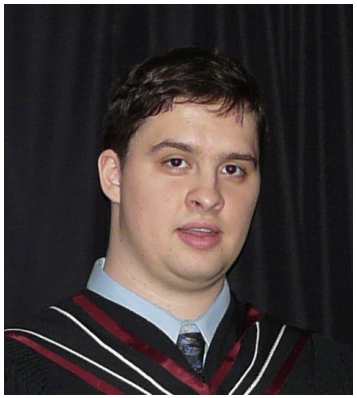
The Logical Approach to Automatic Sequences

Part 1: Automatic Sequences and k-Regular Sequences

Jeffrey Shallit
School of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1
Canada
shallit@cs.uwaterloo.ca
https://cs.uwaterloo.ca/~shallit

Some of my co-authors


Luke Schaeffer


Hamoon Mousavi


Narad Rampersad


Daniel Goč


Chen Fei Du


Émilie Charlier


Eric Rowland


Jean-Paul Allouche

This Short Course — Summarized

1. A decision procedure for **proving** theorems about a large class of interesting sequences exists (and handles many famous sequences such as Thue-Morse, Rudin-Shapiro, etc.), based on first-order logic
2. Many properties that have long been studied in the literature can be phrased in first-order logic (including some for which this is not obvious!)
3. The decision procedure is relatively easy to implement and often runs remarkably quickly, despite its formidable worst-case complexity — and we have an implementation that is publicly available (`Walnut`)
4. The method can also be used to not simply decide, but also *enumerate*, many aspects of sequences

Seven Points of the Talk

5. Many results already in the literature (in dozens of papers and Ph. D. theses) can be reproved by our program in a matter of seconds (including fixing at least one that was wrong!)
6. Many new results can be proved, such as avoidability of the pattern xxx^R
7. However, there are some well-defined limits to what we can do with the method because either
   - the property is not expressible in first-order logic; or
   - the underlying sequence leads to undecidability

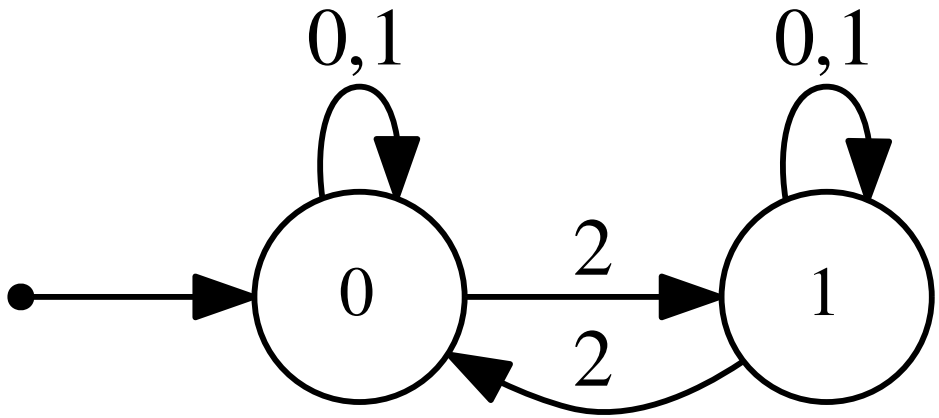
What is an automatic sequence?

- Also called *k-recognizable* in the literature.
- An automatic sequence is an infinite sequence (sometimes called an infinite word or infinite string)

$$\mathbf{a} = a_0 a_1 a_2 \cdots$$

over a finite alphabet of letters, generated by a finite-state machine (automaton)
- The automaton, given an integer n as input, computes a_n as follows:
  - n is represented in some fixed integer base $k \geq 2$
  - The automaton moves from state to state according to this input and its internal transition table
  - Each state has an output letter associated with it
  - The output on input n is the output associated with the last state reached
- The notion is base-dependent, so we speak of k-automatic sequences.

