

# Lower Bounds for the Number of Repetitions in 2D Strings

Paweł Gawrychowski<sup>\*1</sup>, Samah Ghazawi<sup>†2</sup>, and Gad M. Landau<sup>†1</sup>

<sup>1</sup>Institute of Computer Science, University of Wrocław, Poland

<sup>2</sup>Department of Computer Science, University of Haifa, Israel

<sup>3</sup>NYU Tandon School of Engineering, New York University, Brooklyn, NY, USA

## Abstract

A two-dimensional string is simply a two-dimensional array. We continue the study of the combinatorial properties of repetitions in such strings over the binary alphabet, namely the number of distinct tandems, distinct quartics, and runs. First, we construct an infinite family of  $n \times n$  2D strings with  $\Omega(n^3)$  distinct tandems. Second, we construct an infinite family of  $n \times n$  2D strings with  $\Omega(n^2 \log n)$  distinct quartics. Third, we construct an infinite family of  $n \times n$  2D strings with  $\Omega(n^2 \log n)$  runs. This resolves an open question of Charalampopoulos, Radoszewski, Rytter, Waleń, and Zuba [ESA 2020], who asked if the number of distinct quartics and runs in an  $n \times n$  2D string is  $\mathcal{O}(n^2)$ .

## 1 Introduction

The study of repetitions in strings goes back at least to the work of Thue from 1906 [35], who constructed an infinite square-free word over the ternary alphabet. Since then, multiple definitions of repetitions have been proposed and studied, with the basic question being focused on analyzing how many such repetitions a string of length  $n$  can contain. The most natural definition is perhaps that of palindromes, which are fragments that read the same either from left to right or right to left. Of course, any fragment of the string  $\mathbf{a}^n$  is a palindrome, therefore we would like to count distinct palindromes. An elegant folklore argument shows that this is at most  $n + 1$  for any string of length  $n$  [19], which is attained by  $\mathbf{a}^n$ .

Another natural definition is that of squares, which are fragments of the form  $xx$ , where  $x$  is a string. Again, because of the string  $\mathbf{a}^n$  we would like to count distinct squares. Using a combinatorial result of Crochemore and Rytter [16], Fraenkel and Simpson [21] proved that a string of length  $n$  contains at most  $2n$  distinct squares (see also [25] for a simpler proof, and [26] for an upper bound of  $2n - \Theta(\log n)$ ). They also provided an infinite family of strings of length  $n$  with  $n - o(n)$  distinct squares. It is conjectured that the right upper bound is actually  $n$ , however so far we only know that it is at most  $11/6n$  [18]. Interestingly, a proof of the conjecture for the binary alphabet would imply it for any alphabet [29].

Perhaps a bit less natural, but with multiple interesting applications, is the definition of runs. A run is a maximal periodic fragment that is at least twice as long as its smallest period. Roughly

---

<sup>\*</sup>Partially supported by the Bekker programme of the Polish National Agency for Academic Exchange (PPN/BEK/2020/1/00444).

<sup>†</sup>Partially supported by the Israel Science Foundation grant 1475/18, and Grant No. 2018141 from the United States-Israel Binational Science Foundation (BSF).

speaking, runs capture all the repetitive structure of a string, making them particularly useful when constructing algorithms [15]. A well-known result by Kolpakov and Kucherov [28] is that a string of length  $n$  contains  $\mathcal{O}(n)$  runs; they conjectured that it is actually at most  $n$ . After a series of improvements [12, 31, 32], with the help of an extensive computer search the upper bound was decreased to  $1.029n$  [13, 24]. Finally, in a remarkable breakthrough Bannai et al. [9] confirmed the conjecture. On the lower bound side, we current know an infinite family of strings with at least  $0.944575712n$  runs [22, 30, 33]. Interestingly, better bounds are known for the binary alphabet [20].

Given that we seem to have a reasonably good understanding of repetitions in strings, it is natural to consider repetitions in more complex structures, such as circular strings [7, 17, 34] or trees [14, 23, 27]. In this paper, we are interested in repetitions in 2D strings. Naturally, algorithms operating on 2D strings can be used for image processing, and combinatorial properties of such strings can be used for designing efficient pattern matching algorithms [1–4, 11]. Therefore, we would like to fully understand what is a repetition in a 2D string, and what is the combinatorial structure of such repetition.

Apostolico and Brimkov [8] introduced the notions of tandems and quartics in 2D strings. Intuitively, a tandem consists of two occurrences of the same block  $W$  arranged in a  $1 \times 2$  or  $2 \times 1$  pattern, while a quartic consists of 4 occurrences of the same block  $W$  arranged in a  $2 \times 2$  pattern. They considered tandems and quartics with a primitive  $W$ , meaning that it cannot be partitioned into multiple occurrences of the same  $W'$  (called primitively rooted in the subsequent work [10]), and obtained asymptotically tight bounds of  $\Theta(n^2 \log^2 n)$  and  $\Theta(n^2 \log n)$  for the number of such tandems and quartics in an  $n \times n$  2D string, respectively. Both tandems and quartics should be seen as an attempt to extend the notion of squares in a 1D string to 2D strings, and thus the natural next step is to consider distinct tandems and quartics (without restricting  $W$  to be primitive). Very recently, Charalampopoulos et al. [10] studied the number of distinct tandems and quartics in an  $n \times n$  2D string. For distinct tandems, they showed a tight bound of  $\Theta(n^3)$  with the construction used in the lower bound using an alphabet of size  $n$ . For distinct quartics, they showed an upper bound of  $\mathcal{O}(n^2 \log^2 n)$  and conjectured that it is always  $\mathcal{O}(n^2)$ , similarly to the number of distinct squares in a 1D string of length  $n$  being  $\mathcal{O}(n)$ .

Amir et al. [5, 6] introduced the notion of runs in 2D strings. Intuitively, a 2D run is a maximal subarray that is both horizontally and vertically periodic; we defer a formal definition to the next section. They proved that an  $n \times n$  2D string contains  $\mathcal{O}(n^3)$  runs, showing an infinite family of  $n \times n$  2D strings with  $\Omega(n^2)$  runs. Later, Charalampopoulos et al. [10] significantly improved on this upper bound, showing that an  $n \times n$  2D string contains  $\mathcal{O}(n^2 \log^2 n)$  runs, and conjectured that it is always  $\mathcal{O}(n^2)$ , similarly to the number of runs in a 1D string of length  $n$  being  $\mathcal{O}(n)$ .

