The repetition threshold for binary Pell-automatic sequences is 7/3

Qianghao Wu

27 November 2026

Abstract

The critical exponent of an infinite word is the supremum of the exponents of its finite factors. The repetition threshold of a set of infinite words is the infimum of their critical exponents. Allouche, Rampersad and Shallit proved that the repetition threshold of the binary Fibonacci-automatic sequences and of the binary Tribonacci-automatic sequences is 7/3. They wrote that 7/3 is likely the critical exponent for many other binary generalized automatic sequences, such as the Pell-automatic sequences. We prove that the repetition threshold of the binary Pell-automatic sequences is 7/3, and that one Pell-automatic sequence attains it. The upper bound uses a 4-state Pell automaton whose output word $\mathbf{s}$ over 4 letters is squarefree. The theorem prover Walnut proves that $\mathbf{s}$ is squarefree, and a separate decision procedure that does not use Walnut confirms it. A 21-uniform morphism of Karhumäki and Shallit then gives a binary Pell-automatic sequence with critical exponent exactly 7/3. The lower bound follows the argument of Allouche, Rampersad and Shallit, with Durand's theorem for substitutions. We give the non-erasing substitution that this argument needs for the Pell numeration system. The proof is computer-assisted and uses published lemmas that we do not prove again. The method is the method of Allouche, Rampersad and Shallit, applied to one more numeration system; the general question for other numeration systems remains open. This work has not yet been reviewed by experts. The computations, the proofs and the text were produced by AI models (from Anthropic and OpenAI), working under the author's direction.

1 Introduction

Let w be a finite nonempty word. An integer p with $1 \le p \le |w|$ is a period of w if $w[i] = w[i+p]$ for all i with $1 \le i \le |w| - p$. The exponent of w is $\exp(w) = |w|/p$, where p is the least period of w. For example, 1001001 has least period 3 and exponent 7/3. A word is *squarefree* if it has no factor of exponent 2 or more. A word is *α-free* if it has no factor of exponent α or more, and *α^+-free* if it has no factor of exponent more than α. The *critical exponent* $\mathrm{ce}(\mathbf{x})$ of an infinite word $\mathbf{x}$ is the supremum of $\exp(w)$ over all finite nonempty factors w of $\mathbf{x}$. The *repetition threshold* $\mathrm{rt}(S)$ of a nonempty set S of infinite words is the infimum of $\mathrm{ce}(\mathbf{x})$ over $\mathbf{x} \in S$. These are the definitions of Allouche, Rampersad and Shallit [1].

The introduction of [1] gives the history of the repetition threshold. For all infinite binary words, the value is 2, and the Thue–Morse word attains it. Repetition thresholds are also known for subsets such as the Sturmian, balanced, rich and episturmian words; [1] lists the references.

Allouche, Rampersad and Shallit [1] study the binary automatic sequences. Let $\mathcal{A}_k$ be the set of binary k-automatic sequences. They prove:

- $\mathrm{rt}(\mathcal{A}_k) = 2$ if k is a power of 2, and $\mathrm{rt}(\mathcal{A}_k) = 7/3$ otherwise [1, Theorem 1];
- $\mathrm{rt}(\mathcal{A}_F) = 7/3$ for the binary Fibonacci-automatic sequences [1, Theorem 11];

- the repetition threshold of the binary Tribonacci-automatic sequences is 7/3 [1, Theorem 13].

In each case the value 7/3 comes from two parts. A squarefree automatic word over 4 letters and a 21-uniform morphism h of Karhumäki and Shallit [4] give the upper bound. A structure theorem for 7/3-free binary words [4, 6] and a Cobham-type theorem (Cobham's theorem for base k, Durand's theorem [3] for Fibonacci and Tribonacci) give the lower bound. Section 4 of [1] (v3, February 2026), which has the title "Going further", is the following sentence:

> "It seems likely that 7/3 is the critical exponent for many other binary generalized automatic sequences, such as the Pell-automatic sequences [3], but a general theorem seems hard to prove."

Here [3] is Baranwal and Shallit [2], who use the Pell numeration system in Walnut to find the least critical exponent of balanced words over 5 letters.

Result. Let $\mathcal{A}_P$ be the set of binary Pell-automatic sequences (Definition 2).