An example: the "Mephisto Waltz" automaton



This 3-automaton generates the "Mephisto Waltz sequence"

$$\mathbf{w} = (w_n)_{n \geq 0}$$

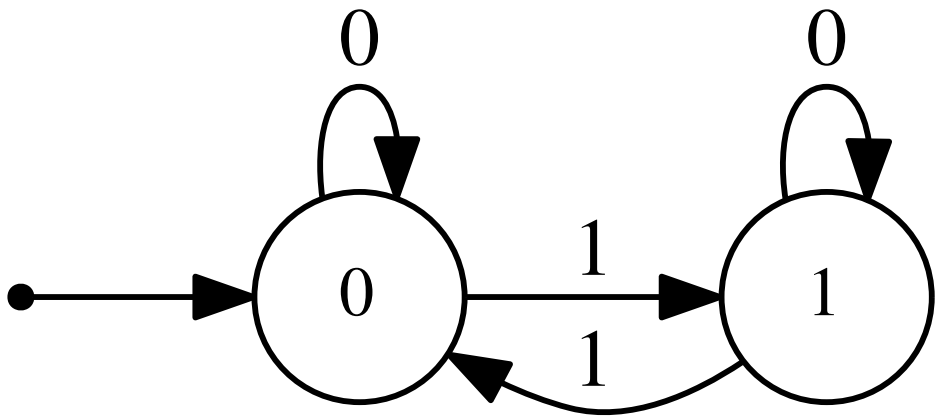
n	0	1	2	3	4	5	6	7	8	9	10	11	12	13
$\mathbf{w}[n]$	0	0	1	0	0	1	1	1	0	0	0	1	0	0

Here the input is in base 3, and the output is the name of the state.

Some obvious questions to ask

- Is the Mephisto Waltz sequence **w** ultimately periodic?
- Does it ever have two consecutive identical blocks? Three? Four? What are their sizes? Where do they appear?
- Does it have infinitely many distinct palindromes?
- How many distinct blocks of length n are there?
- Does every block that occurs, occur infinitely often?
- What is the maximum distance between two consecutive occurrences of the same block?

Another example: the Thue-Morse sequence



This automaton generates the Thue-Morse sequence

$$\mathbf{t} = (t_n)_{n \geq 0} = 0110100110010110 \cdots .$$

Here input is in base 2, and the output is the name of the state.

Appearances of the Thue-Morse sequence

- The famous Tarry-Escott problem asks to find two disjoint sets of positive integers S and T such that

$$\sum_{s \in S} s^i = \sum_{t \in T} t^i$$

for $i = 0, 1, \ldots, n-1$.

- For example, for $n = 3$ one solution is

$$0^i + 3^i + 5^i + 6^i = 1^i + 2^i + 4^i + 7^i$$

for $i = 0, 1, 2$.

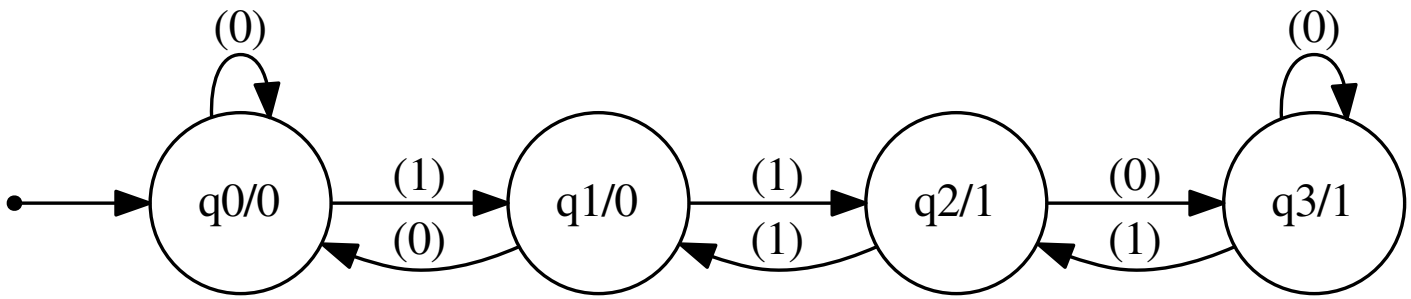
- If we define $S_n = \{0 \leq i < 2^n \ : \ t_i = 0\}$ and $T_n = \{0 \leq i < 2^n \ : \ t_i = 1\}$, we get a solution.