**Our results.** In this paper, we consider 2D strings and obtain improved lower bounds for the number of distinct tandems, distinct quartics, and runs. We start with the number of distinct tandems and extend the lower bound of Charalampopoulos et al. [10] over the binary alphabet in Section 3 by showing the following.

**Theorem 1.1.** *There exists an infinite family of  $n \times n$  2D strings over the binary alphabet containing  $\Omega(n^3)$  distinct tandems.*

Then, we move to the number of distinct quartics in Section 4 and the number of runs in Section 5, and show the following.

**Theorem 1.2.** *There exists an infinite family of  $n \times n$  2D strings over the binary alphabet containing  $\Omega(n^2 \log n)$  distinct quartics.*

**Theorem 1.3.** *There exists an infinite family of  $n \times n$  2D strings over the binary alphabet containing  $\Omega(n^2 \log n)$  runs.*

By the above theorem, the algorithm of Amir et al. [6] for locating all 2D runs in  $\mathcal{O}(n^2 \log n + \text{output})$  time is worst-case optimal.

Our constructions exhibit a qualitative difference between distinct squares and runs in 1D strings and distinct quartics and runs in 2D strings. The number of the former is linear in the size of the input, while the number of the latter, surprisingly, is superlinear.

**Our techniques.** For distinct tandems, our construction is similar to that of [10], except that we use distinct characters only in two columns. This allows us to replace them by their binary expansions, with some extra care as to not lose any counted tandems.

For both distinct quartics and runs, we proceed recursively, constructing larger and larger 2D strings  $A_i$  starting from the initial 2D string  $A_1$ . The high-level ideas behind both constructions are different, though.

For distinct quartics, our high-level idea is to consider subarrays with  $\Theta(\log n)$  different aspect ratios. For each such aspect ratio, we create  $\Omega(n^2)$  distinct quartics, for an  $n \times n$  array. Each step of the recursion corresponds to a different aspect ratio and creates multiple new special characters, as to make the new quartics distinct; later we show how to implement this kind of approach with the binary alphabet.

For runs, we directly proceed with a construction for the binary alphabet, and build on the insight used by Charalampopoulos et al. [10] to show that the same quartic can be induced by  $\Theta(n^2)$  runs. Each step of the recursion corresponds to runs with asymptotically the same size. This needs to be carefully analyzed in order to lower bound the overall number of runs.

## 2 Preliminaries

Let  $\Sigma$  be a fixed finite alphabet. A two-dimensional string (or 2D string, for short) over  $\Sigma$  is an  $m \times n$  array  $A[0..m-1][0..n-1]$  with  $m$  rows and  $n$  columns, with every cell  $A[i][j]$  containing an element of  $\Sigma$ . Furthermore, we use  $\epsilon$  to denote an empty 2D string. A subarray  $A[x_1..x_2][y_1..y_2]$  of  $A[0..m-1][0..n-1]$  is an  $(x_2 - x_1 + 1) \times (y_2 - y_1 + 1)$  array consisting of cells  $A[i][j]$  with  $i \in [x_1, x_2], j \in [y_1, y_2]$ .

We consider three notions of repetitions in 2D strings.

**Tandem.** A subarray  $T$  of  $A$  is a tandem if it consists of  $2 \times 1$  (or  $1 \times 2$ ) subarrays  $W \neq \epsilon$ . Two tandems  $T = \begin{bmatrix} W & W \end{bmatrix}$  and  $T' = \begin{bmatrix} W' & W' \end{bmatrix}$  are distinct when  $W \neq W'$ .

**Quartic.** A subarray  $Q$  of  $A$  is a quartic if it consists of  $2 \times 2$  subarrays  $W \neq \epsilon$ . Two quartics

$$Q = \begin{bmatrix} W & W \\ W & W \end{bmatrix} \text{ and } Q' = \begin{bmatrix} W' & W' \\ W' & W' \end{bmatrix} \text{ are distinct when } W \neq W'.$$

**Run.** Consider an  $r \times c$  subarray  $R$  of  $A$ . We define a positive integer  $p$  to be its horizontal period if the  $i^{\text{th}}$  column of  $R$  is equal to the  $(i+p)^{\text{th}}$  column of  $R$ , for all  $i = 1, 2, \dots, c-p$ . The horizontal period of  $R$  is its smallest horizontal period, and we say that  $R$  is  $h$ -periodic when its horizontal period is at most  $c/2$ . Similarly, we define a vertical period, the vertical period, and a  $v$ -periodic subarray. An  $h$ -periodic and  $v$ -periodic  $R$  is called a run when extending  $R$  in any direction would result in a subarray with a larger horizontal or vertical period. Informally, for such  $R$  there exists a subarray  $W$  such that we can represent  $R$  as follows,

with at least two repetitions of  $W$  in both directions, and we cannot extend  $R$  in any direction while maintaining this property.

$$R = \begin{array}{|c|c|c|c|} \hline W & \dots & W & W' \\ \hline \dots & \dots & \dots & \dots \\ \hline W & \dots & W & W' \\ \hline W'' & \dots & W'' & W''' \\ \hline \end{array}$$

Where  $W = \begin{bmatrix} W' & U \end{bmatrix}$ ,  $W = \begin{bmatrix} W'' \\ U' \end{bmatrix}$  and  $W = \begin{bmatrix} W''' & V \\ V' & V''' \end{bmatrix}$ , and any of the subarrays  $W', W'', W''', U, U', V, V'$  and  $V'''$  may be  $\epsilon$ .

### 3 Distinct Tandems

In this section, we show how to construct an  $n \times n$  array  $A$  over the binary alphabet with  $\Theta(n^3)$  distinct tandems, for any  $\ell \geq 1$ , where  $n = 3 \cdot 2^\ell + 2\ell$ . The  $i^{\text{th}}$  row of  $A$  is divided into 5 parts, see Figure 1. The first, third, and fifth part each consists of  $2^\ell$  cells, each containing the binary representation of 1. The second and fourth part each consists of  $\ell$  cells that contains the binary representation of the number  $i - 1$ . Hence, all rows of  $A$  are different, see Figure 2.

