Avoiding squares over words with lists of size three amongst four symbols

Author: Matthieu Rosenfeld

Presented by: Maciej Nemś

21st April 2022

Square word

a word of the form uu where u is a non-empty word. Period of square uu is u by period we can also mean the length $|u|$.

Square-free word

a word is square-free if it does not contain any squares.

List assignement

List assignement is a sequence $(l_i)_{i\geq 1}$ of subsets of the integers. A k-list assignement is a list assignement $(l_i)_{i\geq 1}$ such that each list is of size k, so for all $i \geq 1$, $|l_i| = k$.
We say a word $w = w_1 \dots w_n$ respects list assignement $(l_i)_{i\geq 1}$ if for all $i \in \{1, \dots, n\}$, $w_i \in l_i$.

Question

Let Σ be an infinite alphabet. Is it true that for any sequence $(l_i)_{i\geq 1}$ of subsets of Σ of size 3 there exists an infinite square-free word w respecting list assignment $(l_i)_{i\geq 1}$?

If 3 is replaced with 4 then answer to this question is **yes**.

4-list assignement

Perfect word

a perfect word is of length at least 3 and its suffix of length 3 contains 3 distinct letters. Examples:

$$abc\boldsymbol{abc}$$
$$abcdabc\boldsymbol{abd}$$

Nice word

a nice word is of length at least 3 and its suffix of length 3 is 010 up to permutation of alphabet. Examples:

$$aba\boldsymbol{aba}$$
$$abcdabc\boldsymbol{aba}$$

4-list assignement

Perfect word

a perfect word is of length at least 3 and its suffix of length 3 contains 3 distinct letters. Examples:

$$abc\boldsymbol{abc}$$
$$abcdabc\boldsymbol{abd}$$

Nice word

a nice word is of length at least 3 and its suffix of length 3 is 010 up to permutation of alphabet. Examples:

$$aba\boldsymbol{aba}$$
$$abcdabc\boldsymbol{aba}$$

All square-free words are either perfect or nice.

4-list assignement

Summing of words

For any set of words S, $\widehat{S}$ is a quantity obtained by summing the number of perfect words in S together with $\sqrt{3}-1$ times the number of nice words in S.

For $S = \{acb, aba, bab, cbc\}$ we get $\widehat{S} = 3\sqrt{3} - 2$

4-list assignement

Lemma 1

Let $l = (l_i)_{i \geq 1}$ be a 4-list assignment and let T_n be the number of square-free words of lenght n that respect l. Let $\beta > 1$ be a real number such that $1 + \sqrt{3} - \frac{1}{\beta(\beta-1)} \geq \beta$. Then for any $n \geq 3$

$$\widehat{T_{n+1}} \geq \beta \widehat{T_n}$$

4-list assignement

Lemma 1

Let $l = (l_i)_{i \geq 1}$ be a 4-list assignment and let T_n be the number of square-free words of lenght n that respect l. Let $\beta > 1$ be a real number such that $1 + \sqrt{3} - \frac{1}{\beta(\beta-1)} \geq \beta$. Then for any $n \geq 3$

$$\widehat{T_{n+1}} \geq \beta \widehat{T_n}$$

Proof

By induction on n.

Proof

A word of length $n+1$ is **good** if:

- it respects lists l
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

A word is **wrong** if it is good, but not square-free. Let G be a set of good words of length $n+1$ and F be a set of wrong words of length $n+1$.

Proof

A word of length $n+1$ is **good** if:

- it respects lists I
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

A word is **wrong** if it is good, but not square-free. Let G be a set of good words of length $n+1$ and F be a set of wrong words of length $n+1$. We get:

$$\widehat{T_{n+1}} \geq \widehat{G} - \widehat{F}$$

Proof

A word of length $n+1$ is **good** if:

- it respects lists l
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

A word is **wrong** if it is good, but not square-free. Let G be a set of good words of length $n+1$ and F be a set of wrong words of length $n+1$. We get:

$$\widehat{T_{n+1}} \geq \widehat{G} - \widehat{F}$$

We compute lower bound for $\widehat{G}$ and upper bound for $\widehat{F}$

Lower bound of $\widehat{G}$

A word of length $n+1$ is **good** if:

- it respects lists l
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

Let u be a perfect word in T_n with $u = w\boldsymbol{abc}$.

Lower bound of $\widehat{G}$

A word of length $n+1$ is **good** if:

- it respects lists l
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

Let u be a perfect word in T_n with $u = w\boldsymbol{abc}$.

If $b, c \in l_{n+1}$, u can be extanded to create 2 perfect words and 1 nice word, so we add at least $2+\sqrt{3}-1 = 1+\sqrt{3}$ to $\widehat{G}$

Lower bound of $\widehat{G}$

A word of length $n+1$ is **good** if:

- it respects lists I
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

Let u be a perfect word in T_n with $u = w\boldsymbol{abc}$.