Appearances of the Thue-Morse sequence

- The Cooper-Dutle fair dueling problem: given two shooters Alice and Bob firing at each other, what firing order results when the order is chosen greedily to make probability in prefixes as close as possible to each other?
- Example: $p = 1/3$: Alice goes first, wins with probability $1/3$. Bob goes next, wins with probability $(2/3)(1/3) = 2/9$. This is less than $1/3$, so Bob goes again, now wins with probability $2/9 + (2/3)^2(1/3) = 10/27$. This is more than Alice's probability, so Alice goes next.
- Continuing, the sequence is $ABBABABB\cdots$
- As $p \to 0$, the sequence goes to the Thue-Morse sequence

Another example: the Rudin-Shapiro sequence

- The *Rudin-Shapiro sequence* counts the parity of the number of 11's in the binary expansion of n.
- It is generated by the following automaton, where the states record the parity of the number of 11's seen so far, and whether the last character seen was a 1.



Why study automatic sequences?

- They are a nontrivial class of self-similar sequences
- They appear in many different areas of mathematics, computer science, theoretical physics, etc.
- Many "naturally-occurring" sequences are automatic
- Halfway between periodic and chaotic
- The class is invariant under many natural kinds of transformations (shifts, linearly-indexed subsequences, block compression, etc.)
- They provide canonical examples for various kinds of avoidance problems

What we're going to see in this course

Claim. Many properties of automatic sequences can be proved purely mechanically, by a machine computation that implements a decision procedure.

Often, there is no longer any need for long case-based arguments that are prone to error.

To illustrate the complexity of "traditional" case-based methods, let's prove that the Thue-Morse sequence $\mathbf{t}$ has no overlaps.

(An *overlap* is a block of the form $axaxa$, where a is a single letter and x is a block, like the English word `alfalfa`.)

First, we need some basics on words and languages.

Basics on words (strings)

- $|x|$: length of the word x
- $|x|_a$: number of occurrences of the letter a in the word x
- ϵ: empty word, of length 0
- xy: concatenation of two words; ϵ is identity element
- $x^n = \overbrace{xx\cdots x}^{n}$: power of a word x: $(\mathtt{ma})^2 = \mathtt{mama}$
- x^R: reversal of word x: $(\mathtt{drawer})^R = \mathtt{reward}$
- $x[i]$: i'th letter of word x (usually indexed starting at 0)

Basics on words

- If $z = wxy$, then
  - w is a *prefix* of z
  - y is a *suffix* of z
  - x is a *factor* of z (also sometimes called *subword*)
- Prefixes, suffixes, and factors of z are *proper* if they do not equal z.
- $x[a..b]$: the factor starting at position a and ending at position b of x

Basics on words

- Two words x and y are *conjugates* if y is a cyclic shift of x; alternatively if there exist words u, v such that $x = uv$ and $y = vu$: `listen` and `enlist` are conjugates
- w^ω: the infinite word (or infinite sequence) $www\cdots$
- An infinite word $\mathbf{x}$ is *ultimately periodic* if there exist finite words y, z such that $\mathbf{x} = yz^\omega$.
- Morphism: map h from words to words satisfying $h(xy) = h(x)h(y)$ for all words x, y.
- Morphisms can be defined by their action on single letters; e.g., the Thue-Morse morphism μ defined by $\mu(0) = 01$ and $\mu(1) = 10$

Basics on languages

- language L: a set of words
- $|L|$: cardinality of language L
- $L_1 L_2$: $\{x_1 x_2 \ : \ x_1 \in L_1, x_2 \in L_2\}$
- L^n: $\overbrace{LL\cdots L}^{n}$
- L^*: $\bigcup_{n\geq 0} L^n$
- L^ω: language of infinite words defined by $\{x_1 x_2 \cdots \ : \ x_i \in L\}$

Number representations

- represent integers in an integer base $k \geq 2$ using the alphabet $\Sigma_k = \{0, 1, \ldots, k-1\}$.
- for example, 101011 is the representation of 43 in base 2.
- We write $(43)_2 = 101011$; this is a map from $\mathbb{N}$ to Σ_k^*.
- The representation of 0 is ϵ, the empty word.
- Similarly, we can define a map from Σ_k^* to $\mathbb{N}$ by

$$[a_1 a_2 \cdots a_n]_k = \sum_{1 \leq i \leq n} a_i k^{n-i}.$$

- So $[000101011]_2 = 43$.

