := \forall j \ \textsc{In}(j, 1, n-m-1) \implies \neg\textsc{FactorEq}(i, i+j, m).$$

There is a similar expression for whether the suffix $\mathbf{x}[i+n-m..i+n-1]$ occurs exactly once in $\mathbf{x}[i..i+n-1]$:

$$\textsc{UniqueSuff}(i,m,n) := \forall j\ \textsc{In}(j,1,n-m-1) \implies \neg\textsc{FactorEq}(i+n-m, i+n-m-j, m).$$

And finally, our first characterization of privileged words is

$$\textsc{Priv}(i,n) := (n \leqslant 1) \ \vee \ (\forall m\ \textsc{In}(m,1,n) \implies$$
$$(\exists p\ \textsc{In}(p,1,m) \wedge \textsc{Border}(i,p,n) \wedge \textsc{UniquePref}(i,p,m) \wedge \textsc{UniqueSuff}(i+n-m,p,m))).$$

Alternatively, we can write

$$\begin{aligned}\textsc{Priv}'(i,n) := (n \leqslant 1) \ \vee \ (\forall m\ \textsc{In}(m,1,n) \implies \\ (\exists p\ \textsc{In}(p,1,m) \ \wedge \ \textsc{Border}(i,p,n) \wedge \\ \neg\textsc{Occurs}(i,i+1,p,m-1) \ \wedge \ \neg\textsc{Occurs}(i,i+n-m,p,m-1))).\end{aligned}$$

Theorem 9. *(a) There is a 46-state automaton accepting the base-2 expansions of those n for which the Thue-Morse sequence has a privileged factor of length n.*

(b) There is an 84-state automaton accepting the base-2 expansions of those n for which the Rudin-Shapiro sequence has a privileged factor of length n.

(c) There is a 47-state automaton accepting the base-2 expansions of those n for which the paperfolding sequence has a privileged factor of length n.

(d) The set of n for which the period-doubling sequence has a privileged factor of length n is

$$\{0,2\} \ \cup \ \{2n+1 \ : \ n \geqslant 0\}.$$

There is a 4-state automaton accepting the base-2 expansions of those n for which the period-doubling sequence has a privileged factor of length n. It is illustrated below in Figure 2.

(e) There is a 20-state automaton accepting the Zeckendorf representations of those pairs (i,n) for which $\mathbf{f}[i..i+n-1]$ is privileged. It is illustrated below in Figure 3. The Fibonacci word has privileged factors of every length. If n is even there is exactly one privileged factor. If n is odd there are exactly two privileged factors.

Remark 10. For (a)–(d) we used $\textsc{Priv}$ and for (e) we used $\textsc{Priv}'$.

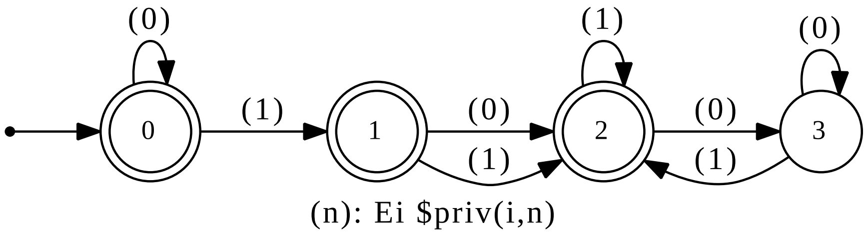

Figure 2: Automaton for lengths of privileged factors of the period-doubling word

