Complexity of linear subsequences of *k*-automatic sequences

Delaram Moradi[1], Narad Rampersad[2]
Jeffrey Shallit[1]

[1]David R. Cheriton School of Computer Science
University of Waterloo

[2]Department of Mathematics and Statistics
University of Winnipeg

State complexity of arithmetic operations

- The arithmetic operations of addition and multiplication by a constant can be performed by a finite automaton reading its input in base-k.
- How many states does such an automaton need to perform these operations?

Automatic sequences

- Finite automata can also be used to compute infinite sequences.
- A k-automatic sequence $(a(i))_{i \geq 0}$ is computed by a deterministic finite automaton with output (DFAO) that reads i written in base-k and outputs $a(i)$.
- We consider both most significant digit (msd) first and least significant digit (lsd) first representations.

Linear subsequences

- Let n and c be fixed constants.
- Given an m-state DFAO for a k-automatic sequence $(a(i))_{i \geq 0}$, how many states are required for a DFAO computing a shift $(a(i + c))_{i \geq 0}$?
- What about an arbitrary linear subsequence $(a(ni + c))_{i \geq 0}$?

A problem of Zantema and Bosma

- Suppose $(a(i))_{i \geq 0}$ is computed by an m-state lsd-first DFAO.
- Zantema and Bosma (2022) showed that for $0 \leq c < n$, the linear subsequence $(a(ni+c))_{i \geq 0}$ can be computed by a DFAO with at most mn states.
- They showed that this bound does not hold for msd-first representation and left it as an open problem to obtain a good bound in this case.
- We address this problem in this paper.

Presburger and Büchi arithmetic

- The first order theory of the natural numbers with addition (but not multiplication) is called Presburger arithmetic.
- Extending Presburger arithmetic with the predicate

$$V_k(x) = k^n,$$

where k^n is the largest power of k that divides x, gives Büchi arithmetic.

Decidability of Büchi arithmetic

- Both theories are decidable using automata.
- There is an algorithm that, given a formula in the logic, produces a DFA accepting the base-k representations of the numbers satisfying the formula.
- How big is this DFA?

Size of automata for a logical formula

- Boudet and Comon (1996) (lsd-first) and Wolper and Boigelot (2000) (msd-first) studied this problem for Presburger arithmetic.
- Büchi arithmetic (and hence its Presburger fragment) has been implemented in Walnut.
- Amat, Ganty, and Mansutti (2025) gave lower bounds for the size of Presburger DFAs and compared these with the automata computed by Walnut and Amaya.

Size of Walnut automata

- In this paper we give bounds on the sizes of automata as constructed by Walnut for some basic operations.

Representing pairs

- To represent pairs of numbers, we use the alphabet $\{0,1\} \times \{0,1\}$ and pad the shorter representation with leading/trailing zeros, if necessary.
- For example, in msd-first base 2,

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

  represents the pair $(5,20)$.
- If w and x have the same length then we write $w \times x$ to mean the word over $\{0,1\} \times \{0,1\}$ whose first component is w and whose second component is x.

Addition in base-k

Theorem

For all integers $c \geq 1$ and both lsd- and msd-first input, there exists an automaton with at most $O(\log_k c)$ states accepting $x \times y$ where $[x]_k + c = [y]_k$.

Construction of M_c (msd version)

- M_c reads input $x \times y$ in parallel and accepts if and only if $[y]_k = [x]_k + c$.
- We first define $D(x, y) = [y]_k - [x]_k$.
- States of M_c record values of $D(x, y)$.
- We have $[yb]_k - [xa]_k = k([y]_k - [x]_k) + b - a$.
- So if $D(x, y) \notin [0, c]$, it can never return to this interval.

How many possible values of $D(x, y)$?

- The transition function on input $[a, b]$ maps $d \to kd + b - a$.
- The final state is c.
- The only states that can reach c are $c' = \lfloor c/k \rfloor$ and $c'' = \lceil c/k \rceil$.
- Now find the states that can reach c' and c'' and repeat.