Representations of $\mathbb{N}^n$

- represent n-tuples of integers as words over the alphabet Σ_k^n, padding with leading zeroes, if necessary.
- For example, the pair $(21, 7)$ can be represented in base 2 by the word

$$x = (21, 7)_2 = [1, 0][0, 0][1, 1][0, 1][1, 1].$$

- Projections: $\pi_1(x) = 10101$ and $\pi_2(x) = 00111$.

The Thue-Morse sequence has no overlaps

We define $\overline{0} = 1$ and $\overline{1} = 0$.

Lemma. Let μ be the *Thue-Morse morphism* defined by $\mu(0) = 01$ and $\mu(1) = 10$. Then $\mu(t_n) = t_{2n}t_{2n+1}$.

Proof. By induction on n.

The base case is $n = 0$. Then we have $\mu(t_0) = \mu(0) = 01 = t_0t_1$.

For the induction step, assume the result is true for $i < n$; we prove it for $i = n$.

Now the binary expansion of $2n$ is the same as that for n, except with an extra 0 on the end. The binary expansion of $2n + 1$ is that same as that for n, except with an extra 1 on the end. Hence $t_{2n} = t_n$ and $t_{2n+1} = \overline{t_n}$.

It follows that $\mu(t_n) = t_n\ \overline{t_n} = t_{2n}t_{2n+1}$.

The Thue-Morse sequence has no overlaps

Note that this lemma implies that

- If $t_r = a$ and r is even, then $t_{r+1} = \overline{a}$;
- If $t_r = a$ and r is odd, then $t_{r-1} = \overline{a}$.

Theorem. The Thue-Morse word $\mathbf{t}$ is overlap-free.

Proof. We assume that $\mathbf{t}$ has an overlap $axaxa$ beginning at position k, with $|ax| = n$. Thus it looks like

$$\mathbf{t} = t_0 t_1 \cdots t_{k-1} \overbrace{t_k}^{a} \overbrace{t_{k+1} \cdots t_{k+n-1}}^{x} \overbrace{t_{k+n}}^{a} \overbrace{t_{k+n+1} \cdots t_{k+2n-1}}^{x} \overbrace{t_{k+2n}}^{a} \cdots .$$

To get a contradiction we assume that (a) this overlap is smallest among all overlaps in $\mathbf{t}$ and (b) among all overlaps of this size, it appears earliest in $\mathbf{t}$. In other words, we assume that n is as small as possible and k as small as possible for this n.

The Thue-Morse sequence has no overlaps

There are a number of cases to consider:

Case 1: k even, n even.

Since $k + 2n$ is even, we get $t_{k+2n+1} = \overline{t_{k+2n}} = \overline{a}$.

Letting $u = \mathbf{t}[\frac{k}{2} + 1..\frac{k}{2} + \frac{n}{2} - 1]$ and $v = \mathbf{t}[\frac{k}{2} + \frac{n}{2}..\frac{k}{2} + n - 1]$, by the lemma we see $\mu(auava) = axaxa\overline{a} = \mathbf{t}[k..k + 2n + 1]$.

Furthermore, since $\mu(au) = ax = \mu(av)$, we must have $u = v$. So $auaua = \mathbf{t}[\frac{k}{2}..\frac{k}{2} + n]$ is an overlap with $|au| = n/2 < n$, a contradiction.

The Thue-Morse sequence has no overlaps

Case 2: k odd, n even.

Since k is odd, we have $t_{k-1} = \overline{t_k} = \bar{a}$.

Similarly, since $k+n$ and $k+2n$ are both odd, we get $t_{k+n-1} = \overline{t_{k+n}} = \bar{a}$ and $t_{k+2n-1} = \overline{t_{k+2n}} = \bar{a}$.

So the equation above can be rewritten as

$$\mathbf{t} = t_0 t_1 \cdots \overbrace{t_{k-1}}^{\bar{a}} \overbrace{t_k}^{a} \overbrace{t_{k+1} \cdots t_{k+n-2}}^{y} \overbrace{t_{k+n-1}}^{\bar{a}} \overbrace{t_{k+n}}^{a} \overbrace{t_{k+n+1} \cdots t_{k+2n-2}}^{y} \overbrace{t_{k+2n-1}}^{\bar{a}} \overbrace{t_{k+2n}}^{a} \cdots,$$

where $x = y\bar{a}$.