If $b, c \in l_{n+1}$, u can be extanded to create 2 perfect words and 1 nice word, so we add at least $2 + \sqrt{3} - 1 = 1 + \sqrt{3}$ to $\widehat{G}$

If $b \notin l_{n+1} \vee c \notin l_{n+1}$, u can be extanded to create at least 3 perfect words, so we add at least 3 to $\widehat{G}$

Lower bound of $\widehat{G}$

A word of length $n+1$ is **good** if:

- it respects lists l
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

Let u be a perfect word in T_n with $u = w\boldsymbol{abc}$.

If $b, c \in l_{n+1}$, u can be extanded to create 2 perfect words and 1 nice word, so we add at least $2 + \sqrt{3} - 1 = 1 + \sqrt{3}$ to $\widehat{G}$

If $b \notin l_{n+1} \vee c \notin l_{n+1}$, u can be extanded to create at least 3 perfect words, so we add at least 3 to $\widehat{G}$

In case u is perfect we will add at least $1 + \sqrt{3}$ to $\widehat{G}$

Lower bound of $\widehat{G}$

A word of length $n+1$ is **good** if:

- it respects lists I
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

Let u be a nice word in T_n with $u = w\boldsymbol{aba}$.

Lower bound of $\widehat{G}$

A word of length $n+1$ is **good** if:

- it respects lists I
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

Let u be a nice word in T_n with $u = w\boldsymbol{aba}$.

It can be extended into at least 2 perfect words. Since contribution of a nice word to $\widehat{T_n}$ is $\sqrt{3}-1$ then the contribution of extension of a nice word to $\widehat{G}$ is at least $\frac{2}{\sqrt{3}-1} = 1+\sqrt{3}$ times as large as its contribution to $\widehat{T_n}$

Lower bound of $\widehat{G}$

A word of length $n+1$ is **good** if:

- it respects lists I
- its prefix of length n is in T_n
- it contains no square of period 1 or 2

Let u be a nice word in T_n with $u = w\boldsymbol{aba}$.

It can be extended into at least 2 perfect words. Since contribution of a nice word to $\widehat{T_n}$ is $\sqrt{3}-1$ then the contribution of extension of a nice word to $\widehat{G}$ is at least $\frac{2}{\sqrt{3}-1} = 1+\sqrt{3}$ times as large as its contribution to $\widehat{T_n}$

Because both contributions are at least $1+\sqrt{3}$ times as large we get

$$\widehat{G} \geq (1+\sqrt{3})\widehat{T_n}$$

Upper bound of $\widehat{F}$

Let F_i be a set of words from F that end with a square of period i.

$$\widehat{F} \leq \sum_{i \geq 1} \widehat{F_i}$$

Upper bound of $\widehat{F}$

Let F_i be a set of words from F that end with a square of period i.

$$\widehat{F} \leq \sum_{i \geq 1} \widehat{F_i}$$

$|F_1| = |F_2| = 0$. Let us take $i > 3$, $u \in F_i$. After removing a prefix of length i we get a square-free word of length $n + 1 - i$.
Because this prefix is in T_{n+1-i} and 3 last letters of prefix are the same as 3 last letters of u, we get:

$$\widehat{F_i} = \widehat{T_{n+1-i}}$$

Upper bound of $\widehat{F}$

Let F_i be a set of words from F that end with a square of period i.

$$\widehat{F} \leq \sum_{i \geq 1} \widehat{F_i}$$

$|F_1| = |F_2| = 0$. Let us take $i > 3$, $u \in F_i$. After removing a prefix of length i we get a square-free word of length $n + 1 - i$.
Because this prefix is in T_{n+1-i} and 3 last letters of prefix are the same as 3 last letters of u, we get:

$$\widehat{F_i} = \widehat{T_{n+1-i}}$$

We get

$$\widehat{F} \leq \sum_{i \geq 3} \widehat{F_i} = \sum_{i \geq 3} \widehat{T_{n+1-i}} \leq \sum_{i \geq 3} \frac{\widehat{T_n}\beta}{\beta^i} \leq \frac{\widehat{T_n}}{\beta(\beta - 1)}$$

We get

$$\widehat{T_{n+1}} \geq \left(1 + \sqrt{3} - \frac{1}{\beta(\beta - 1)}\right) \widehat{T_n} \geq \beta \widehat{T_n}$$

Since $\beta = 2.45$ satisfies $1 + \sqrt{3} - \frac{1}{\beta(\beta-1)} \geq \beta$ we get following theorem:

Theorem 1

Fix a 4-list assignment $(l_i)_{i \geq 1}$ and let T_n be the set of square-free words of length n that respect this assignment. Then for all $n \geq 1$,

$$|T_n| \geq 2.45^n$$

$(\Sigma, 3)$-list assignment

Let $\Sigma = \{0, 1, 2, 3\}$. An $(\Sigma, 3)$-list assignment is a problem, where Σ is the alphabet

Word is normalized if it is the smallest of all words obtained by a permutation of the alphabet.

