FROBENIUS NUMBERS AND AUTOMATIC SEQUENCES

Jeffrey Shallit[1]
School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada
shallit@uwaterloo.ca

Received: 7/18/21, Revised: 10/24/21, Accepted: 12/7/21, Published: 12/13/21

Abstract

The Frobenius number $g(S)$ of a set S of non-negative integers with $\gcd(S) = 1$ is the largest integer not expressible as a linear combination of elements of S. Given a sequence $\mathbf{s} = (s_i)_{i\geq 0}$, we can define the associated sequence $G_{\mathbf{s}}(i) = g(\{s_i, s_{i+1}, \ldots\})$. In this paper we compute $G_{\mathbf{s}}(i)$ for some classical automatic sequences: the evil numbers, the odious numbers, and the lower and upper Wythoff sequences. In contrast with the usual methods, our proofs are based largely on automata theory and logic.

1. Introduction

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ be the natural numbers. Let S be a nonempty set of natural numbers with $\gcd(S) = 1$, possibly infinite. A classical result then says that every sufficiently large integer can be written as a linear combination of elements of S with natural number coefficients. Then $g(S)$, the *Frobenius number* of S, is defined to be the greatest integer t such that t does *not* have such a representation. For example, $g(\{6, 9, 20\}) = 43$.

The Frobenius number has received a lot of attention in recent years. For a detailed discussion of the function g, see the books of Ramírez Alfonsín [17] and Rosales and García-Sánchez [19].

Let $\mathbf{s} = (s_i)_{i\geq 0}$ be an increasing sequence of natural numbers such that

$$\gcd(s_i, s_{i+1}, \ldots) = 1$$

for all $i \geq 0$. For $i \geq 0$ define $G_{\mathbf{s}}(i) = g(\{s_i, s_{i+1}, \ldots\})$, the Frobenius number of a final segment of $\mathbf{s}$, beginning with s_i. Given a sequence $\mathbf{s}$, it can be an interesting and challenging problem to compute the Frobenius numbers $G_{\mathbf{s}}(i)$ exactly, or estimate their growth rate. Computing the Frobenius number is notoriously difficult because the problem is NP-hard [16].

[1]Supported by NSERC grant 2018-04118.

For example, for the case where $\mathbf{s}$ is $1, 4, 9, 16, \ldots$, the sequence of squares, Dutch and Rickett [8] proved that $G_{\mathbf{s}}(n) = o(n^{2+\epsilon})$ for all $\epsilon > 0$, and this bound was improved to $O(n^2)$ by Moscariello [13].

In this paper we are interested in calculating $G_{\mathbf{s}}(i)$ for some famous integer sequences $\mathbf{s}$ whose characteristic sequence is automatic. (The characteristic sequence is the binary sequence $\mathbf{b} = (b_i)_{i\geq 0}$ where $b_i = 1$ if i occurs in $\mathbf{s}$ and $b_i = 0$ otherwise.) By "automatic" we mean generated by a finite automaton in a certain way, described in the next section. In particular, we completely characterize the Frobenius number for the sequence of evil numbers, the sequence of odious numbers, and the lower and upper Wythoff sequences. Although our results are number-theoretic in nature, our approach is largely via combinatorics, automata theory, and logic.

2. Automatic and Synchronized Sequences

A *numeration system* is a method for writing a non-negative integer N as a linear combination $N = \sum_{0\leq i\leq t} a_i d_i$ of some increasing sequence $d_0 = 1 < d_1 < d_2 < \cdots$ of integers, where the a_i are chosen from a finite set that includes 0. Suppose

(a) the representation for N is always unique (up to leading zeros);

(b) the set of all valid representations forms a regular language;

(c) there is a finite automaton recognizing the triples (x, y, z) for which $x + y = z$, where the inputs to the automaton are representations of integer triples, padded with leading zeros, if necessary, to make them all of the same length.

If all three conditions hold, then we call the numeration system *regular*. A sequence $\mathbf{a} = (a_i)_{i\geq 0}$ over a finite alphabet is *automatic* if it is computed by an automaton taking the representation of i as input, starting with the most significant digit, and returning a_i as the output associated with the last state reached. For more about automatic sequences, see [3].

If a sequence is automatic, then there is a decision procedure for proving or disproving assertions about it, provided these assertions are phrased in first-order logic and using only the logical operations, comparisons of natural numbers, addition, and indexing into the sequence [7]. This decision procedure has been incorporated in a theorem-proving system called `Walnut`, written by Hamoon Mousavi [14], and it is the main tool we use to prove the results in this paper.

Automatic sequences are restricted in scope because they take their values in a finite alphabet. However, it is possible for automata to compute sequences taking their values in $\mathbb{N}$, the natural numbers, using a different meaning of "compute". We say a sequence $(a_i)_{i\geq 0}$ is *synchronized* if there is a finite automaton recognizing exactly the representations of pairs (i, a_i), read in parallel. In this case, the alphabet

of the automaton consists of *pairs* of symbols; by reading the first element of each pair we get the representation of i and by the reading the second element of each pair we get a_i. It may be necessary to pad the shorter representation with leading zeros to ensure that the two inputs have the same length.