Now $\mathbf{t}[k-1..k+2n-1] = \bar{a}ay\bar{a}ay\bar{a}$ is an overlap of the same length as before, but occurring one place earlier than before, that is, $k-1 < k$. This is a contradiction.

The Thue-Morse sequence has no overlaps

Case 3: k even, n odd.

There are two subcases here: $n = 1$ and $n > 1$. If $n = 1$ then the overlap is just aaa. But from the lemma we know that $\mathbf{t} \in \{01, 10\}^\omega$, so we cannot have three consecutive identical symbols in $\mathbf{t}$.

The second subcase is $n > 1$. The idea here is to "ping-pong" back and forth between the two copies of x, learning more and more symbols of x.

We start with the first copy.

The Thue-Morse sequence has no overlaps

Since $k + n - 1$ is even, the lemma gives us $t_{k+n-1} = \overline{t_{k+n}} = \overline{a}$.

So the last symbol of x must be $\overline{a}$, which tell us that in the second copy of x we have $t_{k+2n-1} = \overline{a}$.

Now $k + 2n - 1$ is odd, so the lemma gives us $t_{k+2n-2} = \overline{t_{k+2n-1}} = a$.

Back in the first copy of x, this gives us $t_{k+n-2} = a$.

Now $k + n - 2$ is odd, so the lemma gives us $t_{k+n-3} = \overline{t_{k+n-2}} = \overline{a}$.

In the second copy of x, we get $t_{k+2n-3} = \overline{a}$, too.

The Thue-Morse sequence has no overlaps

We continue ping-ponging back and forth, learning more and more symbols of x. You can see that the situation above continues, giving

$$x = \cdots a\,\overline{a}\,a\,\overline{a}\,a\,\overline{a}.$$

But since $n = |ax|$ is odd, this means that ax must begin and end with the same symbol. But ax begins with a and ends with $\overline{a}$, a contradiction.

Case 4: k odd, n odd.

This is just like the previous case. In the second subcase, we "ping-pong" back and forth in the same manner, except that we start with the second copy.

Thus, all cases give us a contradiction, so there is no overlap in $\mathbf{t}$.

The logical approach to overlaps

Instead of a tedious case-based argument, we can prove that **t** has no overlaps using a general approach.

Idea: we write a formula (predicate) in a certain logical language expressing the claim that **t** has an overlap, and then use a decision procedure to evaluate the truth of the claim.

$$\exists i\ \exists n \geq 1\ \forall j \leq n\ \mathbf{t}[i+j] = \mathbf{t}[i+j+n].$$

When we run our decision procedure on this statement, we obtain `false` as the output.

Similarly with the formula

$$\exists i\ \exists n \geq 1\ \forall k < 3n\ \mathbf{w}[i+k] = \mathbf{w}[i+k+n],$$

we can prove that the Mephisto Waltz word **w** has no fourth powers in it (no four consecutive identical blocks).

Formal definition of our automaton model

Formally, a deterministic finite automaton with output (DFAO) is a 6-tuple $M = (Q, \Sigma, \delta, q_0, \Delta, \tau)$ as follows:.

- Q is a finite nonempty set of states;.
- Σ is the input alphabet (usually Σ_k);.
- $\delta : Q \times \Sigma \to Q$ is the transition function, which is extended to $Q \times \Sigma^*$;.
- q_0 is the initial (or "start") state;.
- Δ is the output alphabet;.
- $\tau : Q \to \Delta$ is the output map.

Deterministic finite automata (DFA's)

A different, slightly simpler model:.

- No output function, just a distinguished set of states F called the *final* or *accepting* states.
- An automaton *accepts* a word w if processing w takes M from q_0 to a state of F.
- The *language* accepted by automaton M, written $L(M)$, is the set of all words accepted.
- A language is *regular* if it is accepted by some DFA
- There is an efficient algorithm to find the (unique) minimal DFA equivalent to a given DFA (e.g., Valmari).