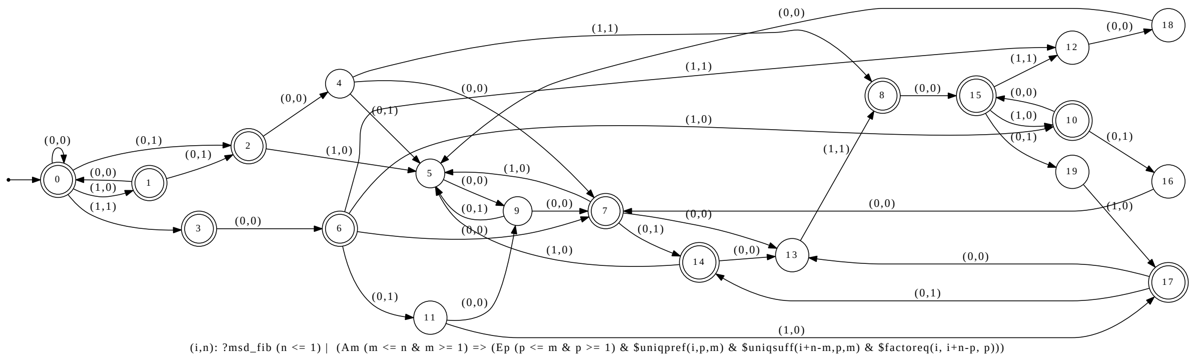

Figure 3: Automaton for privileged factors of the Fibonacci word

We now turn to recovering some of the results of [29] on the number $a(n)$ of privileged factors of the Thue-Morse sequence. Here are the first few values of this sequence

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$a(n)$	1	2	2	2	2	0	4	0	8	0	8	0	4	0	0	0	0

As we did above for closed words, we first make an automaton for the first occurrences of each privileged factor of length n. We then convert this to a linear representation (v, μ, w), obtaining

$$
M_0 = \begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

$$
M_1 = \begin{bmatrix}
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

$$
v = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
$$

$$
w = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}
$$

We can then obtain relations for the sequence $(a(n))_{\geqslant 0}$:

$$\begin{aligned}
a(4n+3) &= a(4n+1)\\
a(8n+1) &= a(4n+1)\\
a(8n+5) &= 0\\
a(16n+6) &= a(4n+1) + a(4n+2) - \frac{1}{2}a(16n+2) + \frac{1}{2}a(16n+4)\\
a(16n+8) &= 3a(4n+1) + 3a(4n+2) - \frac{1}{2}a(16n+2) - \frac{3}{2}a(16n+4)\\
a(16n+10) &= 3a(4n+1) + 3a(4n+2) - \frac{1}{2}a(16n+2) - \frac{3}{2}a(16n+4)\\
a(16n+12) &= a(4n+1) + a(4n+2) - \frac{1}{2}a(16n+2) + \frac{1}{2}a(16n+4)\\
a(32n) &= a(2n+1) - \frac{1}{2}a(4n+1) + 3a(8n+2) - 3a(8n+4)\\
a(32n+2) &= -ra(2n+1) + a(4n+1) + 3a(8n+2) - 2a(8n+4)\\
a(32n+4) &= -a(2n+1) + a(4n+1) + a(8n+2)\\
a(32n+14) &= -a(2n+1) + a(8n+4)\\
a(32n+16) &= -a(2n+1) + a(8n+4)\\
a(32n+20) &= a(32n+18)\\
a(32n+30) &= 2a(2n+1) + a(8n+2) - 3a(8n+4) + 2a(8n+6) - a(32n+18)\\
a(64n+18) &= a(4n+1)\\
a(64n+50) &= 0
\end{aligned}$$

We can also do the same thing for the number of privileged palindromes $(b(n))_{n\geqslant 0}$ in the Thue-Morse sequence. Here are the first few values:

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$b(n)$	1	2	2	2	2	0	4	0	4	0	4	0	4	0	0	0	0

We omit the details and just present the computed relations:

$$
\begin{aligned}
b(4n+3) &= b(4n+1)\\
b(8n+1) &= b(4n+1)\\
b(8n+4) &= b(8n+2)\\
b(8n+5) &= 0\\
b(16n+6) &= b(4n+1)+b(4n+2)\\
b(16n+8) &= b(4n+1)+b(4n+2)\\
b(16n+10) &= b(4n+1)+b(4n+2)\\
b(16n+14) &= -b(4n+1)+b(16n+2)\\
b(32n) &= b(2n+1)-\frac{1}{2}b(4n+1)\\
b(32n+2) &= -b(2n+1)+b(4n+1)+b(8n+2)\\
b(32n+16) &= -b(2n+1)+b(8n+2)\\
b(64n+18) &= b(4n+1)\\
b(64n+50) &= 0
\end{aligned}
$$

9 Trapezoidal words

Trapezoidal words have many different characterizations. The characterization that proves useful to us is the following [8, Prop. 2.8]: a word w is trapezoidal if and only if $|w| = R_w + K_w$. Here R_w is the minimal length ℓ for which w contains no right-special factor of length ℓ, and K_w is the minimal length ℓ for which there is a length-ℓ suffix of w that appears nowhere else in w.