If a sequence is synchronized, then we can compute it in linear time. If the underlying numeration system is base k, then the quantities $\limsup_i a_i/i$ and $\liminf_i a_i/i$ are computable [20]. For more about synchronized sequences, see [6, 22].

3. The Odious and Evil Numbers

Let $s_2(n)$ denote the sum of bits of n when represented in base 2. Define $t(n) = s_2(n) \bmod 2$, the famous Thue-Morse sequence [2]. The *evil numbers* $0, 3, 5, 6, 9, \ldots$ are those n with $t(n) = 0$, and the *odious numbers* $1, 2, 4, 7, \ldots$ are those n with $t(n) = 1$. (The somewhat painful terminology is from [4, p. 431].) More precisely we have [1]

$$e_n = 2n + t(n)$$
$$o_n = 2n + 1 - t(n)$$

for $n \geq 0$. Observe that both sequences have gcd 1, because

$$\gcd(e_i, e_{i+1}, e_{i+2}) = \gcd(o_i, o_{i+1}, o_{i+2}) = 1$$

for all $i \geq 0$. Additive properties of these numbers were studied previously in [15, Thm. 2].

Our first goal is to prove the following result.

Theorem 1. *Let* $\mathbf{e} = (e_0, e_1, e_2, \ldots)$ *be the sequence of evil numbers. The function* $G_{\mathbf{e}}(n)$ *is* 2*-synchronized.*

In order to prove this we need two lemmas.

Lemma 1. *If* $n \geq 1$ *can be written as the sum of four evil numbers greater than or equal to* m*, then* n *can be written as the sum of one, two, or three evil numbers greater than or equal to* m*.*

Lemma 2. *Let* $n \geq 1$ *be a non-negative integer linear combination of evil numbers greater than or equal to* m*. Then* n *can be written as the sum of either one, two, or three evil numbers greater than or equal to* m*.*

Remark 1. When we speak of such sums we never insist that the sum be of distinct integers.

Proof of Lemma 1. This assertion can be phrased as a first-order formula, namely,

$$\forall m, n \ \ \mathrm{evil}_4(m,n) \Longrightarrow \mathrm{evil}_{1,2,3}(m,n),$$

where

$$\begin{aligned}\mathrm{evil}_{1,2,3}(m,n) := \exists j,k,\ell \ t(j) = t(k) = t(\ell) = 0 \ \wedge \ j,k,\ell \geq m \wedge \\ (n = j \ \vee \ n = j+k \ \vee \ n = j+k+\ell) \\ \mathrm{evil}_4(m,n) := \exists i,j,k,\ell \ t(i) = t(j) = t(k) = t(\ell) = 0 \ \wedge \ i,j,k,\ell \geq m \wedge \\ n = i+j+k+\ell.\end{aligned}$$

To prove this, we use the theorem-proving software called `Walnut`, where we can simply translate the statement of the previous paragraph into `Walnut`'s syntax and evaluate it [14].

```
def evil123rep "Ej,k,l (T[j]=@0) & (T[k]=@0) & (T[l]=@0) &
     j>=m & k>=m & l>=m & (n=j | n=j+k | n=j+k+l)":

def evil4rep "Ei,j,k,l (T[i]=@0) & (T[j]=@0) & (T[k]=@0) &
(T[l]=@0) & i >= m & j>=m & k>=m & l>=m & (n=j+k+l+m)":

eval evilcheck "Am,n $evil4rep(m,n) => $evil123rep(m,n)":
```

This last line returns `TRUE`, so the lemma is proved. □

Remark 2. To help in understanding the syntax of `Walnut`, we note the following:

- `E` represents $\exists$
- `A` represents $\forall$
- `T` represents the Thue-Morse sequence
- the natural number constant i is written `@i`
- `&` represents logical "and"
- `|` represents logical "or"
- `~` represents logical negation
- `=>` is logical implication
- `def` defines a formula for future use
- `eval` evaluates a logical formula and returns `TRUE` or `FALSE`.

See [14] for more information.

Proof of Lemma 2. Let n be written as a non-negative integer linear combination of evil numbers greater than or equal to m. Without loss of generality choose a representation for n that minimizes s, the sum of the coefficients. If this sum is at least 4, we can write $n = u + v$ where u is the sum of 4 evil numbers and v is the sum of $s - 4$ evil numbers, all greater than or equal to m. But then by Lemma 1, we can write u as the sum of 3 evil numbers greater than or equal to m, so n is the sum of $s - 1$ evil numbers greater than or equal to m, a contradiction. So s is no more than 3, as desired. □

We can now prove Theorem 1.