Theorem 1. $\mathrm{rt}(\mathcal{A}_P) = 7/3$. *Moreover, there is a sequence in $\mathcal{A}_P$ whose critical exponent is exactly 7/3.*

This settles the Pell case that [1] names. The proof is the proof of [1, Theorem 11] with two new parts: a Pell automaton for a squarefree word (Section 3), and the non-erasing substitution that Durand's theorem needs for the Pell system (Section 4). The general question of [1] for "many other binary generalized automatic sequences" remains open.

Prior work on this question. We searched the two public collections of AI-produced results that we track, the issue trackers and archives where such results are posted, Zenodo, GitHub and the web for the Pell case (27 November 2026). We found no proof. One public list of open targets names this case as open. Version 3 of [1] (February 2026) still calls the question open. Semantic Scholar and OpenAlex list no paper that cites [1] (checked 27 November 2026). These indexes can be incomplete, so we cannot exclude that a paper we did not find settles the case.

Kind of result. The result is computer-assisted. The squarefree property of the 4-letter word rests on Walnut and on a second decision procedure. Four published results are used and not proved again here: [4, Lemma 8], [6, Theorem 1], [1, Lemma 10] and [3, Theorem 1]. There is no Lean proof.

2 The Pell numeration system

The Pell numbers that we use as weights are $P_0 = 1$, $P_1 = 2$ and $P_{i+2} = 2P_{i+1} + P_i$, so $(P_i) = 1, 2, 5, 12, 29, 70, \ldots$. Each $n \geq 0$ has a unique representation $n = \sum_{0 \leq i < m} d_i P_i$ with digits $d_i \in \{0, 1, 2\}$ that satisfy the conditions of Baranwal and Shallit [2, Section 2.1]:

1. $d_0 \in \{0, 1\}$;
2. if $d_i = 2$ with $i > 0$, then $d_{i-1} = 0$.

The greedy algorithm gives this representation. We write it most significant digit first, $d_{m-1} \cdots d_0$, and call a word over $\{0, 1, 2\}$ *valid* if it satisfies conditions 1 and 2. A valid word can have leading zeros. Let W_m be the set of valid words of length m. Then $W_0 = \{\varepsilon\}$, $W_1 = \{0, 1\}$, and for $m \geq 1$,

$$W_{m+1} = 0W_m \cup 1W_m \cup 20W_{m-1}. \tag{1}$$

The words of W_m in lexicographic order represent $0, 1, \ldots, P_m - 1$ in increasing order. This is Walnut's numeration system `msd_pell`.

Definition 2. A *Pell DFAO* is a tuple (Q, δ, q_0, τ) with a finite set Q, a map $\delta : Q \times \{0, 1, 2\} \to Q$, a state $q_0 \in Q$ with $\delta(q_0, 0) = q_0$, and a map $\tau : Q \to \Delta$ to a finite alphabet Δ. It computes the sequence $x(n) = \tau(\delta(q_0, \mathrm{rep}(n)))$, $n \geq 0$, where $\mathrm{rep}(n)$ is the greedy representation of n. A sequence is *Pell-automatic* if a Pell DFAO computes it.

A DFAO that reads the digits least significant digit first gives the same class of sequences, because the reversal of a regular language is regular.

The condition $\delta(q_0, 0) = q_0$ costs no generality. If a DFAO reads the representations without leading zeros and has $\delta(q_0, 0) \neq q_0$, add a new initial state q_0' with $\delta(q_0', 0) = q_0'$, $\delta(q_0', d) = \delta(q_0, d)$ for $d \in \{1, 2\}$ and $\tau(q_0') = \tau(q_0)$. The new DFAO computes the same sequence and ignores leading zeros. The Pell system is *addable*: a finite automaton recognizes the relation $x + y = z$ on valid representations [2, Section 2.2]. Walnut ships this automaton, and our second decision procedure builds its own (Section 5).

3 The upper bound

Let $\mathbf{s}$ be the sequence over $\{0, 1, 2, 3\}$ that the Pell DFAO of Table 1 computes. Each state outputs its own number.

state q	$\delta(q, 0)$	$\delta(q, 1)$	$\delta(q, 2)$
0 (initial)	0	1	2
1	0	2	0
2	3	1	2
3	1	2	2

Table 1: The Pell DFAO for $\mathbf{s}$; $\tau(q) = q$. File `data/PSQ.txt`, in Walnut's format.

The first 60 letters of $\mathbf{s}$ are

`010230131012010231202301023130102301310120231301310120102301`.