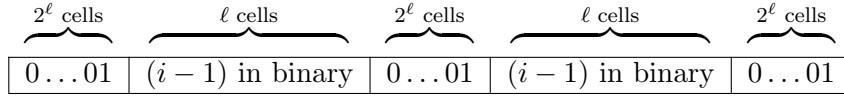


Figure 1: The  $i^{\text{th}}$  row of  $A$ .

**Theorem 1.1.** *There exists an infinite family of  $n \times n$  2D strings over the binary alphabet containing  $\Omega(n^3)$  distinct tandems.*

*Proof.* To lower bound the number of tandems in  $A$ , consider any  $1 \leq i \leq j \leq n$  and  $k \in \{1, 2, \dots, 2^\ell\}$ . Then, let  $T$  be the subarray of width  $2(2^\ell + \ell)$  starting in the  $i^{\text{th}}$  row and ending in the  $j^{\text{th}}$  row with the top left cell of  $T$  being  $A[i][k]$ . We claim that for each choice of  $i, j, k$  we obtain a distinct tandem, making the number of distinct tandems in  $A$  at least  $n^2 \cdot 2^\ell = \Omega(n^3)$ . It is clear that each such  $T$  is a tandem. To prove that all of them are distinct, consider any such  $T$ . The position of the leftmost 1 in its top row allows us to recover the value of  $k$ . Then, the next  $\ell$  cells contain the binary expansion of  $(i - 1)$ , so we can recover  $i$ . Finally, the height of  $T$  together with  $i$  allows us to recover  $j$ . Thus, we can uniquely recover  $i, j, k$  from  $T$ , and all such tandems are distinct.  $\square$

### 4 Distinct Quartics

In this section, we show how to construct an  $n \times n$  array  $A_\ell$  with  $\Omega(n^2 \log n)$  distinct quartics, for any  $\ell \geq 1$ , where  $n = 3^\ell - 1$ . The construction is recursive, that is, we construct a series of arrays  $A_1, A_2, \dots, A_\ell$ , with  $A_i$  being defined using  $A_{i-1}$ . The number of columns of each array  $A_i$  is the same and equal to  $n$ . The number of rows is increasing, starting with 2 rows in  $A_1$  and ending with  $n$  rows in the final array  $A_\ell$ . We provide the details of the construction in the next subsection, then analyze the number of distinct quartics in  $A_\ell$  in the subsequent subsection. Finally, in the



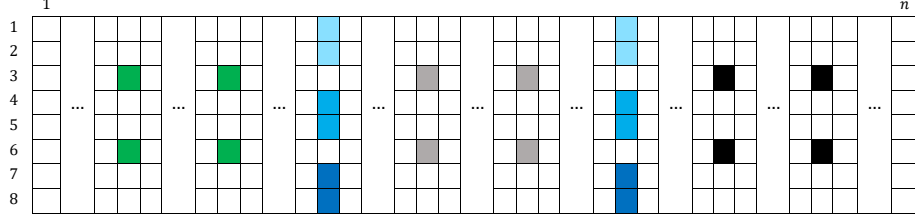


Figure 4: Array  $A_2$ , where rows 1-2 are the first copy of  $A_1$ , rows 4-5 are the second copy of  $A_1$ , and rows 7-8 are the third copy of  $A_1$ . Rows 3 and 6 are the separating rows. Each color corresponds to a different special character.

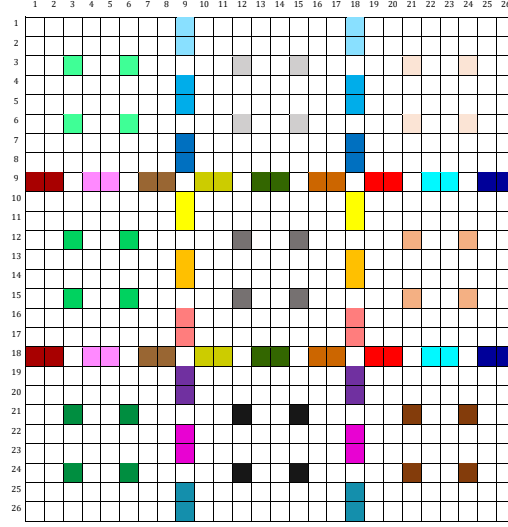


Figure 5: Array  $A_3$  includes 3 copies of  $A_2$  in rows 1-8, 10-17 and 19-26. The separating rows are 9 and 18. Note that  $A_3$  is the final array for  $n = 26$ . Additionally, each color represent a different special character, in total 27 special characters are used in  $A_3$ .

We now analyze the number of distinct quartics in each  $A_i$ . We will be only counting some of them, and denote by  $Q_i$  the distinct quartics counted in the following argument that, similarly to the construction, considers first  $i = 1$  and then the general case.

For  $i = 1$ , we count distinct quartics that contain special characters. Each of them is of width  $2(\frac{n-2}{3} + 1)$  and height 2. There are  $Q_1 = \frac{n-2}{3} + 1 = \frac{n+1}{3}$  such quartics and all of them are distinct, see Figure 6.

For the general case of  $i \geq 2$ , we consider two groups of distinct quartics. The first group consists of distinct quartics contained in the copies of  $A_{i-1}$ . For each of the  $3^{i-1}$  maximal range of  $N_{i-1}$  columns of  $A_{i-1}$  consisting of 0s, the second group consists of all possible  $(2M_{i-1} + 2) \times \frac{2N_{i-1} + 1}{3}$  subarrays contained in that range. For each such range, we have  $\frac{N_{i-1}-2}{3} + 1$  possible horizontal shifts and  $M_{i-1} + 1$  possible vertical shifts and for each of them we obtain a distinct quartic containing the new special character created for the range. As we use different special characters in every copy of  $A_{i-1}$  and, for every range, in the separating rows of  $A_i$ , overall we have at least  $Q_i = 3Q_{i-1} + 3^{i-1}(\frac{N_{i-1}-2}{3} + 1)(M_{i-1} + 1)$  distinct quartics. See Figure 7 for an illustration with  $i = 2$ .

Substituting the formulas for  $N_{i-1}$  and  $M_{i-1}$ , we conclude that  $Q_1 = \frac{n+1}{3}$  and  $Q_i = 3Q_{i-1} +$

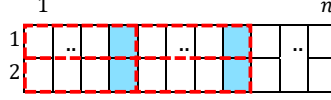


Figure 6: Array  $A_1$ , where the red border corresponds to the leftmost quartic that contains the special character cells, and by shifting it to the right we obtain distinct quartics.