How many possible values of $D(x, y)$?

- Note that $\{\lfloor c'/k \rfloor, \lfloor c''/k \rfloor, \lceil c'/k \rceil, \lceil c''/k \rceil\}$ has size at most 2.
- Continue finding predecessor states in this way.
- We reach state 0 after $O(\log_k c)$ steps and two new states are found at each step.
- There are thus $O(\log_k c)$ states total.

Exact bound for base-k msd

Theorem

For all integers $c \geq 1$ and msd-first input, the minimal automaton accepting $x \times y$ where $[x]_k + c = [y]_k$ has

$$2|(c)_k| + 1 - \nu_k(c) - [(c)_k \text{ starts with } 1]$$
$$- [k = 2, (c)_2 \text{ starts with } 10 \text{ and } c \text{ is not a power of } 2]$$

states.

($\nu_k(c) = e$ where k^e is the largest power of k dividing c.)

Multiplication in base-k

Theorem

Let $n \geq 1$, $0 \leq c < n$ be fixed constants. For both lsd- and msd-first input there is an n-state automaton accepting $x \times y$ where $n * [x]_k + c = [y]_k$.

k-automatic sequences

- A k-DFAO $M = (Q, \{0, \ldots, k-1\}, \Delta, \delta, q_0, \tau)$ is a DFA with an output function $\tau : Q \longrightarrow \Delta$ instead of accepting states.
- It generates a k-automatic sequence $\mathbf{h} = (h(i))_{i \geq 0}$ in the following way:
- The input is a representation x of i in base-k, and the output $h(i)$ is $\tau(\delta(q_0, x))$.

The interior sequence

- There is a unique associated interior sequence $\mathbf{h}' = (h'(i))_{i \geq 0}$ taking its values in Q, whose DFAO is obtained by replacing $\tau : Q \rightarrow \Delta$ by the identity map on Q.
- Thus $\mathbf{h}$ is the image of $\mathbf{h}'$ under the coding $q \rightarrow \tau(q)$.

Subword complexity

- The subword complexity of **h** is the function $\rho_{\mathbf{h}}(n)$ that counts the number of subwords (factors) of **h** of length n.
- If **h** is generated by a msd-first k-DFAO with m states then

$$\rho_{\mathbf{h}}(n) \leq kn\rho_{\mathbf{h}'}(2) \leq knm^2.$$

Linear subsequences

Theorem

Let $(h(i))_{i\geq 0}$ be a k-automatic sequence generated by an m-state msd-first DFAO M and let $n \geq 1$ and $c \geq 0$ be fixed integer constants.

(a) *If $c < n$, then there is a msd-first DFAO of at most $\rho_{\mathbf{h}'}(n)$ states generating $(h(ni+c))_{i\geq 0}$.*

(b) *If $c \geq n$, then there is a msd-first DFAO of at most $\rho_{\mathbf{h}'}(c+1)$ states generating $(h(ni+c))_{i\geq 0}$.*

The automaton for $(h(ni + c))_{i \geq 0}$

- We define a DFAO M' that on input x reaches a state of the form

$$[h'(n[x]_k), \ldots, h'(n[x]_k + n - 1)];$$

  i.e., the states of M' are length-n factors of the interior sequence $\mathbf{h}'$.

- The output function of M' maps the factor $[r_0, \ldots, r_{n-1}]$ to r_c.
- The number of states of M' is $\rho_{\mathbf{h}'}(n)$.