Proof. It suffices to give a first-order definition of $G_{\mathbf{e}}(n)$. We can do this as follows:

```
def evilg "(Aj (j>n) => $evil123rep(2*m,j)) & ~$evil123rep(2*m,n)":
```

This gives a 58-state synchronized automaton computing $G_{\mathbf{e}}(n)$, which we omit because it is too large to display compactly. □

Now that we have a synchronized automaton, we can determine the asymptotic behavior of $G_{\mathbf{e}}(n)$.

Theorem 2. *We have $4m \leq G_{\mathbf{e}}(m) \leq 6m + 7$ for all $m \geq 0$. These bounds are optimal, because they are attained for infinitely many m.*

Proof. We run the following `Walnut` commands, which all evaluate to `TRUE`.

```
eval upperb "Am,n $evilg(m,n) => n <= 6*m+7":
eval upperopt "Ai Em,n (m>i) & $evilg(m,n) & n=6*m+7":
eval lowerb "Am,n $evilg(m,n) => n >= 4*m":
eval loweropt "Ai Em,n (m>i) & $evilg(m,n) & n=4*m":
```

□

Corollary 1. *We have*

$$\inf_{i\geq 1} G_{\mathbf{e}}(i)/i = 4 \qquad \sup_{i\geq 1} G_{\mathbf{e}}(i)/i = 7$$
$$\liminf_{i\geq 1} G_{\mathbf{e}}(i)/i = 4 \qquad \limsup_{i\geq 1} G_{\mathbf{e}}(i)/i = 6.$$

The sequence $(G_{\mathbf{e}}(i))_{i\geq 0}$ has a rather erratic behavior. In particular we can prove the following result.

Theorem 3.

(a) The difference $G_{\mathbf{e}}(i+1) - G_{\mathbf{e}}(i)$ can be arbitrarily large.

(b) There are arbitrarily long blocks of indices on which $G_{\mathbf{e}}(i)$ is constant.

OEIS number	n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A001969	e_n	0	3	5	6	9	10	12	15	17	18	20	23	24	27	29	30
A342581	$G_{\mathbf{e}}(n)$	7	7	13	14	16	31	31	31	32	55	55	55	55	55	61	62
A000069	o_n	1	2	4	7	8	11	13	14	16	19	21	22	25	26	28	31
A342579	$G_{\mathbf{o}}(n)$	−1	5	10	17	23	23	24	34	39	39	45	46	71	71	71	71

Table 1: The evil and odious sequences and their Frobenius numbers.

Proof. We use the following `Walnut` code.

```
eval evilgdiff "Ai Ej,m,n1,n2 (j>=i) & $evilg(m,n1) &
   $evilg(m+1,n2) & n2=n1+j":
eval evalmonotone "Ai Ej,m,u (j>=i) & $evilg(m,u) &
   (At,v ((t>m) & (t<m+j) & $evilg(t,v)) => u=v)":
```

Both return `TRUE`. □

We can carry out exactly the same analysis for the odious numbers. The analogues of Theorem 1 and Lemma 1 and 2 all hold. Here are the results.

Theorem 4. *Let* $\mathbf{o} = (o_0, o_1, o_2, \ldots)$ *be the sequence of odious numbers. We have* $4m \leq G_{\mathbf{o}}(m) \leq 6m - 1$ *for all* $m \geq 1$*. These bounds are optimal, because they are attained for infinitely many* m*.*

Corollary 2. *We have*

$$\inf_{i\geq 1} G_{\mathbf{o}}(i)/i = 4 \qquad \sup_{i\geq 1} G_{\mathbf{o}}(i)/i = 6$$

$$\liminf_{i\geq 1} G_{\mathbf{o}}(i)/i = 4 \qquad \limsup_{i\geq 1} G_{\mathbf{o}}(i)/i = 6.$$

Furthermore, the analogue of Theorem 3 also holds.

The first few terms of the sequences we have discussed in this section, together with their numbers from the *On-Line Encyclopedia of Integer Sequences* (OEIS) [23], are given in Table 1.

4. Results for the Wythoff sequences

We can carry out a similar analysis for the lower and upper Wythoff sequences, defined as follows. Here we find substantially different behavior than for the odious and evil numbers.

Let $\varphi = (1+\sqrt{5})/2$, the golden ratio. As usual we write $\lfloor x \rfloor$ for the greatest integer less than or equal to x. Define

$$L_n = \lfloor n\varphi \rfloor$$
$$U_n = \lfloor n\varphi^2 \rfloor$$

for $n \geq 0$. Here, instead of base-2 representation, all numbers are represented in Fibonacci representation (also called Zeckendorf representation) [12, 26]. In this representation a number is represented as a linear combination $\sum_{2\leq i\leq t} a_i F_i$ with $a_i \in \{0,1\}$ and subject to the condition that $a_i a_{i+1} = 0$. We then define $(n)_F = a_t a_{t-1} \cdots a_2$ and $[x]_F = \sum_{1\leq i\leq t} a_i F_{t+2-i}$ if $x = a_1 \cdots a_t$. For example, $(7)_F = 1010$ and $[1010]_F = 7$.