This can be translated into $\mathrm{Th}(\mathbb{N}, +, n \to \mathbf{x}[n])$ as follows: $\textsc{RtSp}(j, n, p)$ is true if and only if $\mathbf{x}[j..j+n-1]$ has a right special factor of length p, and false otherwise:

$$
\begin{aligned}
\textsc{RtSp}(j,n,p) := \exists r\ \exists s\ (&\textsc{Subs}(r,j,p+1,n)\ \wedge\ \textsc{Subs}(s,j,p+1,n)\ \wedge\\
&\textsc{FactorEq}(r,s,p)\ \wedge\ \mathbf{x}[s+p] \neq \mathbf{x}[r+p]).
\end{aligned}
$$

$\textsc{MinRt}(j, n, p)$ is true if and only if p is the smallest integer such that $\mathbf{x}[j..j+n-1]$ has no right special factor of length p:

$$\textsc{MinRt}(j,n,p) := (\neg\textsc{RtSp}(j,n,p))\ \wedge\ (\forall c\ (\neg\textsc{RtSp}(j,n,c)) \implies (c \geqslant p)).$$

$\textsc{UnrepSuf}(j, n, q)$ is true if and only if the suffix of length q of $\mathbf{x}[j..j+n-1]$ is unrepeated in $\mathbf{x}[j..j+n-1]$:

$$\textsc{UnrepSuf}(j,n,q) := \neg\textsc{Occurs}(j+n-q, j, q, n-1).$$

$\textsc{MinUnrepSuf}(j, n, p)$ is true if and only if p is the length of the shortest unrepeated suffix of $\mathbf{x}[j..j+n-1]$:

$$\textsc{MinUnrepSuf}(j,n,p) := \textsc{UnrepSuf}(j,n,q)\ \wedge\ (\forall c\ \textsc{UnrepSuf}(j,n,c) \implies (c \geqslant q)).$$

$\textsc{Trap}(j,n)$ is true if and only if $\mathbf{x}[j..j+n-1]$ is trapezoidal:

$$\textsc{Trap}(j,n) := \exists p\ \exists q\ (n = p+q)\ \wedge\ \textsc{MinUnrepSuf}(j,n,p)\ \wedge\ \textsc{MinRt}(j,n,q).$$

Finally, we can determine those n for which $\mathbf{x}$ has a trapezoidal factor of length n as follows:

$$\exists j\ \textsc{Trap}(j,n).$$

Theorem 11. *(a) There are exactly* 43 *trapezoidal factors of the Thue-Morse sequence. The longest is of length* 8.

(b) There are exactly 185 *trapezoidal factors of the Rudin-Shapiro sequence. The longest is of length* 12.

(c) There are exactly 57 *trapezoidal factors of the ordinary paperfolding sequence. The longest is of length* 8.

(d) There are exactly 77 *trapezoidal factors of the period-doubling sequence. The longest is of length* 15.

(e) Every factor of the Fibonacci word is trapezoidal.

For parts (b) and (c) above, we used the least-significant-digit first representation in order to have the computation terminate.

10 Balanced words

Our definition of balanced word above does not obviously lend itself to a definition in first-order arithmetic. However, for binary words, there is an alternative characterization (due to Coven and Hedlund [14]) that we can use: a binary word w is unbalanced if and only if there exists a palindrome v such that both $0v0$ and $1v1$ are factors of w.

Thus we can write define $\textsc{Unbal}(i,n)$, a predicate which is true if and only if $\mathbf{x}[i..i+n-1]$ is unbalanced, as follows:

$$\begin{aligned}\exists m\ (m \geqslant 2)\ \wedge\ (\exists j\ \exists k\ (\textsc{Subs}(j,i,m,n)\ \wedge\ \textsc{Subs}(k,i,m,n)\ \wedge\ \textsc{Pal}(j,m)\\ \wedge\ \textsc{Pal}(k,m)\ \wedge\ \textsc{FactorEq}(j+1,k+1,m-2)\ \wedge\ \mathbf{x}[j] \neq \mathbf{x}[k]))\end{aligned}$$

Theorem 12. *(a) The Thue-Morse word has exactly* 41 *balanced factors. The longest is of length* 8*. The Thue-Morse word has unbalanced factors of length n exactly when $n \geqslant 4$.*

(b) The Rudin-Shapiro word has exactly 157 *balanced factors. The longest is of length* 12*. The Rudin-Shapiro word has unbalanced factors of length n exactly when $n \geqslant 4$.*

(c) The ordinary paperfolding word has exactly 51 *balanced factors. The longest is of length* 8*. The ordinary paperfolding word has unbalanced factors of length n exactly when $n \geqslant 4$.*

(d) *The period-doubling word has exactly* 69 *balanced factors. The longest is of length* 15. *The period-doubling word has unbalanced factors of length* n *exactly when* $n \geqslant 6$.