Subword complexity of $(h(ni))_{i\geq 0}$

Theorem

Let $\mathbf{h} = (h(i))_{i\geq 0}$ *be a* k*-automatic sequence and* $\mathbf{h}' = (h'(i))_{i\geq 0}$ *its interior sequence. Let* $\mathbf{h}_n$ *denote the linear subsequence* $(h(ni))_{i\geq 0}$ *and similarly for* $\mathbf{h}'_n$*. Then* $\rho_{\mathbf{h}_n}(\ell) \leq k\ell\rho_{\mathbf{h}'}(2n)$ *for all* $\ell, n \geq 1$*.*

The Thue–Morse word

- The Thue-Morse sequence $\mathbf{t} = (t(i))_{i \geq 0}$ is the fixed point of the morphism $0 \to 01$, $1 \to 10$; or,
- $t(i)$ is equal to the number of 1's modulo 2 in the binary representation of i.

Subword complexity of $(t(ni))_{i\geq 0}$

Corollary

Let $\mathbf{t}_n$ *denote the linear subsequence* $(t(ni))_{i\geq 0}$*. We have* $\rho_{\mathbf{t}_n}(\ell) \leq \frac{40}{3}\ell n$ *for* $\ell, n \geq 1$*.*

State complexity of $(t(ni))_{i\geq 0}$

Theorem

For all integers $n \geq 1$, the number of states in the minimal DFAO generating $(t(ni))_{i\geq 0}$ with msd-first input is $\rho_{\mathbf{t}}(n/2^{\nu_2(n)})$.

State complexity of $(t(i+c))_{i\geq 0}$

Theorem

For all integers $c \geq 1$, the number of states in the minimal automaton generating $(t(i+c))_{i\geq 0}$ with msd-first input is at most $\frac{10}{3}c$.

State complexity of $(t(i+c))_{i\geq 0}$

Theorem

The minimal DFAO computing $(t(i+c))_{i\geq 0}$ with msd-first input has more than $c^{0.694}$ states for infinitely many c.

Jeff Shallit has subsequently been able to improve this to $c^{0.9039}$.

Büchi arithmetic

- We can also study these state complexity results from another point of view.
- Instead of trying to directly construct the automaton for a given arithmetic operation, we analyze the size obtained when constructing it using the primitives available in an implementation of Büchi arithmetic, such as Walnut.
- We also want to analyze the time complexity of performing such a construction.

Multiplication in Büchi arithmetic

- Let $M_{\mathrm{add}}(x, y, z)$ be an automaton for addition $[x]_k + [y]_k = [z]_k$.
- We can compute an automaton $M_n(x, z)$ that recognizes $n[x]_k = [z]_k$ recursively.
- $M_n(x, z)$ accepts if the following hold:
- n even: $\exists y\ M_{n/2}(x, y) \wedge M_{\mathrm{add}}(y, y, z)$,
- n odd: $\exists y\ M_{n-1}(x, y) \wedge M_{\mathrm{add}}(x, y, z)$.
- This recursion results in an $O(n \log^2 n)$-time algorithm to compute M_n.

Comparing factors in Büchi arithmetic

- Let $\mathbf{h} = (h(i))_{i \geq 0}$ be a k-automatic sequence.
- We very often want to be able to define equality of factors of $\mathbf{h}$.
- i.e., compute an automaton M' that on input $x \times y \times z$, accepts exactly when the length-$[z]$ factors of $\mathbf{h}$ occurring at positions $[x]$ and $[y]$ are equal.
- This is definable in Büchi arithmetic.
- Open Problem: How big is the automaton M'?

Other numeration systems

- In a companion paper (DLT 2026), we have examined the same questions for the Fibonacci-base numeration system.
- Let $(f(i))_{i \geq i}$ denote the Fibonacci word (fixed point of $0 \to 01$, $1 \to 0$).
- It is Fibonacci-automatic.
- Moradi, Popoli, Shallit, and Vukusic (DCFS 2026) have shown a $O(\log c)$ state complexity bound for $(f(i+c))_{i \geq 0}$ for both msd- and lsd-first Fibonacci representation.

Other open questions

- What is the tight bound for the state complexity of $(t(i+c))_{i\geq 0}$ (msd- and lsd-first representations)?

The End