Let Λ be the set of normalized prefixes of minimal squares of period $\leq p$.

For a word w let $\Lambda(w)$ be the longest word from Λ that is a suffix of w up to permutaion of the alphabet.

$S^{(w)} = \{u \in S : \Lambda(u) = w\}$

$S_{free}^{\leq p}$ words not containing square of period $\leq p$.

Lemma 2

There exist coefficients $(C_w)_{w\in\Lambda}$ such that $C_\varepsilon > 0$ and for all $v \in \Lambda$

$$\alpha C_v \leq \min_{I\subseteq\Sigma,|I|=3} \sum_{a\in I, va\in S_{free}^{\leq p}} C_{\Lambda(va)}$$

where $\alpha \approx 1.301$.

We denote quantity similar to quantity declared in 4-list assignment

$$\widehat{S} = \sum_{w\in\Lambda} C_w |S^{(w)}|$$

We are going to inductively count the total weight of the square-free words of size n that respect a fixed 3-list assignment.

Λ plays the same rola as $\{010, 012\}$ and α plays the same role as $1+\sqrt{3}$.

Lemma 3

Let $l = (l_i)_{i\geq 1}$ be a $(\Sigma, 3)$-list assignment and let S_n be the number of square-free words of lenght n that respect l. Let $\beta > 1$ be a real number such that $\alpha - \frac{\beta^{1-p}}{\beta-1} \geq \beta$. Then for any $n \geq 3$

$$\widehat{S_{n+1}} \geq \beta \widehat{S_n}$$

Similar proof as for 4-list assignment.

A word of length $n+1$ is **good** if:

- it respects lists l
- its prefix of length n is in S_n
- it contains no square of period $\leq p$

A word is wrong if it is not square-free. We compute

$$\widehat{S_{n+1}} \geq \widehat{G} - \widehat{F}$$

Lower bound of $\widehat{G}$

A word of length $n+1$ is **good** if:

- it respects lists l
- its prefix of length n is in S_n
- it contains no square of period $\leq p$

A word is wrong if it is not square-free.

We want to extend v to get some $va \in S_{free}^{\leq p}$. It is possible if and only if $\Lambda(v)a \in S_{free}^{\leq p}$. Also $\Lambda(va) = \Lambda(\Lambda(v)a)$

Lower bound of $\widehat{G}$

$$\sum_{\substack{a\in\ell_{n+1}\\ va\in\mathcal{S}^{\leq p}_{free}}} C_{\Lambda(va)} = \sum_{\substack{a\in\ell_{n+1}\\ \Lambda(v)a\in\mathcal{S}^{\leq p}_{free}}} C_{\Lambda(\Lambda(v)a)} \geq \min_{\substack{l\subseteq\mathcal{A}\\ |l|=3}} \sum_{\substack{a\in l\\ \Lambda(v)a\in\mathcal{S}^{\leq p}_{free}}} C_{\Lambda(\Lambda(v)a)}\,.$$

$$\widehat{G} \geq \sum_{v\in S_n} \alpha C_{\Lambda(v)} = \sum_{u\in\Lambda} \alpha C_u |S_n^{(u)}| = \alpha\widehat{S_n}\,.$$

Upper bound of $\widehat{F}$

$$\widehat{F}_i \leq \widehat{S_{n+1-i}} \leq \widehat{S_n}\beta^{1-i}$$

$$\widehat{F} \leq \sum_{i \geq p+1} \widehat{S_n}\beta^{1-i} \leq \widehat{S_n}\frac{\beta^{1-p}}{\beta - 1}$$

$$\widehat{S_{n+a}} \geq \widehat{G} - \widehat{F} \geq \alpha\widehat{S_n} - \frac{\beta^{1-p}}{\beta - 1}\widehat{S_n} \geq \beta\widehat{S_n}$$

$\beta = 1.25$ satisfies this.

Theorem 2

Fix a $(\Sigma, 3)$-list assignment $(l_i)_{i \geq 1}$ and let S_n be the set of square-free words of length n that respect this assignment. Then for all $n \geq 1$,

$$|S_n| \geq 1.25^n$$

How to prove lemma 2?

Lemma 2

There exist coefficients $(C_w)_{w\in\Lambda}$ such that $C_\varepsilon > 0$ and for all $v \in \Lambda$

$$\alpha C_v \leq \min_{I\subseteq\Sigma,|I|=3} \sum_{a\in I, va\in S_{free}^{\leq p}} C_{\Lambda(va)}$$

where $\alpha \approx 1.301$.

We start with a random vector of $(C_w)_{w\in\Lambda}$. We iterate until vector converges towards the desired vector. Convergence of this algorithm is not proven. Authors suspect that the procedure is convergent but did not try to prove anything in this directions.

"The C++ implementation can be found in the ancillary file on the arXiv. Running this program took approximately 2 hours of computation and occupied 76.4 Go of RAM."

Bibliography

M. Rosenfeld.
Avoiding squares over words with lists of size three amongst four symbols, 2021.