(e) *All factors of the Fibonacci word are balanced.*

Of course, (e) was already well known, in much greater generality: every factor of every Sturmian word is balanced [27].

11 Consequences

As a consequence we get

Theorem 13. *Suppose* $\mathbf{x}$ *is a* k*-automatic sequence. Then*

(a) *The characteristic sequence of those* n *for which* $\mathbf{x}$ *has a closed (resp., palindromic, maximal palindromic, privileged, rich, trapezoidal, balanced) factor of length* n *is* k*-automatic.*

(b) *The sequence counting the number of closed (resp., palindromic, maximal palindromic, privileged, rich, trapezoidal, balanced) factors of length* n *is* k*-regular.*

(c) *It is decidable, given a* k*-automatic sequence, whether it contains arbitrarily long closed (resp., palindromic, maximal palindromic, privileged, rich, trapezoidal, balanced) factors.*

(d) *There exists a function* $g(k, \ell, n)$ *such that if a* k*-automatic sequence* $\mathbf{w}$ *taking values over an alphabet of size* ℓ*, generated by an* n*-state automaton, has at least one closed (resp., palindromic, maximal palindromic, privileged, rich, trapezoidal, balanced) factor, then it has a factor of length* $\leqslant g(k, \ell, n)$*. The function* g *does not depend on* $\mathbf{w}$.

(e) *There exists a function* $h(k, \ell, n)$ *such that if a* k*-automatic sequence* $\mathbf{w}$ *taking values over an alphabet of size* ℓ*, generated by an* n*-state automaton, has a closed (resp., palindromic, maximal palindromic, privileged, rich, trapezoidal, balanced) factor of length* $\geqslant h(k, \ell, n)$*, then it has arbitrarily large such factors. The function* h *does not depend on* $\mathbf{w}$.

Proof. Parts (a) and (c) follow from, for example, [30, Theorem 1]. For part (b) see [13]. Parts (d) and (e) follow from the construction converting the logical predicate for the property to an automaton. □

Acknowledgments

We thank the referee for several helpful comments. We are very grateful to Hamoon Mousavi for his public distribution of the `Walnut` software package, without which this paper could not have been written.

References

[1] J.-P. Allouche, M. Baake, J. Cassaigne, and D. Damanik. Palindrome complexity. *Theoret. Comput. Sci.*, 292:9–31, 2003.

[2] J.-P. Allouche and J. Shallit. *Automatic Sequences: Theory, Applications, Generalizations*. Cambridge University Press, 2003.

[3] G. Badkobeh, G. Fici, and Z. Lipták. On the number of closed factors in a word. In A.-H. Dediu, E. Formenti, C. Martín-Vide, and B. Truthe, editors, *Language and Automata Theory and Applications, LATA 2015*, volume 8977 of *Lecture Notes in Computer Science*, pages 381–390. Springer-Verlag, 2015.

[4] J. Berstel and C. Reutenauer. *Noncommutative Rational Series with Applications*, volume 137 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, 2010.

[5] V. Bruyère, G. Hansel, C. Michaux, and R. Villemaire. Logic and p-recognizable sets of integers. *Bull. Belgian Math. Soc.*, 1:191–238, 1994. Corrigendum, *Bull. Belg. Math. Soc.* **1** (1994), 577.

[6] M. Bucci, A. De Luca, A. Glen, and L. Q. Zamboni. A new characteristic property of rich words. *Theoret. Comput. Sci.*, 410:2860–2863, 2009.

[7] M. Bucci, A. de Luca, and A. De Luca. Rich and periodic-like words. In V. Diekert and D. Nowotka, editors, *Developments in Language Theory, DLT 2009*, volume 5583 of *Lecture Notes in Computer Science*, pages 145–155. Springer-Verlag, 2009.

[8] M. Bucci, A. De Luca, and G. Fici. Enumeration and structure of trapezoidal words. *Theoret. Comput. Sci.*, 468:12–22, 2013.