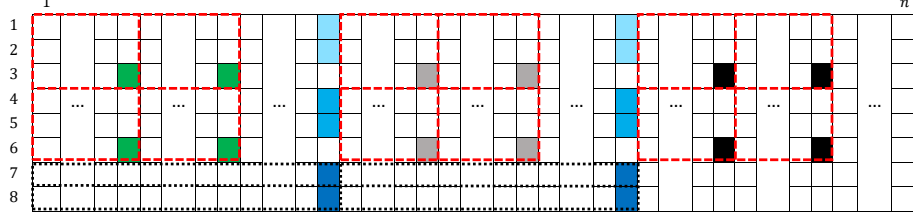


Figure 7: Array  $A_2$ , where the black border corresponds to the leftmost quartic in the third copy of  $A_1$ , and by shifting it to the right we obtain distinct quartics. The red borders correspond to the leftmost quartic in each of the 3 maximal ranges of columns of  $A_1$  consisting of 0s; by shifting each of them to the right and down we obtain distinct quartics.

$3^{i-2}(n+1)$  for  $i \geq 2$ . Unwinding the recurrence, we obtain that  $Q_i = 3^{i-1}Q_1 + (i-1)3^{i-2}(n+1) = 3^{i-1}\frac{n+1}{3} + (i-1)3^{i-2}(n+1)$ . Therefore,  $Q_i = 3^{i-2}i(n+1)$ .

**Theorem 4.1.** *There exists an infinite family of  $n \times n$  2D strings containing  $\Omega(n^2 \log n)$  distinct quartics.*

*Proof.* For each  $\ell \geq 1$ , we take  $n = 3^\ell - 1$  and define arrays  $A_1, A_2, \dots, A_\ell$  as described above. The final array  $A_\ell$  consists of  $M_\ell = n$  rows and  $n$  columns, and contains at least  $Q_\ell = 3^{\ell-2}\ell(n+1)$  distinct quartics, which is  $\Omega(n^2 \log n)$ .  $\square$

While we were not concerned with the size of the alphabet in this construction, observe that the number of distinct special characters  $S_i$  in  $A_i$  is described by the recurrences  $S_1 = 1$  and  $S_i = 3S_{i-1} + 3^{i-1}$  for  $i \geq 2$ . This is because we are using new special characters in each copy of  $A_{i-1}$  and adding  $3^{i-1}$  new special characters to divide the maximal ranges of  $N_{i-1}$  columns into 3 parts. Therefore, the size of the alphabet used to construct  $A_\ell$  is  $S_\ell = \frac{(n+1)\log(n+1)}{3} + 1$ .

### 4.3 Reducing Alphabet

In this subsection, we show how to modify the array  $A_\ell$  to obtain an array  $A'_\ell$  over the binary alphabet, for any  $\ell \geq 4$ . Informally speaking, we will replace each special character by a small gadget encoding its binary representation, and then carefully revise the parameters of the new construction, particularly the number of distinct quartics.

Let  $\Sigma = \{1, 2, \dots, \sigma\}$  be the alphabet used to construct  $A_\ell$ , where  $\sigma = \frac{(n+1)\log(n+1)}{3} + 1$ . We define arrays  $B_1, B_2, \dots, B_\sigma$  of the same size  $k \times k$ , where  $k = \sqrt{\log \sigma} + 2$ . The first row and column of every array  $B_c$  contain only 0s, while the remaining cells of the last row and column contain only 1s. The concatenation of cells from the middle of  $B_c$  (without the first and last row and column), in the left-right top-bottom order, should be equal to the binary representation of  $c$ . Now, we construct the array  $A'_\ell$  from the array  $A_\ell$  by repeating the recursive construction of arrays



$A_1, A_2, \dots, A_\ell$ , but replacing a cell containing the character  $c$  with the array  $B_c$ . We denote the resulting arrays  $A'_1, A'_2, \dots, A'_\ell$ .

We now set  $n' = n \cdot k$ . Each of the arrays  $A'_i$  consists of  $n'$  columns and  $M_i \cdot k$  rows, so the final array,  $A'_\ell$ , is of size  $n' \times n'$ . We now analyze the number of distinct quartics in  $A'_i$ . This will be done similarly as it was for  $A_i$ , but we must be more careful about arguing quartics as being distinct, because we no longer have multiple distinct special characters. We first argue that, for all sufficiently wide and tall subarrays of  $R$ , the horizontal and vertical shifts are uniquely defined modulo  $k$ .

**Lemma 4.2.** *Consider a subarray  $R = A'_i[x_1..x_2][y_1..y_2]$  with width and height at least  $k$ . Then  $(x_1 \bmod k)$  and  $(y_1 \bmod k)$  can be recovered from  $R$ .*

*Proof.* We only analyze how to recover  $(y_1 \bmod k)$ , recovering  $(x_1 \bmod k)$  is symmetric. By construction of  $B_1, B_2, \dots, B_\sigma$ , every  $k^{\text{th}}$  row of  $A'_i$  consists of only 0s, while in every other row there is at least one 1 in every block of  $k$  cells. Therefore, because the width of  $R$  is at least  $k$ , a row of  $R$  consists of 0s if and only if it is aligned with a row of  $A'_i$  that consists of 0s. Because the height of  $R$  is at least  $k$  such a row surely exists and allows us to recover  $(y_1 \bmod k)$ .  $\square$

We argue that the number of distinct quartics in  $A'_1$  is at least  $Q'_1 = \frac{n \cdot k - 2k}{3} + 1$ . To show this, we consider subarrays spanning the whole height of  $A'_1$  and of width  $2(\frac{n \cdot k - 2k}{3} + k)$ . There are  $\frac{n \cdot k - 2k}{3} + 1$  such subarrays and each of them is a quartic that fully contains some  $B_c$ . Furthermore, subarrays starting in columns with different remainders modulo  $k$  are distinct by Lemma 4.2. Subarrays starting in columns with the same remainder modulo  $k$  are also distinct, as in such a case we can recover the special character from  $B_c$  fully contained in the subarray.