The pumping lemma: a basic tool for automata

Theorem. If L is a regular language, then there is a constant $n = n(L)$ such that for all $z \in L$ with $|z| \geq n$ there exists a decomposition $z = uvw$ with $|uv| \leq n$ and $|v| \geq 1$ such that $uv^i w \in L$ for all $i \geq 0$.

Proof. A long word accepted $\Longrightarrow$ a long acceptance path $\Longrightarrow$ a loop (repeated state) in the acceptance path $\Longrightarrow$ we can go around this loop as often as we like (or not at all) and still have an acceptance path for the resulting word. ■

Equivalent descriptions of automatic sequences

- Example: the Mephisto Waltz morphism $\omega : 0 \to 001$, $1 \to 110$.
- A morphism is *k-uniform* if every letter's image has the same length k.
  - If $k = 1$ it is called a *coding*.

.

- If $\Delta \subseteq \Sigma$, then we can iterate a morphism.
- $h^2(x) = h(h(x))$, $h^3(x) = h(h(h(x)))$, etc.
- If $h(a) = ax$, then we can iterate h starting on a, producing $h^\omega(a) = axh(x)h^2(x)h^3(x)\cdots$.
- Then $h(h^\omega(a)) = h^\omega(a)$, so $h^\omega(a)$ is sometimes called a *fixed point* of h.

Cobham's little theorem

Theorem. A sequence **s** over the alphabet Δ is k-automatic if and only if there is a k-uniform morphism $h : \Sigma^* \to \Sigma^*$, prolongable on the letter a, and a coding $\tau : \Sigma \to \Delta$, such that $\mathbf{s} = \tau(h^\omega(a))$.

Proof sketch. For one direction, given h and τ, make an automaton where the states q are the letters of Σ, and the initial state is a. The transition from state q on input i goes from q to the i'th letter of $h(q)$. The output associated with q is $\tau(q)$.

For the other direction, from the automaton, let $a = q_0$, the initial state; define $h(q) = \delta(q, 0)\delta(q, 1) \cdots \delta(q, k-1)$, and let $\tau(q)$ be the output associated with state q.

Kernels

The k-kernel of a sequence $\mathbf{a} = a_0 a_1 a_2 \ldots$ is the set $K_k(\mathbf{a})$ of subsequences of the form

$$\{(a_{k^e n+i})_{n\geq 0} \ : \ e \geq 0 \text{ and } 0 \leq i < k^e\}.$$

For example, for $k = 2$ the k-kernel of $\mathbf{a}$ is the set of the following subsequences:

$$(s_n)_{n\geq 0}, (s_{2n})_{n\geq 0}, (s_{2n+1})_{n\geq 0},$$

$$(s_{4n})_{n\geq 0}, (s_{4n+1})_{n\geq 0}, (s_{4n+2})_{n\geq 0}, (s_{4n+3})_{n\geq 0}, \ldots$$

Eilenberg's theorem

Theorem. The sequence $\mathbf{a} = a_0 a_1 a_2 \ldots$ is k-automatic if and only if the k-kernel of $\mathbf{a}$ has finite cardinality.

Proof sketch. Similar to the proof about morphisms, except now we use automata accepting the *reversed* representation of integers, starting with the least significant digit.

Closure properties of automatic sequences

If $\mathbf{a} = a_0 a_1 a_2 \cdots$ and $\mathbf{b} = b_0 b_1 b_2 \cdots$ are k-automatic sequences, so are

- expansion: $h(\mathbf{a})$ for a uniform morphism h;
- shift: $(a_{i+c})_{i \geq 0}$ for any $c \in \mathbb{Z}$;
- linearly-indexed subsequence: $(a_{rn+i})_{n \geq 0}$ for any pair of integers $r \geq 1$, $i \geq 0$.
- cross-product: $\mathbf{a} \times \mathbf{b} = [a_0, b_0][a_1, b_1][a_2, b_2] \cdots$

Cobham's big theorem

We say integers $k, \ell \geq 1$ are *multiplicatively dependent* if there exists an integer $m \geq 1$ and exponents e, f such that $k = m^e$ and $\ell = m^f$. Example: $36 = 6^2$ and $216 = 6^3$ are multiplicatively dependent, with $m = 6$.

