Two-dimensional pattern matching with k mismatches

Jonas Ellert[1], Paweł Gawrychowski[2], Adam Górkiewicz[3], and Tatiana Starikovskaya[4]

[1]?
[2]?
[3]?
[4]?

Abstract

1 Related Work

J: work in progress ↓, unfinished

Hamming-Threshold PM in 1D

- **J:** ...

Exact PM in 2D Assuming $n \times n$ text and $m \times m$ pattern, $m \leq n$.

- Two-dimensional pattern matching dates back (at least) to the 1970s, when Richard Bird generalized the Knuth-Morris-Pratt algorithm [38] for one-dimensional pattern matching, achieving $\mathcal{O}(n^2 + m^2)$ time [17].
- By using multiple pattern matching on only $\mathcal{O}(n/m)$ rows of the text, Baeza-Yates and Régnier achieve average time complexity $\mathcal{O}(n^2/m)$, using $\mathcal{O}(m^2)$ space and still running in $\mathcal{O}(n^2)$ time in the worst case [14]. This was followed by more results optimizing the average time complexity [34, 48].
- Crochremore at al. [26] solve two-dimensional pattern matching space efficiently, using an $\mathcal{O}(m^2)$ time and $\mathcal{O}(\log m)$ space preprocessing followed by an $\mathcal{O}(n^2)$ time and constant space search phase.
- On a PRAM, the problem can be solved in constant time and $O(n^2)$ work [25], after preprocessing the pattern in $\mathcal{O}(\log \log m)$ time and $\mathcal{O}(m^2)$ work [23].
- Other works studying the problem are, e.g., [3, 15, 29, 37, 49, 50]

Hamming-Threshold PM in 2D Assuming $n \times n$ text and $m \times m$ pattern, $m \leq n$, and threshold $k < m^2$.

- First result in 1987 by Krithivasan and Sitalakshmi [39], who show that $\tilde{\mathcal{O}}(kmn^2)$ time can be achieved.
- Faster algorithm by Rank and Heywood runs in $\tilde{\mathcal{O}}((k+m)n^2)$ time [46].

- Current state of the art is $\mathcal{O}(kn^2)$ [6] **J:** check
- More work [13, 34, 44] is aimed at optimizing the average time complexity. This has also been done for the setting in which arbitrary rotations of the pattern are allowed [28].
- Dynamic and online algorithms [22]

2D Periodicity

- Two-dimensional periodicity was introduced in [1, 2] and refined in [29]. All periods (and witnesses for all non-periods) can be computed in $\mathcal{O}(n^2)$ time [24]. Witnesses can be used to efficiently eliminate potential occurrences of periodic patterns. Since witness computation is costly for run-length encoded strings, Amir at al. [4] show additional properties of two-dimensial periodicities that allow the elimination of occurrences without witnesses.
- Famous periodicity theorems have been generalized to two dimensions, for example the Fine-Wilf theorem [42] and the Lyndon-Schützenberger theorem [31].
- Smallest rectangular cover in $\mathcal{O}(n^2)$ time, all aperiodic covers in $\mathcal{O}(n^2 \log n)$ time [20]. Covers of two-dimensional strings were also considered in [27, 30].
- All tile covers in $\mathcal{O}(n^{2+\epsilon})$ time [45].
- There can be $\Omega(n^3 \log n)$ tandem substrings (repeated rectangle, like square in a normal string) [10].
- All tandems can be computed in $\mathcal{O}(n^3 \log n)$ time [11].
- There can be $\Omega(n^2 \log n)$ two-dimensional runs (maximal repetitions) [32].
- There are at most $\mathcal{O}(n^2 \log^2 n)$ two-dimensional runs [19].
- All runs can be computed in $\mathcal{O}(n^2 \log^2 n)$ time [7].

Other 2D Shannanigans

- A survey by Rytter [47] discusses compressed representations of two-dimensional strings and the computation on such representations without explicit decompression. If the text is compressed as a two-dimensional SLP, then merely deciding if a pattern is present is NP-complete with respect to the compressed size [16]. If both text and pattern are compressed using a two-dimensional version of Lempel-Ziv compression, then exact matching can be done in $\mathcal{O}(n^2 + m^3)$ time, using additional working space proportional to the compressed representation of the pattern [8]. If text and pattern are given in two-dimensional run-length encoding, then the time for exact matching is linear in the size of the compressed text, and the space is linear in the size of the compressed pattern [9]. If multiple patterns are given, then they can be compressed into a space efficient self-index that allows fast dictionary matching [43].
- More work on structural elements in two-dimensionsal strings, e.g., frames (rectangles with matching outer columns and rows) [18], Lyndon words [41], motif patterns (allowing "don't cares", and satisfying special properties) [12], patterns with at least two occurrences [36], and palindromic substrings [40].
- There is an efficient algorithm for matching a half-rectangular pattern (fixed height, but irregular shape) in a text, allowing a fixed number of mismatches, insertions and deletions [5].

J: work in progress, unfinished ↑

2 Introduction

We consider the one-dimensional all-substring Hamming distance problem (HD1D), where for a given text string T of length n and a string P of length m ($m < n$), we want to calculate the Hamming distance between P and every fragment T of length m.