For the general case, we claim that the number of distinct quartics in  $A'_i$  is at least  $Q'_i = 3Q'_{i-1} + 3^{i-1}(\frac{N_{i-1} \cdot k - 2k}{3} + 1)(M_{i-1} \cdot k + 1)$  for  $i \geq 2$ . The argument proceeds as for  $A_i$ ; however, we must argue that the counted quartics are all distinct. By construction, each of them fully contains some  $B_c$ . Thus, quartics starting in columns with different remainders modulo  $k$  (and also in rows with different remainders modulo  $k$ ) are distinct by Lemma 4.2. Now consider all counted quartics starting in columns with remainder  $y$  modulo  $k$  and rows with remainder  $x$  modulo  $k$ . For each of them, we can recover the special character from  $B_c$  fully contained in the quartic, so all of them are distinct.

Finally, we lower bound and solve the recurrence for  $Q'_i$  as follows.

$$\begin{aligned}
Q'_i &= 3Q'_{i-1} + 3^{i-1}(\frac{N_{i-1} \cdot k - 2k}{3} + 1)(M_{i-1} \cdot k + 1) \\
&> 3Q'_{i-1} + 3^{i-2} \cdot k^2(N_{i-1} - 2)M_{i-1} \\
&= 3Q'_{i-1} + 3^{i-2} \cdot k^2(\frac{n+1}{3^{i-1}} - 3)(3^{i-1} - 1) \\
&= 3Q'_{i-1} + 3^{i-2} \cdot k^2(n - 3^i) \frac{3^{i-1} - 1}{3^{i-1}} \\
&> 3Q'_{i-1} + 3^{i-3} \cdot k^2(n - 3^i) \quad \text{using } i \geq 2.
\end{aligned}$$

Unwinding the recurrence, we obtain that  $Q'_i > \sum_{j=2}^i 3^{i-j} \cdot 3^{j-3} \cdot k^2(n - 3^j) > 3^{i-3} \cdot k^2((i-1)n - \frac{3^i}{2})$ .

**Theorem 1.2.** *There exists an infinite family of  $n \times n$  2D strings over the binary alphabet containing  $\Omega(n^2 \log n)$  distinct quartics.*



*Proof.* For each  $\ell \geq 4$ , we take  $n = 3^\ell - 1$  and define arrays  $A'_1, A'_2, \dots, A'_\ell$  as described above. The final array  $A'_\ell$  is over the binary alphabet by construction, consists of  $n'$  rows and  $n'$  columns, where  $n' = n \cdot k$ , and contains at least  $Q'_\ell$  distinct quartics. Finally,

$$\begin{aligned}
Q'_\ell &> 3^{\ell-3} \cdot k^2 \cdot ((\ell-1)n - \frac{3^\ell}{2}) \\
&= 3^{\ell-3} \cdot k^2 \cdot ((\ell-1)(3^\ell - 1) - \frac{3^\ell}{2}) \\
&> 3^{\ell-3} \cdot k^2 \cdot (\ell-2)(3^\ell - 1) && \text{because } \frac{3^\ell}{2} < 3^\ell - 1 \\
&\geq 3^{\ell-3} \cdot k^2 \cdot \frac{\ell}{2} \cdot (3^\ell - 1) && \text{because } \ell-2 \geq \frac{\ell}{2}.
\end{aligned}$$

Therefore, the number of runs in  $A'_\ell$  is  $\Omega(3^{2\ell} \cdot k^2 \cdot \ell) = \Omega(n'^2 \log n')$ .  $\square$

## 5 Runs

In this section, we show how to construct an  $n \times n$  array  $A_\ell$  with  $\Omega(n^2 \log n)$  runs, for any  $\ell \geq 2$ , where  $n = 2 \cdot 4^\ell$ . As in the previous section, the construction is recursive, i.e., we construct a series of arrays  $A_1, A_2, \dots, A_\ell$ , with  $A_i$  being defined using  $A_{i-1}$ . Both the number of rows and columns in  $A_i$  is equal to  $2 \cdot 4^i$ , starting with 8 rows and columns in  $A_1$ . We describe the construction in the next subsection, then analyze the number of runs in  $A_\ell$  in the subsequent subsection.

### 5.1 Construction

First, we provide array  $A_1$  of size  $8 \times 8$  with 1s in the cells  $A_1[1][2]$ ,  $A_1[2][1]$ ,  $A_1[7][8]$  and  $A_1[8][7]$ , and 0s in the other cells, see Figure 8 (left).

Second, we obtain array  $A_i$  by concatenating  $4 \times 4$  copies of array  $A_{i-1}$  while using 1s to fill the antidiagonals in the upper left and bottom right copy of  $A_{i-1}$ , with  $A'_{i-1}$  denoting such modified copy of  $A_{i-1}$ , see Figure 8 (right) for an illustration with  $i = 2$  and Figure 9 for an example with  $n = 128$  and  $A_3$  being the final array.

The intuition behind the recursive construction is to duplicate the runs obtained in the previous arrays. For example, the array  $A_1$  produces one run that does not touch the boundaries. This is

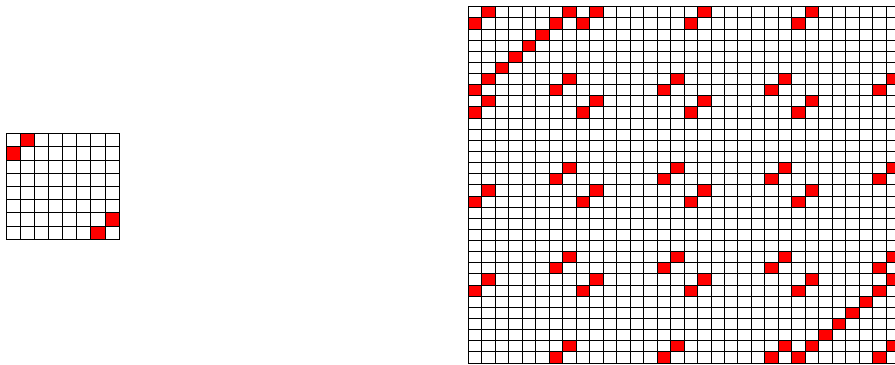


Figure 8: Left: array  $A_1$ , where red cells contain 1s and white cells contain 0s. Right: Array  $A_2$  consists of 14 copies of  $A_1$  and 2 copies of  $A'_1$ . Red cells include 1s and white cells include 0s. Red is used to fill the antidiagonals of  $A'_1$ .