[9] J. R. Büchi. Weak secord-order arithmetic and finite automata. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 6:66–92, 1960. Reprinted in S. Mac Lane and D. Siefkes, eds., *The Collected Works of J. Richard Büchi*, Springer-Verlag, 1990, pp. 398–424.

[10] A. Carpi and V. D'Alonzo. On the repetitivity index of infinite words. *Internat. J. Algebra Comput.*, 19:145–158, 2009.

[11] A. Carpi and V. D'Alonzo. On factors of synchronized sequences. *Theoret. Comput. Sci.*, 411:3932–3937, 2010.

[12] A. Carpi and C. Maggi. On synchronized sequences and their separators. *RAIRO Inform. Théor. App.*, 35:513–524, 2001.

[13] E. Charlier, N. Rampersad, and J. Shallit. Enumeration and decidable properties of automatic sequences. *Internat. J. Found. Comp. Sci.*, 23:1035–1066, 2012.

[14] E. M. Coven and G. A. Hedlund. Sequences with minimal block growth. *Math. Systems Theory*, 7:138–153, 1973.

[15] F. D'Alessandro. A combinatorial problem on trapezoidal words. *Theoret. Comput. Sci.*, 273:11–33, 2002.

[16] A. de Luca. On the combinatorics of finite words. *Theoret. Comput. Sci.*, 218:13–39, 1999.

[17] A. de Luca, A. Glen, and L. Q. Zamboni. Rich, Sturmian, and trapezoidal words. *Theoret. Comput. Sci.*, 407:569–573, 2008.

[18] X. Droubay, J. Justin, and G. Pirillo. Episturmian words and some constructions of de Luca and Rauzy. *Theoret. Comput. Sci.*, 255:539–553, 2001.

[19] C. F. Du, H. Mousavi, L. Schaeffer, and J. Shallit. Decision algorithms for Fibonacci-automatic words, with applications to pattern avoidance. Presented at 15^{e} Journées Montoises d'Informatique Théorique, 23-26 September 2014, Nancy, France. Preprint available at arXiv:1406.0670.

[20] M. Forsyth, A. Jayakumar, J. Peltomäki, and J. Shallit. Remarks on privileged words. To appear, *Int. J. Found. Comput. Sci.*, 2015.

[21] A. Glen, J. Justin, S. Widmer, and L. Q. Zamboni. Palindromic richness. *European J. Combinatorics*, 30:510–531, 2009.

[22] D. Goč, H. Mousavi, and J. Shallit. On the number of unbordered factors. In A.-H. Dediu, C. Martín-Vide, and B. Truthe, editors, *Language and Automata Theory, LATA 2013*, volume 7810 of *Lecture Notes in Computer Science*, pages 299–310. Springer-Verlag, 2013.

[23] D. Goč, L. Schaeffer, and J. Shallit. Subword complexity and k-synchronization. In M. P. Béal and O. Carton, editors, *Developments in Language Theory, 17th International Conference, DLT 2013*, volume 7907 of *Lecture Notes in Computer Science*, pages 252–263. Springer-Verlag, 2013.

[24] B. Hodgson. Décidabilité par automate fini. *Ann. Sci. Math. Québec*, 7:39–57, 1983.

[25] T. I, S. Inenaga, H. Bannai, and M. Takeda. Counting and verifying maximal palindromes. In E. Chavez and S. Lonardi, editors, *String Processing and Information Retrieval – 17th International Symposium, SPIRE 2010*, volume 6393 of *Lecture Notes in Computer Science*, pages 135–146. Springer-Verlag, 2010.

[26] J. Kellendonk, D. Lenz, and J. Savinien. A characterization of subshifts with bounded powers. *Discrete Math.*, 313:2881–2894, 2013.

[27] M. Morse and G. A. Hedlund. Symbolic dynamics II. Sturmian trajectories. *Amer. J. Math.*, 62:1–42, 1940.

[28] J. Peltomäki. Introducing privileged words: privileged complexity of Sturmian words. *Theoret. Comput. Sci.*, 500:57–67, 2013.

[29] J. Peltomäki. Privileged factors in the Thue-Morse word — a comparison of privileged words and palindromes. *Disc. Appl. Math.*, 193:187–199, 2015.

[30] J. Shallit. Decidability and enumeration for automatic sequences: a survey. In A. A. Bulatov and A. M. Shur, editors, *CSR 2013*, volume 7913 of *Lecture Notes in Computer Science*, pages 49–63. Springer-Verlag, 2013.