For example, $\mathrm{rep}(3) = 11$, and $\delta(0, 1) = 1$, $\delta(1, 1) = 2$, so $\mathbf{s}(3) = 2$.

We found this DFAO with a depth-first search over Pell DFAOs with at most 4 states and 4 output letters, for a squarefree prefix of length 3000 (`checks/search/Sqf.java`). It is the first DFAO that the search finds. The same search with at most 3 states finds none; its longest squarefree prefix has length 11.

Proposition 3. *The infinite word* $\mathbf{s}$ *is squarefree.*

Proof (computer-assisted). Walnut [5, 7] (commit `c61231a`) evaluates the first-order sentence

$$\neg\exists i, n \ \big(n \geq 1 \wedge \forall t \ (t < n \Rightarrow \mathbf{s}[i + t] = \mathbf{s}[i + t + n])\big)$$

in `msd_pell` and returns `TRUE`. This is the form that [1, proof of Theorem 11] uses for the Fibonacci case. Section 5 gives a second decision procedure that does not use Walnut and gives the same answer. □

Let $h : \{0,1,2,3\}^* \to \{0,1\}^*$ be the 21-uniform morphism of Karhumäki and Shallit [4, Section 5], as quoted in [1, Lemma 5]:

$$h(0) = 011010011001001101001, \qquad h(1) = 100101100100110010110,$$
$$h(2) = 100101100110110010110, \qquad h(3) = 011010011011001101001.$$

Lemma 4 ([4, Lemma 8], [1, Lemma 5])**.** *If* $\mathbf{a}$ *is a squarefree word over* $\{0,1,2,3\}$*, then* $h(\mathbf{a})$ *is* $(7/3)^+$*-free.*

Lemma 5. *The binary sequence* $h(\mathbf{s})$ *is Pell-automatic.*

Proof. Bit n of $h(\mathbf{s})$ is 1 if and only if there are q and r with $n = 21q+r$, $r < 21$, and $h(\mathbf{s}(q))[r] = 1$. For each letter a the set of such r is finite, so this is a first-order formula over $\langle \mathbb{N}, +, \mathbf{s}\rangle$. Since the Pell system is addable, Walnut's decision procedure turns such a formula into a Pell automaton. Walnut builds this automaton (`def hb` in `checks/walnut/hb.txt`; it has 1518 states) and confirms that its first four values are $0,1,1,0$, the first four letters of $h(0)$. A script checks that the positions r in this formula are exactly the positions of the letters 1 in $h(0), \ldots, h(3)$ (`checks/ci.sh`). □

Corollary 6. $\mathrm{ce}(h(\mathbf{s})) = 7/3$.

Proof. By Proposition 3 and Lemma 4, $h(\mathbf{s})$ has no factor of exponent more than 7/3. Its factor at positions 8 to 14 (counted from 0) is 1001001, inside $h(0)$. This factor has length 7, least period 3 and exponent 7/3. □

4 The lower bound

Proposition 7. *Every* $\mathbf{x} \in \mathcal{A}_P$ *has* $\mathrm{ce}(\mathbf{x}) \geq 7/3$.

The proof is the argument of [1, proofs of Theorems 1(b) and 11]. We state the steps and the facts that each step uses. Let μ be the Thue–Morse morphism $0 \mapsto 01$, $1 \mapsto 10$, let $\mathbf{t} = \mu^\omega(0)$, and let $\overline{\mathbf{t}}$ be its complement.

Step 1. Suppose $\mathrm{ce}(\mathbf{x}) < 7/3$. Then each factor of $\mathbf{x}$ has exponent less than 7/3, so $\mathbf{x}$ is 7/3-free. By [6, Theorem 1] (Lemma 7 and Corollary 8 of [1]), $\mathbf{x}$ contains $\mu^i(0)$ and $\mu^i(1)$ for all i. So every prefix of $\overline{\mathbf{t}}$ is a factor of $\mathbf{x}$, and $\overline{\mathbf{t}}$ is in the orbit closure of $\mathbf{x}$.

Step 2. Each sequence in the orbit closure of $\mathbf{x}$ is 7/3-free. By [1, Lemma 9], the lexicographically least 7/3-free binary sequence that starts with 1 is $\overline{\mathbf{t}}$. So the lexicographically least sequence in the orbit closure of $\mathbf{x}$ that starts with 1 is $\overline{\mathbf{t}}$.