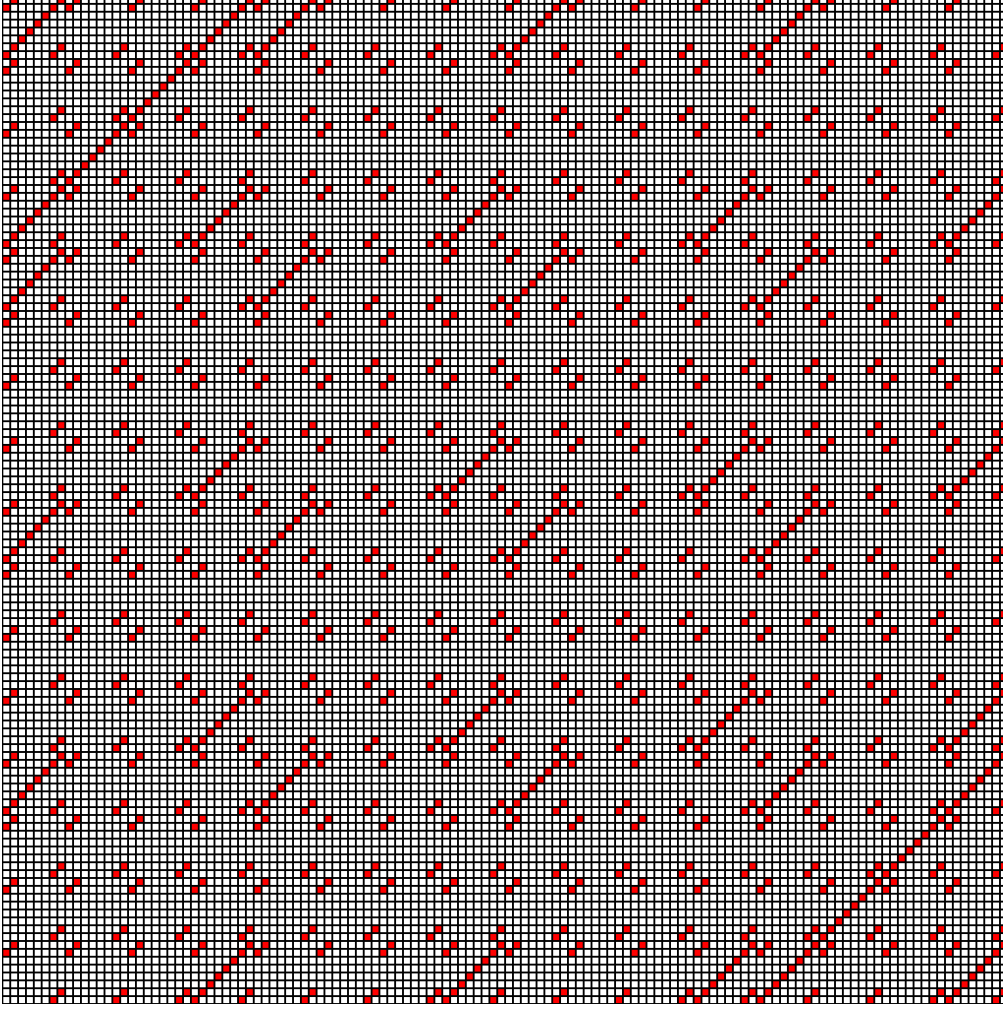


Figure 9: Array  $A_3$  includes 14 copies of  $A_2$ , 2 copies of  $A'_2$ , 216 copies of  $A_1$  and 40 copies of  $A'_1$ . Note that,  $A_3$  is the final array for  $n = 128$ .

uplicated 14 times in  $A_2$ , hence,  $A_1$  contributes 14 runs to the total number of runs produced by  $A_2$ . Moreover, the intuition behind filling the antidiagonals is to produce new runs such that the number of the new runs is equal to the size of the array up to some constant. As an example,  $A_2$  produces  $7^2$  new runs between the antidiagonals of  $A'_1$  such that the upper left and the bottom right corners of each run touch exactly two cells of the antidiagonals of the two copies of  $A'_1$ . See Figure 10 for an illustration. Therefore, overall the number of runs produced by  $A_2$  is  $14 + 7^2 = 63$ . The general case is analyzed in detail in the next subsection.

## 5.2 Analysis

The number  $N_i$  of rows and columns in  $A_i$  is described by the recurrence  $N_1 = 8$  and  $N_i = 4N_{i-1}$  for  $i \geq 2$ , so  $N_i = 2 \cdot 4^i$ . By straightforward induction, the antidiagonal of every  $A_i$  is filled with 0s.

We analyze the number of runs in  $A_i$ . First, we have  $R_i$  new runs not contained in any of the copies of  $A_{i-1}$  or  $A'_{i-1}$  such that the upper left and the bottom right corners touch exactly two cells of the antidiagonals of the two copies of  $A'_{i-1}$ .

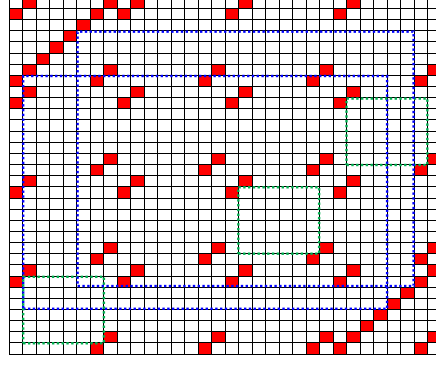


Figure 10: Array  $A_2$ , where the blue borders correspond to new runs produced by  $A_2$ . The green borders correspond to runs previously produced by  $A_1$ .