The additive properties of the upper and lower Wythoff sequences were studied previously in [11, 21]. Also see [25].

Theorem 5. *The functions L_n and U_n are Fibonacci synchronized.*

Proof. We start by showing that the function $n+1$ is Fibonacci synchronized. We can construct an automaton $\mathrm{inc}(x,y)$ that computes the relation $y = x+1$ for x and y in Fibonacci representation. Using the following easily-proven identities,

(a) $[x00(10)^i]_F + 1 = [x010^{2i}]_F$; and

(b) $[x0(01)^i]_F + 1 = [x010^{2i-1}]_F$,

we can obtain the incrementer depicted in Figure 1.

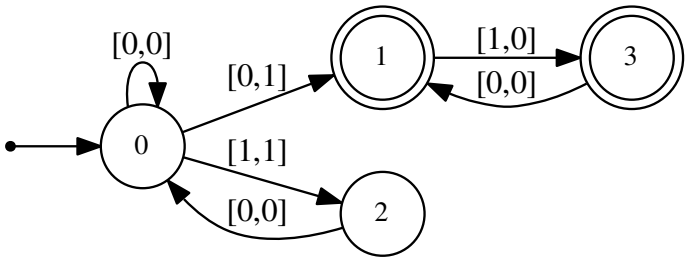

Figure 1: Incrementer for Fibonacci representation

As an example, this automaton accepts the input $[0,1][1,0][0,0][1,0]$. Here the first components spell out 0101, which represents 4 in Fibonacci representation, and

the second components spell out 1000, which represents 5 in Fibonacci representation.

Next, we use the identities

$$[(n)_F 0]_F = \lfloor (n+1)\varphi \rfloor - 1$$
$$[(n)_F 01]_F = \lfloor (n+1)\varphi^2 \rfloor - 1$$

for $n \geq 0$, whose proof can be found, for example, in [18]. Substituting $n-1$ for n, this gives us formulas for $\lfloor n\varphi \rfloor$ and $\lfloor n\varphi^2 \rfloor$ in terms of shifting and incrementation in the Fibonacci representation. Shifting can be carried out using the finite automaton in Figure 2.

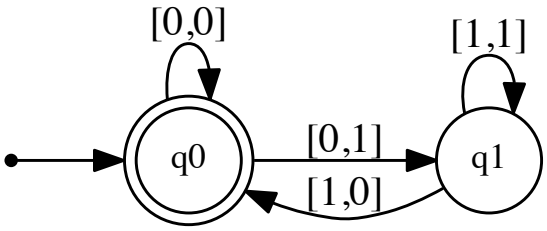

Figure 2: Synchronized automaton for the shift.

So we get synchronized automata computing L_n and U_n as follows:

```
def lower "?msd_fib ((s=0)&(n=0)) | Et,u $fibinc(u,n) & $shift(u,t)
     & $fibinc(t,s)":
def upper "?msd_fib ((s=0)&(n=0)) | Et,u,v,w $fibinc(u,n)
     & $shift(u,t) & $shift(t,v) & $fibinc(v,w) & $fibinc(w,s)":
```

with the automata depicted in Figure 3. □

Let $\mathbf{L} = (L_0, L_1, L_2, \ldots)$ denote the lower Wythoff sequence. Our next goal is to prove that $G_{\mathbf{L}}(n)$ is Fibonacci synchronized (that is, there is an automaton recognizing the pairs $(n, G_{\mathbf{L}}(n))$ represented in the Fibonacci numeration system). We start with some lemmas. Recall that $\mathbf{f} = (f_n)_{n\geq 0} = 01001010\cdots$ is the Fibonacci word [5], the fixed point of the morphism $0 \to 01$, $1 \to 0$.

Lemma 3. *Let n be a positive integer. Then n is a member of the lower Wythoff sequence $(L_n)_{n\geq 1}$ if and only if $f_{n-1} = 0$, and n is a member of the upper Wythoff sequence $(U_n)_{n\geq 1}$ if and only if $f_{n-1} = 1$.*