Step 3. By [1, Lemma 10], which holds for every addable numeration system, this least sequence is generalized automatic in the same system. So $\overline{\mathbf{t}}$ is Pell-automatic. (The proof of [1, Lemma 10] is one sentence that refers to Walnut methods; we use the lemma as published.)

Step 4. Lemma 8 below shows that every Pell-automatic sequence is $(1+\sqrt{2})$-substitutive in the sense of Durand [3, Definition 2]. The sequence $\overline{\mathbf{t}}$ is also 2-substitutive: it is the fixed point of μ that starts with 1, with the identity coding, and the incidence matrix of μ has dominating eigenvalue 2.

Step 5. The numbers $1+\sqrt{2}$ and 2 are Perron numbers. They are multiplicatively independent: for $k \geq 1$, $(1+\sqrt{2})^k = a_k + b_k\sqrt{2}$ with integers $a_k, b_k \geq 1$, so $(1+\sqrt{2})^k$ is irrational and is not equal to 2^l. The cases $k = 0$ and $k < 0$ reduce to this case. By Durand's theorem [3, Theorem 1], a sequence that is both $(1+\sqrt{2})$-substitutive and 2-substitutive is ultimately periodic. But $\bar{\mathbf{t}}$ is not ultimately periodic. An ultimately periodic word contains a cube uuu with u nonempty, and $\bar{\mathbf{t}}$, like $\mathbf{t}$, has no factor of exponent more than 2 [1, Section 1]. This contradiction proves Proposition 7.

Durand's definition [3, Definition 2] needs a substitution σ that is prolongable on a letter a, such that every letter occurs in $\sigma^\omega(a)$, the incidence matrix of σ has dominating eigenvalue α, and the sequence is the image of $\sigma^\omega(a)$ under a *coding* (a letter-to-letter map). Durand shows by an example that the theorem fails if the coding is replaced by an erasing morphism [3, after Corollary 6]. The obvious automaton for valid Pell words has the substitution $A \mapsto AAB$, $B \mapsto A$, but the letter B stands for a digit 2 in the last position, which is not valid. That route therefore gives the sequence only after the letters of type B are erased. The next lemma avoids the erasure.

Lemma 8. *Let (Q, δ, q_0, τ) be a Pell DFAO with $\delta(q_0, 0) = q_0$, and let $\mathbf{x}$ be the sequence that it computes. On the alphabet $\{A_q : q \in Q\} \cup \{C_{q,r} : q, r \in Q\}$ define the substitution*

$$\sigma(A_q) = A_{\delta(q,0)}\, C_{\delta(q,1),\delta(q,2)},$$
$$\sigma(C_{q,r}) = A_{\delta(q,0)}\, C_{\delta(q,1),\delta(q,2)}\, A_{\delta(r,0)},$$

and the coding $\varphi(A_q) = \varphi(C_{q,r}) = \tau(q)$. Then σ is prolongable on A_{q_0}, and $\mathbf{x} = \varphi(\sigma^\omega(A_{q_0}))$. If σ is restricted to the letters that occur in $\sigma^\omega(A_{q_0})$, its incidence matrix has dominating eigenvalue $1+\sqrt{2}$. So $\mathbf{x}$ is $(1+\sqrt{2})$-substitutive.

Proof. For $q \in Q$ and $m \geq 0$, let $X_m(q)$ be the word $\tau(\delta(q, w))$, $w \in W_m$, in lexicographic order of w. Put $X_{-1}(q) = \varepsilon$. For $m \geq 0$ let $Z_m(q, r) = X_m(q)\, X_{m-1}(\delta(r, 0))$. By (1), and since $W_1 = 0W_0 \cup 1W_0$, for all $m \geq 0$

$$X_{m+1}(q) = X_m(\delta(q,0))\, X_m(\delta(q,1))\, X_{m-1}(\delta(\delta(q,2),0)) = X_m(\delta(q,0))\, Z_m(\delta(q,1),\delta(q,2)).$$

Then

$$Z_{m+1}(q,r) = X_{m+1}(q)\, X_m(\delta(r,0)) = X_m(\delta(q,0))\, Z_m(\delta(q,1),\delta(q,2))\, X_m(\delta(r,0)).$$

These two identities have the same form as the two rules of σ. Since $X_0(q) = Z_0(q, r) = \tau(q)$, induction on m gives $\varphi(\sigma^m(A_q)) = X_m(q)$ and $\varphi(\sigma^m(C_{q,r})) = Z_m(q, r)$ for all $m \geq 0$.