We consider the two-dimensional all-substring Hamming distance problem (HD2D), where for a given 2D string T of size $n \times n$ and a string P of size $m \times m$ ($m < n$), we want to calculate the Hamming distance between P and every $m \times m$ fragment of T.

We also consider the bounded variants of HD1D and HD2D, where we are only required to calculate the distances which are not greater than k, for some parameter $k \in \mathbb{Z}^+$.

Theorem 1 (Main result)**.** *Bounded* HD2D *can be solved in* $\tilde{\mathcal{O}}((m^2 + mk^{5/4})n^2/m^2)$ *time.*

3 Preliminaries

For our purposes we will not use the standard definition of a two-dimensional string, where we associate it with a two-dimensional array of characters, and instead we will define it more broadly. Although we will occasionally use the array notation, we will do it exclusively for $n \times m$ strings. For any $n \in \mathbb{Z}^+$ we will denote $[n] = \{ 0, \ldots, n-1 \}$. We will only consider integer points or vectors and we will use these terms interchangeably. Our results hold under word-RAM model of computation.

Definition 1 (Two-dimensional string)**.** We define a **string** S as a partial function $\mathbb{Z}^2 \nrightarrow \Sigma$ which maps some arbitrary set of integer points, denoted as $\mathrm{dom}(S)$, to characters. For simplicity we will write $u \in S$ to denote that $u \in \mathrm{dom}(S)$. We say that a string S is **partitioned** into strings $R_1, \ldots, R_\ell$ when the sets $\mathrm{dom}(R_1), \ldots, \mathrm{dom}(R_\ell)$ partition $\mathrm{dom}(S)$ and $R_i(u) = S(u)$ for all $u \in R_i$. We call a string S **monochromatic** when $S(u) = \sigma$ for every $u \in S$ for some $\sigma \in \Sigma$ and we will write $\mathrm{C}(S)$ to denote the value σ. We say that S is $n \times m$ for some $n, m \in \mathbb{Z}^+$ when $\mathrm{dom}(S) = [n] \times [m]$. Physically we represent a string as a list of point-character pairs.

Definition 2 (Shifting)**.** For a set of points $V \subseteq \mathbb{Z}^2$ and a vector $u \in \mathbb{Z}^2$, we denote $V + u$ as $\{ v + u : v \in V \}$. For a string S and a vector $u \in \mathbb{Z}^2$ we denote $S + u$ as a string R such that $\mathrm{dom}(R) = \mathrm{dom}(S) + u$ and $R(v) = S(v - u)$ for $v \in \mathrm{dom}(R)$. Intuitively, we shift the set of points while maintaining their character values.

Definition 3 (Hamming distance)**.** For a pair of strings S, R we define

$$\mathrm{Ham}(S, R) = |\{ u : u \in \mathrm{dom}(S) \cap \mathrm{dom}(R), S(u) \neq R(u) \}|,$$

which corresponds to the number of mismatches between S and R.

Under such notation, the HD2D problem is equivalent to calculating the (bounded or unbounded) values of $\mathrm{Ham}(P + q, T)$ for all $q \in \mathbb{Z}^2$ such that $\mathrm{dom}(P + q) \subseteq \mathrm{dom}(T)$ (so for $q \in [n - m + 1]^2$).

Definition 4 (Don't care symbol)**.** We define the **don't care** symbol as a special character which matches with every character. We will denote it with `?`. Unless stated otherwise, we assume it is not allowed in Σ and in both HD1D and HD2D every character present in T and P matches only with itself.

Definition 5 (Vector operators)**.** For any $u \in \mathbb{Z}^2$ we refer to its coordinates as $u.x, u.y$. For $u, v \in \mathbb{Z}^2$ we denote $u \cdot v = u.x \cdot v.x + u.y \cdot v.y$ and $u \times v = u.x \cdot v.y - u.y \cdot v.x$. Note that alternatively $u \cdot v = |u||v| \cos \alpha$ and $u \times v = |u||v| \sin \alpha$ where α is the angle between u and v.

Definition 6 (Quadrants)**.** We define the four **quadrants** as

$$\mathcal{Q}_1 = (0, +\infty) \times [0, +\infty),$$
$$\mathcal{Q}_2 = (-\infty, 0] \times (0, +\infty),$$
$$\mathcal{Q}_3 = (-\infty, 0) \times (-\infty, 0],$$
$$\mathcal{Q}_4 = [0, +\infty) \times (-\infty, 0).$$

4 One-dimensional generalizations

In this section we explore some of the methods used for one-dimensional strings. Specifically, as our goal is to generalize the solution for pattern matching with k mismatches described in [33], we are especially interested in two-dimensional variants of the techniques that were used to solve the one-dimensional case.