Proof. See [22]. □

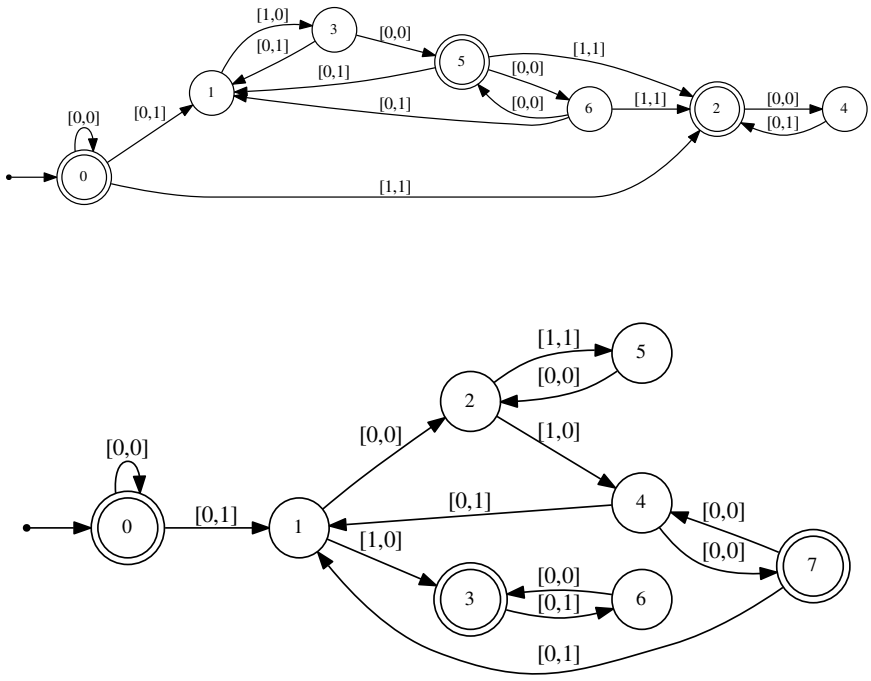

Figure 3: Synchronized automata for $L_n = \lfloor n\varphi \rfloor$ (top) and $U_n = \lfloor n\varphi^2 \rfloor$ (bottom).

Lemma 4. *Let $n \geq 1$ be a non-negative integer linear combination of lower Wythoff numbers greater than or equal to m. Then n can be written as the sum of either one or two lower Wythoff numbers greater than or equal to m.*

Proof. It can be carried out with `Walnut` in analogy with the proof of Lemma 2. Here `F[n]` is `Walnut`'s way of expressing f_n, the n'th bit of the Fibonacci word.

```
def lower12rep "?msd_fib Ej,k (F[j-1]=@0) & (F[k-1]=@0) &
     j>=m & k>=m & (n=j|n=j+k)":
def lower3rep "?msd_fib Ej,k,l (F[j-1]=@0) & (F[k-1]=@0) &
     (F[l-1]=@0) & j>=m & k>=m & l>=m & n=j+k+l":
eval lowercheck "?msd_fib Am An $lower3rep(m,n) => $lower12rep(m,n)":
```

□

We can now prove the result announced above.

Theorem 6. *The function $G_{\mathbf{L}}(n)$ is Fibonacci synchronized.*

Proof. We use the `Walnut` commands:

```
def lowerunrep "?msd_fib (Aj (j>n) => $lower12rep(m,j)) &
      ~$lower12rep(m,n)":
def lowerg "?msd_fib Et $lower(m,t) & $lowerunrep(t,n)":
```

This last command gives us a synchronized automaton with 24 states computing $G_{\mathbf{L}}(n)$, depicted in Figure 4. □

We can now use this automaton to determine the behavior of $G_{\mathbf{L}}(n)$.

Theorem 7. *We have $-3 \leq G_{\mathbf{L}}(n) - 2L_n \leq 1$, and the upper and lower bounds are achieved infinitely often.*

Proof. We use the following `Walnut` commands:

```
eval lowerb1 "?msd_fib Am En,r $lowerg(m,n) & $lower(m,r) & n+3>=2*r":
eval lowerb2 "?msd_fib Am En,r $lowerg(m,n) & $lower(m,r) & n<=2*r+1":
eval lowerbinf1 "?msd_fib As Em,n,r (m>=s) & $lowerg(m,n) &
     $lower(m,r) & n+3>=2*r":
eval lowerbinf2 "?msd_fib As Em,n,r (m>=s) & $lowerg(m,n) &
     $lower(m,r) & n<=2*r+1":
```