Since $\delta(q_0, 0) = q_0$, $\sigma(A_{q_0})$ starts with A_{q_0} and has length 2, so σ is prolongable on A_{q_0}. The word $X_m(q_0)$ lists $\mathbf{x}(0), \ldots, \mathbf{x}(P_m - 1)$, because leading zeros do not change the state. So $\varphi(\sigma^\omega(A_{q_0})) = \mathbf{x}$.

Each image under σ of a letter A contains one letter A and one letter C. Each image of a letter C contains two letters A and one letter C. So $a_m = |\sigma^m(A_q)|$ and $c_m = |\sigma^m(C_{q,r})|$ do not depend on q and r, and $a_{m+1} = a_m + c_m$, $c_{m+1} = 2a_m + c_m$, $a_0 = c_0 = 1$. This gives $a_m = P_m$ and $c_m = P_m + P_{m-1}$ ($P_{-1} = 0$). Both grow as a constant times $(1+\sqrt{2})^m$. By Gelfand's formula, the spectral radius of a square matrix N is $\lim_m \|N^m\|^{1/m}$ for every matrix norm; we use the largest column sum. Let M be the incidence matrix of σ restricted to the letters that occur (this set of letters is closed under σ), with column b counting the letters of $\sigma(b)$. Then the largest column sum of M^m is the largest length $|\sigma^m(b)|$. So the spectral radius is $1+\sqrt{2}$, and by the Perron–Frobenius theorem it is the dominating eigenvalue. All conditions of [3, Definition 2] hold. □

Remark 9. A computer check confirms the identity $\mathbf{x} = \varphi(\sigma^\omega(A_{q_0}))$ for $\mathbf{s}$ on its first 100000 letters, and for 200 random 4-state Pell DFAOs on their first 3000 letters (`checks/prefix.py`). The verifier also checked it for $\mathbf{s}$ on 3000000 letters.

Proof of Theorem 1. Proposition 7 gives $\mathrm{rt}(\mathcal{A}_P) \geq 7/3$. Lemma 5 and Corollary 6 give a sequence in $\mathcal{A}_P$ with critical exponent $7/3$. □

5 How the result was checked

Table 2 lists the checks. "Infinite" means that the check is a decision procedure for the whole infinite word. "Prefix" means a finite check that only supports the result.

Check	Kind	Result
Walnut, Proposition 3 (two forms of the sentence)	infinite	`TRUE`, `TRUE`; solver run, verifier run on a fresh clone, write-stage run with a Walnut jar built from source
Second decision procedure `checks/decide.py`	infinite	no square in $\mathbf{s}$; 442377 reachable states in the final search
Controls for `decide.py`	infinite	32 of 32 changed DFAOs: a square found and confirmed
Walnut, Lemma 5	infinite	automaton for $h(\mathbf{s})$ built; first values $0, 1, 1, 0$
`checks/prefix.py`	prefix	no square in the first 50000 letters of $\mathbf{s}$; largest exponent of a factor of the first 200000 letters of $h(\mathbf{s})$ is exactly $7/3$
Verifier's Java checker	prefix	no square in 3000000 letters of $\mathbf{s}$; no factor of exponent above $7/3$ in 63000000 letters of $h(\mathbf{s})$

Table 2: Checks of the computational claims.

The second decision procedure. The verifier wrote `checks/decide.py` without Walnut code or Walnut data. It reads the DFAO file and builds three things. First, it builds its own Pell adder. Its state is the pending carry $c_1\alpha + c_2$ with $\alpha^2 = 2\alpha + 1$. The program bounds the carries, checks that a doubled bound gives the same automaton, and tests the adder against integer addition. Second, it builds an automaton M for the relation "$\mathbf{s}[u] \neq \mathbf{s}[u+n]$"; after determinization M has 151 states. Third, it builds an automaton for "some u with $i \leq u < i+n$ satisfies $M(u, n)$", and it searches all valid pairs (i, n) with $n \geq 1$ for one where this relation is false. Such a pair is exactly a square $\mathbf{s}[i..i+2n-1]$. The search reaches 442377 states and finds no such pair. The run takes about 6 minutes on one core.