**Lemma 5.1.**  $R_i = (2 \cdot 4^{i-1} - 1)^2 = \frac{16^i}{4} - 4^i + 1$ .

*Proof.* Consider any subarray  $R$  of  $A_i$  with the upper left and the bottom right corners touching exactly two cells of the antidiagonals of the two copies  $A'_{i-1}$ . It is easy to verify that  $N_{i-1}$  is a horizontal and a vertical period of  $R$ . Therefore,  $R$  is  $h$ -periodic and  $v$ -periodic. Now consider extending  $R$  in any direction, say by one column to the left. Then the topmost cell of the new column would contain a 1 from the antidiagonal of  $A'_{i-1}$ . For the horizontal period of the extended array to remain  $N_{i-1}$  we would need a 1 in the corresponding cell of the antidiagonal of  $A_{i-1}$ , but that cell contains a 0, a contradiction. Therefore, any such  $R$  is a run. The number of such subarrays is  $(N_{i-1} - 1)^2 = (2 \cdot 4^{i-1} - 1)^2 = \frac{16^i}{4} - 4^i + 1$ , because we have  $(N_{i-1} - 1)$  possibilities for choosing the upper left and bottom right corner.  $\square$

Second, we have the runs contained in the 14 copies of  $A_{i-1}$ , hence  $A_{i-1}$  contributes  $14 \cdot R_{i-1}$  to the total number of runs in  $A_i$ . Moreover, whenever  $A_{i-1}$  contains a copy of  $A_j$ , for some  $j < i - 1$ , all new runs of  $A_j$  are preserved in  $A_{i-1}$  and consequently in  $A_i$ . Additionally, we have the two copies of  $A'_{i-1}$ . Because we have filled their antidiagonals with 1s, we lose some of the runs. However, whenever  $A'_{i-1}$  contains a copy of  $A_j$  that does not intersect the antidiagonal, for some  $j < i - 1$ , all new runs of  $A_j$  are preserved in  $A'_{i-1}$  and consequently in  $A_i$ . For example, each copy of  $A_{i-1}$  contains 14 copies of  $A_{i-2}$  and each copy of  $A'_{i-1}$  contains 10 copies of  $A_{i-2}$  (5 above and 5 below the antidiagonal). Hence,  $A_i$  contains  $14 \cdot 14 + 2 \cdot 10 = 216$  copies of  $A_{i-2}$ , thus  $A_{i-2}$  contributes  $216 \cdot R_{i-2}$  to the total number of runs in  $A_i$ . Therefore, in order to count the total number of runs in the final array  $A_\ell$ , we need to analyze how many copies of  $A_i$  are in  $A_\ell$ , for  $1 \leq i \leq \ell$ .

Let  $X_i$  denote the number of copies of  $A_i$  in  $A_\ell$ , and  $Y_i$  denote the number of copies of  $A'_i$  in  $A_\ell$ . By construction,  $A_i$  consists of 14 copies of  $A_{i-1}$  and 2 copies of  $A'_{i-1}$ . Similarly,  $A'_i$  consists of 10 copies of  $A_{i-1}$  (5 above and 5 below the antidiagonal) and 6 copies of  $A'_{i-1}$  (4 intersecting the antidiagonal and the top left and bottom right copy). Consequently, we obtain the recurrences  $X_\ell = 1$  and  $X_i = 14X_{i+1} + 10Y_{i+1}$  for  $i < \ell$ ,  $Y_\ell = 0$  and  $Y_i = 6Y_{i+1} + 2X_{i+1}$  for  $i < \ell$ . Instead of solving the recurrences, we show the following.

**Lemma 5.2.**  $X_i \geq \frac{5}{6}16^{\ell-i}$

*Proof.* We first observe that  $X_i + Y_i = 16(X_{i+1} + Y_{i+1})$ , as  $A_{i+1}$  consists of the  $4 \times 4$  smaller subarrays, each of them being  $A_i$  or  $A'_i$ . By unwinding the recurrence,  $X_i + Y_i = 16^{\ell-i}(X_\ell + Y_\ell) = 16^{\ell-i}$ . Furthermore, we argue that  $X_i \geq 5Y_i$  for every  $i < \ell$ . This is proved by induction on  $i$ :

$$i = \ell - 1 \quad X_{\ell-1} = 14X_{\ell} + 10Y_{\ell} = 14 \geq 5Y_{\ell-1} = 5(6Y_{\ell} + 2X_{\ell}) = 10.$$

$i < \ell - 1$  Assuming that  $X_{i+1} \geq 5Y_{i+1}$ , we write  $X_i = 14X_{i+1} + 10Y_{i+1} \geq 10X_{i+1} + 30Y_{i+1}$  and  $5Y_i = 30Y_{i+1} + 10X_{i+1}$ , so  $X_i \geq 5Y_i$ .

Therefore,  $16^{\ell-i} = X_i + Y_i \leq X_i + \frac{X_i}{5}$ , so  $X_i \geq \frac{5}{6}16^{\ell-i}$ .  $\square$

As explained earlier, whenever a copy of  $A_i$  occurs in  $A_{\ell}$ , all of its new runs contribute to the total number of runs in  $A_{\ell}$ . Therefore, the total number of runs in  $A_{\ell}$  is at least  $\sum_{i=1}^{\ell} X_i \cdot R_i$ .

**Theorem 1.3.** *There exists an infinite family of  $n \times n$  2D strings over the binary alphabet containing  $\Omega(n^2 \log n)$  runs.*

*Proof.* For each  $\ell \geq 2$ , we take  $n = 2 \cdot 4^{\ell}$  and construct the arrays  $A_1, A_2, \dots, A_{\ell}$  as described above. The final array  $A_{\ell}$  is over the binary alphabet by construction, consists of  $n$  rows and columns and contains at least  $\sum_{i=1}^{\ell} X_i \cdot R_i$  runs. By Lemma 5.1 and 5.2, this is at least

$$\begin{aligned} \sum_{i=1}^{\ell} \frac{5}{6} 16^{\ell-i} (2 \cdot 4^{i-1} - 1)^2 &= \frac{5}{6} 16^{\ell} \sum_{i=1}^{\ell} 16^{-i} \left( \frac{16^i}{4} - 4^i + 1 \right) \\ &= \frac{5}{6} 16^{\ell} \sum_{i=1}^{\ell} \left( \frac{1}{4} - \frac{1}{4^i} + \frac{1}{16^i} \right) \\ &= \frac{5}{6} 16^{\ell} \left( \frac{\ell}{4} + \frac{1}{3 \cdot 4^{\ell}} - \frac{1}{15 \cdot 16^{\ell}} - \frac{4}{15} \right) \\ &= \frac{5}{6 \cdot 4} \ell \cdot 16^{\ell} + \frac{5}{6 \cdot 3} 4^{\ell} - \frac{1}{6 \cdot 3} - \frac{5}{6 \cdot 3} 16^{\ell} \\ &\geq \frac{5}{24} \ell \cdot 16^{\ell} - \frac{1}{18} - \frac{2}{9} 16^{\ell} \\ &\geq \frac{1}{24} \ell \cdot 16^{\ell} = \Omega(n^2 \log n) \end{aligned} \quad \text{using } \ell \geq 2 \quad \square$$