and `Walnut` returns `TRUE` for all of them. □

Corollary 3. *We have $\lim_{n\to\infty} G_{\mathbf{L}}(n)/n = 1 + \sqrt{5}$.*

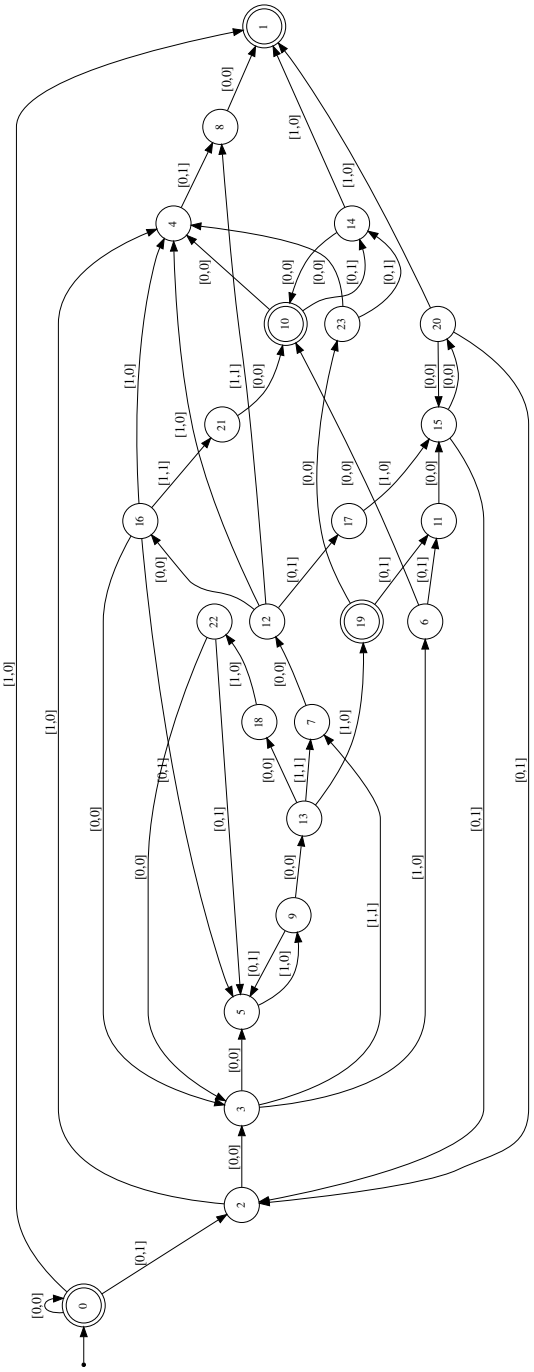

Figure 4: Fibonacci automaton computing $G_{\mathbf{L}}(n)$

Theorem 8. *We have* $G_{\mathbf{L}}(n+1) - G_{\mathbf{L}}(n) \in \{0, 2, 3, 5, 6, 8\}$*, and furthermore each difference occurs infinitely often.*

Proof. We use the following `Walnut` commands:

```
eval lowerdiff "?msd_fib Am Eu,v $lowerg(m,u) & $lowerg(m+1,v)
      & (v=u|v=u+2|v=u+3|v=u+5|v=u+6|v=u+8)":
def ldi "?msd_fib Am Et,u,v (t>=m) & $lowerg(t,u)
      & $lowerg(t+1,v) & v=u+d":
eval lowerdiffinfcheck "?msd_fib $ldi(0) & $ldi(2) & $ldi(3) & $ldi(5)
      & $ldi(6) & $ldi(8)":
```

and `Walnut` returns `TRUE` twice. □

Theorem 9. *There exists a Fibonacci automaton of* 11 *states computing the first difference* $G_{\mathbf{L}}(n+1) - G_{\mathbf{L}}(n)$*.*

The automaton is depicted in Figure 5.

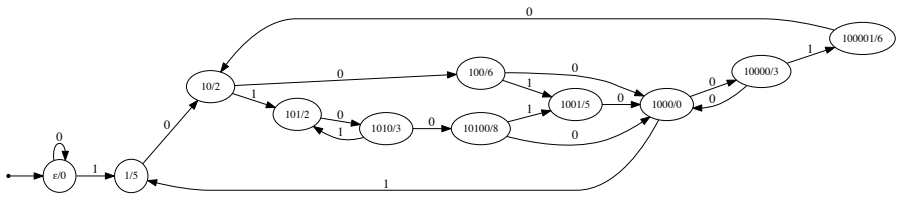

Figure 5: Fibonacci automaton computing $G_{\mathbf{L}}(n+1) - G_{\mathbf{L}}(n)$

Now we turn to the upper Wythoff sequence. The results are completely analogous to the results for the lower Wythoff sequence, and the proofs are also analogous. We omit the details.

Lemma 5. *Let* $n \geq 1$ *be a non-negative integer linear combination of upper Wythoff numbers greater than or equal to* m*, for* $m \geq 3$*. Then* n *can be written as the sum of either one, two, or three Wythoff numbers greater than or equal to* m*.*

Remark 3. Lemma 5 fails for $m = 2$ because 8 is not the sum of one, two, or three Wythoff numbers greater than or equal to 2, while it is the sum of four (since $8 = 2 + 2 + 2 + 2$).

Theorem 10. *Let* $\mathbf{U} = (U_0, U_1, U_2, \ldots)$ *denote the upper Wythoff sequence. The function* $G_{\mathbf{U}}(n)$ *is Fibonacci synchronized.*

OEIS number	n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A000201	L_n	0	1	3	4	6	8	9	11	12	14	16	17	19	21	22
A342715	$G_{\mathbf{L}}(n)$	−1	−1	5	7	13	15	15	20	23	26	31	31	39	41	41
A001950	U_n	0	2	5	7	10	13	15	18	20	23	26	28	31	34	36
A342716	$G_{\mathbf{U}}(n)$	−1	3	16	19	42	42	42	55	58	76	79	79	110	110	110

Table 2: The lower and upper Wythoff sequences and their Frobenius numbers.

Theorem 11. *We have* $-5 \le G_{\mathbf{U}}(n) - 3U_n \le 20$*, and these upper and lower bounds are achieved infinitely often.*

Corollary 4. *We have* $\lim_{n\to\infty} G_{\mathbf{U}}(n)/n = (9 + 3\sqrt{5})/2$.