Otherwise they are *multiplicatively independent*.

Theorem. Let $\mathbf{a} = a_0 a_1 a_2 \cdots$ be a sequence that is k-automatic and ℓ-automatic, for k and ℓ multiplicatively independent. Then $\mathbf{a}$ is ultimately periodic.

Proposition Let $\mathbf{a}$ be a sequence, and let $k \geq 2$ be an integer. Then $\mathbf{a}$ is k^e-automatic for some $e \geq 1$ iff it is k^f-automatic for all $f \geq 1$.

Generalizations: morphic sequences

- More generally, for any morphism prolongable on a we can consider the infinite sequence $h^\omega(a)$
- Example: $h(0) = 01$, $h(1) = 0$ gives the infinite Fibonacci word $0100101001001\cdots$
- Such a sequence is called *pure morphic*; the image under a coding of such a sequence is called *morphic*.
- Morphic sequences have been widely studied, but are harder to understand, in general, than automatic sequences

Non-automatic sequences

Not all morphic sequences are automatic.

Here is an example that we will prove not k-automatic for any k later:

Consider the morphism defined by $h(a) = aab$ and $h(b) = b$.

The infinite fixed point is

$$h^\omega(a) = aabaabbaabaabbb\cdots .$$

This sequence is not k-automatic for any k.

Another generalization: k-regular sequences

Automatic sequences are always defined over a finite alphabet.

Can we generalize to an infinite alphabet, such as $\mathbb{N}$?

Example: The sequence $s_2(n) =$ the sum of the digits in the base-2 representation of n.

The first few terms of $(s_2(n))_{n \geq 0}$ are as follows:

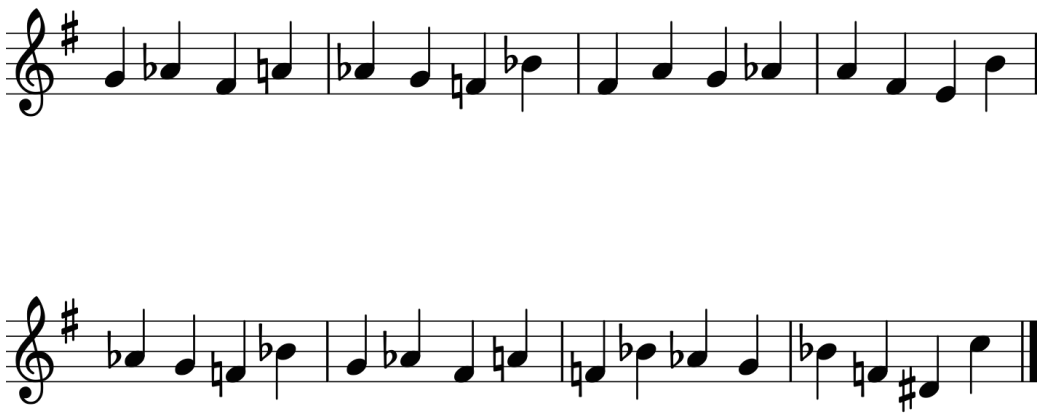
n	0	1	2	3	4	5	6	7	8	9	10
$s_2(n)$	0	1	1	2	1	2	2	3	1	2	2

Note that, for $n \geq 0$,

$$s_2(2n) = s_2(n)$$
$$s_2(2n+1) = s_2(n) + 1$$

Another example: Per Nørgård's "infinity sequence"

This is a sequence $(s(n))_{n\geq 0}$ of integers representing the number of half-steps above or below a given base note. Here are the first 32 notes:



and the first 16 terms:

n	0	1	2	3	4	5	6	7	8	9	10
$s(n)$	0	1	-1	2	1	0	-2	3	-1	2	0

The recurrence is $s(0) = 0$, $s(2n) = -s(n)$ for $n \geq 1$, and $s(2n+1) = s(n) + 1$ for $n \geq 0$.