Controls. A decision procedure that always answers "no square" would pass the main check. So `checks/controls.py` changes one transition of the DFAO of Table 1, in all 33 possible ways that keep $\delta(0, 0) = 0$. In 32 of the 33 changed DFAOs a brute-force search finds a square in the first 4000 letters. For each of these 32, `decide.py` reports a pair (i, n), and the program confirms that the reported factor is a square. In 11 of the 32 cases the reported square has period at least 2. The one remaining change, $\delta(1, 2) = 1$, computes $\mathbf{s}$ itself: a digit 2 in a valid word is always followed by a digit 0, and $\delta(0, 0) = \delta(1, 0) = 0$. The verifier also ran bounded Walnut controls: a square test restricted to $n \leq 6$ returns `FALSE` on a changed DFAO and `TRUE` on $\mathbf{s}$. An unbounded Walnut control ("no factor of exponent at least $3/2$ in $\mathbf{s}$", expected `FALSE`) did not finish: Java ran out of memory at 3 GB. That run gives no evidence for or against the result.

Prefix checks. `checks/prefix.py` computes $\mathbf{s}$ from the greedy representations and, separately, from the substitution of Lemma 8; the two agree. For each period p it finds the longest run of positions j with $z[j] = z[j-p]$, and this gives the exact largest exponent in a finite prefix z. The

largest exponent in the first 200000 letters of $h(\mathbf{s})$ is 7/3. The first factor with this exponent starts at position 8 and has length 7 and period 3. Two controls work as expected: a copy of $\mathbf{s}$ with one changed letter has a square, and the first 20000 letters of the Thue–Morse word have largest exponent 2.

What was not checked. A direct Walnut proof that $h(\mathbf{s})$ is $(7/3)^+$-free ran out of memory (Java heap at 3.2 GB). It is not needed for the proof, because Lemma 4 covers it, but it would be an independent check of Lemma 4 for this word. We did not rebuild the proofs of the four published results listed in the introduction. We read their statements in the sources.

6 Limits

- The result rests on Walnut and on `decide.py` for Proposition 3. Neither program is formally verified. The two programs share no code.
- Lemma 4 and Steps 1, 2, 3 and 5 of Section 4 use published results that we do not prove again. The proof of [1, Lemma 10] in its source is a short sketch. The same lemma carries the Fibonacci and Tribonacci theorems of [1].
- Lemma 5 uses Walnut's construction. The general fact behind it, that a first-order formula over an addable numeration system defines an automatic sequence, is the basis of Walnut.
- Our DFAO for $h(\mathbf{s})$ is not known to be the smallest Pell DFAO for a $(7/3)^+$-free binary word. A direct search for binary Pell DFAOs with at most 10 and at most 14 states did not reach a $(7/3)^+$-free prefix of length 100 in 9 minutes.
- The source asks a wider question, for "many other binary generalized automatic sequences". We answer only the Pell case. The method is the method of [1], and the new parts are the DFAO and the substitution of Lemma 8.

Data and code

The release folder has the DFAO (`data/PSQ.txt`), the Walnut commands and a script that builds Walnut from source and reruns them (`checks/walnut/`), the decision procedure, the controls and the prefix checks (`checks/`), and the search program (`checks/search/Sqf.java`).

References

[1] J.-P. Allouche, N. Rampersad, J. Shallit. Repetition threshold for binary automatic sequences. arXiv:2406.06513v3, 2026. `https://arxiv.org/abs/2406.06513`

[2] A. R. Baranwal, J. Shallit. Critical exponent of infinite balanced words via the Pell number system. In *WORDS 2019*, Lecture Notes in Comput. Sci. 11682, Springer, 2019, pp. 80–92. doi:10.1007/978-3-030-28796-2_6. arXiv:1902.00503.

[3] F. Durand. Cobham's theorem for substitutions. *J. Eur. Math. Soc.* 13 (2011), 1799–1814. doi:10.4171/JEMS/294. arXiv:1010.4009.

[4] J. Karhumäki, J. Shallit. Polynomial versus exponential growth in repetition-free binary words. *J. Combin. Theory Ser. A* 105 (2004), 335–347. doi:10.1016/j.jcta.2003.12.004. arXiv:math/0304095.

[5] H. Mousavi. Automatic theorem proving in Walnut. arXiv:1603.06017, 2016.

[6] N. Rampersad, J. Shallit, A. M. Shur. Fife's theorem for (7/3)-powers. In *WORDS 2011*, EPTCS 63 (2011), 189–198. doi:10.4204/EPTCS.63.25. arXiv:1104.3500.

[7] Walnut theorem prover, source code. `https://github.com/Walnut-Theorem-Prover/Walnut`, commit c61231a (23 November 2026).