Theorem 12. *We have* $G_{\mathbf{U}}(n+1) - G_{\mathbf{U}}(n) \in \{0, 3, 5, 8, 11, 13, 18, 21, 23, 26, 31\}$ *for* $n \ge 1$*, and furthermore each difference occurs infinitely often.*

Theorem 13. *There exists a Fibonacci automaton computing the first difference* $G_{\mathbf{U}}(n+1) - G_{\mathbf{U}}(n)$.

The first few terms of the sequences we have discussed in this section, together with their numbers from the OEIS, are given in Table 2.

5. A Counterexample Sequence

In all of the examples we have seen so far, if a sequence had automatic characteristic sequence, then the characteristic sequence of the associated Frobenius sequence was also automatic. It is natural to conjecture this might always be the case. However, we now prove the following result.

Theorem 14. *Let* $s_i = 2^i + 1$ *for* $i \ge 0$ *and* $\mathbf{s} = (s_i)_{i\ge 0}$*. Then* $G_{\mathbf{s}}(i) = 2^{2i} + 2^i + 1$ *for* $i \ge 1$.

Proof. It suffices to prove that $2^{2i} + 2^i + 1$ cannot be written as a non-negative integer linear combination of $2^i + 1, 2^{i+1} + 1, \ldots 2^{2i} + 1$, while every larger integer can be so expressed.

Suppose $2^{2i} + 2^i + 1 = a_0(2^i + 1) + a_1(2^{i+1} + 1) + \cdots + a_i(2^{2i} + 1)$ with $a_0, \ldots, a_i$ non-negative integers. Considering both sides modulo 2^i, we see the left-hand side is 1, while the right-hand side is $a_0 + a_1 + \cdots + a_i$. So either $a_0 + a_1 + \cdots + a_i = 1$ or $a_0 + a_1 + \cdots + a_i \ge 2^i + 1$. In the former case we would have $2^{2i} + 2^i + 1 = 2^j + 1$ for some j, $i \le j \le 2i$, which is clearly impossible. In the latter case we would have

$$\begin{aligned} 2^{2i} + 2^i + 1 &= a_0(2^i + 1) + a_1(2^{i+1} + 1) + \cdots + a_i(2^{2i} + 1) \\ &\ge (a_0 + a_1 + \cdots + a_i)(2^i + 1) \ge (2^i + 1)^2, \end{aligned}$$

which is also impossible. This shows $2^{2i} + 2^i + 1$ is not representable.

We now argue that if $2^{2i}+2^i+1 < x \leq 2^{2i}+2^{i+1}+2$, then x has a representation. This suffices to show that all $x > 2^{2i} + 2^i + 1$ are representable, because this range contains $2^i + 1$ consecutive integers, and any $x > 2^{2i} + 2^{i+1} + 2$ can then be represented by adding the appropriate multiple of $2^i + 1$.

Given a particular linear combination

$$x = a_0(2^i + 1) + a_1(2^{i+1} + 1) + \cdots + a_i(2^{2i} + 1),$$

call its *weight* $a_0 + \cdots + a_i$. We now repeat the following transformation: given a linear combination $x = a_0(2^i + 1) + \cdots + a_i(2^{2i} + 1)$, find the largest nonzero a_j in the combination. Then form the linear combination of $x + 1$ by adding $2 \cdot (2^{j-1} + 1) - (2^j + 1) = 1$ to the representation for x. Doing so increases the weight of the linear combination by 1, because we add 2 to one coefficient and subtract 1 from another.

Now let us start the process with the number $(2^{2i} + 1) + (2^i + 1)$, which has a representation of weight 2. When we carry out the transformation of the previous paragraph once, the 1 coefficient of $2^{2i}+1$ in the linear combination disappears and a 2 appears as the coefficient of $2^{2i-1} + 1$. Doing it twice more causes this 2 to disappear, and a 4 appears as the coefficient of $2^{2i-2} + 1$. This process continues for a total of $1 + 2 + 4 + \cdots + 2^{i-1} = 2^i - 1$ times, eventually resulting in the representation $2^{2i}+2^i+2+2^i-1 = 2^{2i}+2^{i+1}+1$ as $(2^i+1)(2^i+1)$ of weight 2^i+1. Finally, $2^{2i} + 2^{i+1} + 2$ has the representation $(2^{2i} + 1) + (2^{i+1} + 1)$. This gives us $2^i + 1$ consecutive representable numbers, as desired, and completes the proof. □

We have now shown that $G_{\mathbf{s}}(i) = 2^{2i} + 2^i + 1$ for $i \geq 1$. Hence we get our desired counterexample: the characteristic sequence of $(2^i + 1)_{i\geq 0}$ is automatic, as the set of its base-2 representations is specified by the regular expression 10^*1. But the characteristic sequence of $(G_{\mathbf{s}}(i))_{i\geq 1} = (2^{2i} + 2^i + 1)_{i\geq 1}$ is not automatic, as the set of its base-2 representations is of the form $\{10^i10^i1 : i \geq 0\}$, which can easily be seen to be non-regular using a standard tool from formal language theory called the pumping lemma [10, Lemma 3.1].