More examples of k-regular sequences

- the Mallows sequence: the unique monotone sequence $(a(n))_{n\geq 0}$ of non-negative integers such that $a(a(n)) = 2n$ for $n \neq 1$

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$a(n)$	0	1	3	4	6	7	8	10	12	13	14	15	16	18	20	22	24

It satisfies the recurrence

$$\begin{aligned} a(4n) &= 2a(2n) \\ a(4n+1) &= a(2n) + a(2n+1) \\ a(4n+3) &= -2a(n) + a(2n+1) + a(4n+2) \\ a(8n+2) &= 2a(2n) + a(4n+2) \\ a(8n+6) &= -4a(n) + 2a(2n+1) + 2a(4n+2) \end{aligned}$$

- the number of overlap-free binary words of length n (Carpi; Cassaigne)

More examples of k-regular sequences

Divide-and-conquer linear recurrence for mergesort

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + n - 1$$

for $n \geq 2$.

It satisfies the system

$$\begin{aligned}
T(4n+1) &= 4T(n) - 5T(2n) + T(2n+1) + 2T(4n) \\
T(4n+3) &= -12T(n) + 15T(2n) - 3T(2n+1) - 5T(4n) + 3T(4n+2) \\
T(8n) &= 4T(n) - 8T(2n) + 5T(4n) \\
T(8n+2) &= 12T(n) - 16T(2n) + 6T(4n) + T(4n+2) \\
T(8n+4) &= -4T(n) + 6T(2n) - 6T(2n+1) - 2T(4n) + 5T(4n+2) \\
T(8n+6) &= -36T(n) + 48T(2n) - 16T(2n+1) - 16T(4n) + 11T(4n+2)
\end{aligned}$$

k-regular sequences

- An integer sequence $(a_n)_{n\geq 0}$ is said to be *k-regular* if the $\mathbb{Z}$-module generated by the sequences in the k-kernel is *finitely generated*.
- Example: for $s_2(n)$ the relations $s_2(2n) = s_2(n)$ and $s_2(2n+1) = s_2(n) + 1$ imply that $\langle K_2(s_2)\rangle$ is generated by $(s_2(n))_{n\geq 0}$ and the constant sequence 1.
- k-regular sequences appear in many different fields of mathematics: numerical analysis, topology, number theory, combinatorics, analysis of algorithms, and fractal geometry.

Properties of k-regular sequences

- Every k-automatic sequence is also k-regular.
- If a k-regular sequence is bounded, then it is k-automatic.
- The k-regular sequences are closed under shift, and periodic deletion.
- A sequence is k-regular iff it is k^r-regular for any $r \geq 2$.
- The k-regular sequences are closed under (termwise) sum and product.
- If $f(X) = \sum_{n \geq 0} f_n X^n$ and $g(X) = \sum_{n \geq 0} g_n X^n$ are formal power series with k-regular coefficients, then so is $f(X)g(X)$.

Linear representations

A sequence $(a_n)_{n\geq 0}$ has a *k-linear representation* if there is a $d \times d$ matrix-valued morphism h with domain Σ_k^* and two d-element vectors u, v such that for all $n \geq 0$

$$a_n = uh(x)v^T$$

when $x = (n)_k$.

For example, here is a linear representation of $s_2(n)$:

$$u = [0\ 1], \quad \mu(0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mu(1) = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, v = [1\ 0].$$

Linear representations

Theorem. A sequence has a k-linear representation if and only if it is k-regular.

Closure properties of k-regular sequences

Theorem. Let $k \geq 2$ be an integer. The class of k-regular sequences taking values in $\mathbb{Q}$ is closed under

(a) scalar multiplication by elements of $\mathbb{Q}$: $d(n) = ca(n)$

(b) shift: $d(n) = a(n+1)$

(c) running sum: $d(n) = \sum_{0 \leq i < n} a(n)$

(d) linearly-indexed subsequence: $d(n) = a(mn+i)$ for $m \geq 1$, $i \geq 0$

(e) $\mathbb{Q}$-linear combination: $b(n) = \sum_{1 \leq i \leq t} c_i a_i(n)$

(f) product: $d(n) = \prod_{1 \leq i \leq t} a_i(n)$

(g) convolution: $d(n) = \sum_{i+j=n} a(i)b(n-i)$

(h) perfect shuffle: $d(mn+i) = a_i(n)$ for $0 \leq i < m$

Tomorrow

Tomorrow: logic and automatic sequences.