## References

- [1] A. Amir and G. Benson. Efficient two-dimensional compressed matching. *In Data Compression Conference*, pages 279–288, 1992.
- [2] A. Amir and G. Benson. Two-dimensional periodicity in rectangular arrays. *SIAM Journal on Computing*, 27(1):90–106, 1998.
- [3] A. Amir, G. Benson, and M. Farach-Colton. An alphabet independent approach to two dimensional pattern matching. *SIAM Journal on Computing*, 23(2):313–323, 1995.
- [4] A. Amir, G. Benson, and M. Farach-Colton. Optimal parallel two dimensional text searching on a CREW PRAM. *Information and Computation*, 144:1–17, 1998.
- [5] A. Amir, G. M. Landau, S. Marcus, and D. Sokol. Two-dimensional maximal repetitions. *In 26th ESA*, 112(2):1–14, 2018.
- [6] A. Amir, G. M. Landau, S. Marcus, and D. Sokol. Two-dimensional maximal repetitions. *Theoretical Computer Science*, 812:49–61, 2020.

- [7] M. Amit and P. Gawrychowski. Distinct squares in circular words. *In 24th SPIRE*, pages 27–37, 2017.
- [8] A. Apostolico and V.E. Brimkov. Fibonacci arrays and their two-dimensional repetitions. *Theoretical Computer Science*, 237(1-2):263–273, 2000.
- [9] H. Bannai, T. I. S. Inenaga, Y. Nakashima, M. Takeda, and K. Tsuruta. The “runs” theorem. *SIAM Journal on Computing*, 46(5):1501–1514, 2017.
- [10] P. Charalampopoulos, J. Radoszewski, W. Rytter, T. Waleń, and W. Zuba. The Number of Repetitions in 2D-Strings. *In 28th ESA*, 173(32):1–18, 2020.
- [11] R. Cole, M. Crochemore, Z. Galil, L. Gasieniec, R. Eariharan, S. Muthukrishnan, K. Park, and W. Rytter. Optimally fast parallel algorithms for preprocessing and pattern matching in one and two dimensions. *In 34th FOCS*, pages 248–258, 1993.
- [12] M. Crochemore and L. Ilie. Maximal repetitions in strings. *Journal of Computer and System Sciences*, 74(5), 2008.
- [13] M. Crochemore, L. Ilie, and L. Tinta. The ”runs” conjecture. *Theoretical Computer Science*, 412(27):2931–2941, 2011.
- [14] M. Crochemore, C. S. Iliopoulos, T. Kociumaka, M. Kubica, J. Radoszewski, W. Rytter, W. Tyczynski, and T. Walen. The maximum number of squares in a tree. *In 23th CPM*, 7354:27–40, 2012.
- [15] M. Crochemore, C. S. Iliopoulos, M. Kubica, J. Radoszewski, W. Rytter, and T. Waleń. Extracting powers and periods in a word from its runs structure. *Theoretical Computer Science*, 521:29–41, 2014.
- [16] M. Crochemore and W. Rytter. Squares, cubes, and time-space efficient string searching. *Algorithmica*, 13:405–425, 1995.
- [17] J. D. Currie and D. S. Fitzpatrick. Circular words avoiding patterns. *Developments in Language Theory*, 2450:319–325, 2002.
- [18] A. Deza, F. Franek, and A. Thierry. How many double squares can a string contain? *Discrete Applied Mathematics*, 180:52–69, 2015.
- [19] X. Droubay, J. Justin, and G. Pirillo. Episturmian words and some constructions of de Luca and Rauzy. *Theoretical Computer Science*, 255(1):539–553, 2001.
- [20] J. Fischer, S. Holub, T. I. S. Inenaga, and M. Lewenstein. Beyond the runs theorem. *In 22th SPIRE*, pages 277–286, 2015.
- [21] A. S. Fraenkel and J. Simpson. How many squares can a string contain? *Journal of Combinatorial Theory, Series A*, 82(1):112–120, 1998.
- [22] F. Franek and Q. Yang. An asymptotic lower bound for the maximal number of runs in a string. *International Journal of Foundations of Computer Science*, 19(1):195–203, 2008.
- [23] P. Gawrychowski, T. Kociumaka, W. Rytter, and T. Walen. Tight bound for the number of distinct palindromes in a tree. *In 22th SPIRE*, 9309:270–276, 2015.

- [24] M. Giraud. Not so many runs in strings. *In 2nd LATA, volume 5196 of Lecture Notes in Computer Science*, pages 232–239, 2008.
- [25] L. Ilie. A simple proof that a word of length  $n$  has at most  $2n$  distinct squares. *Journal of Combinatorial Theory, Series A*, 112(1):163–164, 2005.
- [26] L. Ilie. A note on the number of squares in a word. *Theoretical Computer Science*, 380(3):373–376, 2007.
- [27] T. Kociumaka, J. Radoszewski, W. Rytter, and T. Walen. String powers in trees. *Algorithmica*, 79(3):814–834, 2017.
- [28] R. Kolpakov and G. Kucherov. Finding maximal repetitions in a word in linear time. *In 40th FOCS*, pages 596–604, IEEE Computer Society, 1999.
- [29] F. Manea and S. Seki. Square-density increasing mappings. *In 10th WORDS*, 9304:160–169, 2015.
- [30] W. Matsubara, K. Kusano, A. Ishino, H. Bannai, and A. Shinohara. New lower bounds for the maximum number of runs in a string. *In Proceedings of the Prague Stringology Conference 2008*, pages 140–145, 2008.
- [31] S. J. Puglisi, J. Simpson, and W. F. Smyth. How many runs can a string contain? *Theoretical Computer Science*, 401(1-3):165–171, 2008.
- [32] W. Rytter. The number of runs in a string. *Information and Computation*, 205(9):1459–1469, 2007.
- [33] J. Simpson. Modified padovan words and the maximum number of runs in a word. *The Australasian Journal of Combinatorics*, 46:129–146, 2010.
- [34] J. Simpson. Palindromes in circular words. *Theoretical Computer Science*, 550:66–78, 2014.
- [35] A. Thue. Über unendliche Zeichenreihen. *Norske Vid Selsk. Skr. I Mat-Nat Kl.(Christiana)*, 7:1–22, 1906.