Remark 4. Theorem 14, in more generality, was stated in a recent paper of Song [24]. However, the proof was omitted there, so we give it here.

6. Concluding remarks

We conjecture that the analogue of Corollary 3 holds for all Beatty sequences.

For other results of additive number theory based on automata theory, see [15].

All the `Walnut` code we used is available from the author's website, https://cs.uwaterloo.ca/~shallit/papers.html. For `Walnut` itself, see https://cs.

uwaterloo.ca/~shallit/walnut.html.

I am grateful to the referee for pointing out several mistakes in the original draft.

References

[1] J.-P. Allouche, B. Cloitre, and V. Shevelev, Beyond odious and evil, *Aequationes Math.* **90** (2016), 341–353.

[2] J.-P. Allouche and J. Shallit, The ubiquitous Prouhet-Thue-Morse sequence, in C. Ding, T. Helleseth, and H. Niederreiter, editors, *Sequences and Their Applications, Proceedings of SETA '98*, pp. 1–16, Springer-Verlag, London, 1999.

[3] J.-P. Allouche and J. Shallit, *Automatic Sequences*, Cambridge University Press, Cambridge, 2003.

[4] E. R. Berlekamp, J. H. Conway, and R. K. Guy, *Winning Ways for Your Mathematical Plays, Vol. 2: Games in Particular*, Academic Press, London, 1982.

[5] J. Berstel, Mots de Fibonacci, *Séminaire d'Informatique Théorique, LITP* **6-7** (1980–81), 57–78.

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

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

[8] K. Dutch and C. Rickett, Conductors for sets of large integer squares, *Notes on Number Theory and Discrete Mathematics* **18** (1) (2012), 16–21.

[9] P. Erdős and R. L. Graham, On a linear diophantine problem of Frobenius, *Acta Arith.* **21** (1972), 399–408.

[10] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Mass., 1979.

[11] S. Kawsumarng, T. Khemaratchatakumthorn, P. Noppakaew, and P. Pongsriiam, Sumsets associated with Wythoff sequences and Fibonacci numbers, *Period. Math. Hung.* **82** (2021), 98–113.

[12] C. G. Lekkerkerker, Voorstelling van natuurlijke getallen door een som van getallen van Fibonacci, *Simon Stevin* **29** (1952), 190–195.

[13] A. Moscariello, On integers which are representable as sums of large squares, *Intl. J. Number Theory* **11** (2015), 2505–2511.

[14] H. Mousavi, Automatic theorem proving in `Walnut`, Arxiv preprint arXiv:1603.06017 [cs.FL], May 25 2021, available at https://arxiv.org/abs/1603.06017.

[15] A. Rajasekaran, J. Shallit, and T. Smith, Additive number theory via automata theory, *Theor. Comput. Sys.* **64** (2020), 542–567.

[16] J. L. Ramírez Alfonsín, Complexity of the Frobenius problem, *Combinatorica* **16** (1996), 143–147.

[17] J. L. Ramírez Alfonsín, *The Diophantine Frobenius Problem*, Vol. 30 of *Oxford Lecture Series in Mathematics and its Applications*, Oxford University Press, Oxford, 2005.

[18] D. Reble, Zeckendorf vs. Wythoff representations: comments on A007895, manuscript available at https://oeis.org/A007895/a007895.pdf, 2008.

[19] J. C. Rosales and P. A. García-Sánchez, *Numerical Semigroups*, Springer, New York, 2009.

[20] L. Schaeffer and J. Shallit, The critical exponent is computable for automatic sequences, *Internat. J. Found. Comp. Sci.* **23** (2012), 1611–1626.

[21] J. Shallit, Sumsets of Wythoff sequences, Fibonacci representation, and beyond, to appear, *Period. Math. Hung.*, 2021, preprint at https://arxiv.org/abs/2006.04177.

[22] J. Shallit, Synchronized sequences, in T. Lecroq and S. Puzynina, eds., *WORDS 2021*, Lect. Notes in Comput. Sci., Vol. 12847, Springer, Cham, 2021, pp. 1–19.

[23] N. J. A. Sloane et al., *The On-Line Encyclopedia of Integer Sequences*, 2021, available at https://oeis.org .

[24] K. Song, The Frobenius problem for numerical semigroups generated by the Thabit numbers of the first, second kind base b and the Cunningham numbers, *Bull. Korean Math. Soc.* **57** (2020), 623–647.

[25] J. Steuding and P. Stumpf, On the Frobenius problem for Beatty sequences, *Indag. Math.* **28** (2017), 132–137.

[26] E. Zeckendorf, Représentation des nombres naturels par une somme de nombres de Fibonacci ou de nombres de Lucas, *Bull. Soc. Roy. Liège* **41** (1972), 179–182.