Theorem 2 (Instancing)**.** *Consider an algorithm $\mathcal{A}$ which solves* HD2D *(bounded or unbounded), but only when $2|n$ and $n \leq \frac{3}{2}m$. If its running time is $\mathcal{T}(m)$, then the general case can be solved in $\mathcal{O}(\mathcal{T}(m)n^2/m^2)$.*

Proof. Let $r = \lfloor m/2 \rfloor$ and let $n' = r + m - 1$ or $r + m$ if $r + m - 1$ is odd. We see that the set $N = [n']^2$ satisfies the conditions for the text domain. For any vector $q \in [n - m]^2$ we can find a vector u such that $r|u.x, r|u.y$ and $q - u \in [r]^2$, so we have $\mathrm{Ham}(P + q, T) = \mathrm{Ham}(P + q - u, T_u)$ where T_u is the restriction of $T - u$ to N. If $T - u$ is not defined for some $v \in N$, we can pad $T_u(v)$ with any character. We see that $\mathrm{dom}(P + q - u) \subseteq N = \mathrm{dom}(T_u)$. There are $\mathcal{O}(n^2/m^2)$ possible vectors u and we run $\mathcal{A}$ for every pair of T_u and P. □

Theorem 3 (Kangaroo jumps)**.** *Consider an $n \times n$ string T, $m \times m$ string P and set of vectors Q such that $\mathrm{dom}(P + q) \subseteq \mathrm{dom}(T)$ for every $q \in Q$. There exists an algorithm which calculates $d_q = \mathrm{Ham}(P + q, T)$ for every $q \in Q$ in total time $\tilde{\mathcal{O}}(n^2 + \sum_{q \in Q} d_q)$.*

Proof. For the sake of clarity, we will temporarily switch to the classical array notation for strings. Let $T_0, \ldots, T_{n-m}$ denote an array of two-dimensional strings (arrays) such that $T_k[0\,..\,n - 1, 0\,..\,m - 1] = T[0\,..\,n - 1, k\,..\,k + m - 1]$. For every row $P[0], \ldots, P[m - 1]$ of P and every row $T_k[0], \ldots, T_k[n - 1]$ of every T_k we assign an integer identifier so that $\mathrm{Id}(P[i]) = \mathrm{Id}(T_k[j]) \Leftrightarrow P[i] = T_k[j]$ by using the KMR algorithm ([reference]) in $\tilde{\mathcal{O}}(n^2)$.

We use the approach described in [kangaroo reference]. There exists a data structure (suffix array) which for a given one-dimensional array S allows us to detect all mismatches between any given two of its subarrays of equal length. It can be built in $\tilde{\mathcal{O}}(|S|)$ and the query time is $\tilde{\mathcal{O}}(d+1)$ where d is the number of mismatches. We construct the suffix array for the concatenation of the following arrays:

- the rows $P[i]$ for every i,
- the rows $T[i]$ for every i,
- the array $\mathrm{Id}(P[0])\,\mathrm{Id}(P[1])\ldots\mathrm{Id}(P[m - 1])$,
- the arrays $\mathrm{Id}(T_k[0])\,\mathrm{Id}(T_k[1])\ldots\mathrm{Id}(T_k[n - 1])$ for every k,

the total length of which is $\mathcal{O}(n^2)$. Let us consider a problem of detecting mismatches between P and some $T' = T[j\,..\,j + m - 1, k\,..\,k + m - 1]$. We can first find all row indices i for which $P[i] \neq T'[i]$ by finding all mismatches between $\mathrm{Id}(P[0])\ldots\mathrm{Id}(P[m-1])$ and $\mathrm{Id}(T_k[j])\ldots\mathrm{Id}(T_k[j+m-1])$, which we do with query to the data structure. For every such i we can then find all mismatches between $P[i]$ and $T'[i]$ by querying $P[i]$ and $T[i + j][k\,..\,k + m - 1]$. If the distance between P and T' is d, the first query takes $\tilde{\mathcal{O}}(d + 1)$ operations and all subsequent queries take $\tilde{\mathcal{O}}(d + 1)$ operations in total. □

Lemma 1. HD1D *with don't care symbols can be solved in* $\tilde{\mathcal{O}}(n|\Sigma|)$ *by running* $|\Sigma|$ *instances of FFT.*

Lemma 2. *There exists a* $(1+\varepsilon)$*-approximate algorithm (introduced in [35]) which solves* HD1D *with don't care symbols in* $\tilde{\mathcal{O}}(n)$*.*

Theorem 4. HD2D *with don't care symbols can be solved in* $\tilde{\mathcal{O}}(n^2|\Sigma|)$*.*

Proof. We will again use the array notation. We construct one-dimensional strings $\bar{T}$ and $\bar{P}$ by concatenating subsequent rows $T[0], \ldots, T[n-1]$ of T and rows $P[0], \ldots, P[m-1]$ of P padded with don't care symbols:

$$\bar{T} = T[0]\ T[1]\ \ldots\ T[n-1],$$
$$\bar{P} = P[0]\ ?^{n-m}\ P[1]\ ?^{n-m}\ \ldots\ ?^{n-m}\ P[m-1].$$

We run the algorithm from Lemma 1. The distance between $T[i\,..\,i+m-1, j\,..\,j+m-1]$ and P is equal to the distance between $\bar{T}[in+j\,..\,in+j+nm-n+m-1]$ and $\bar{P}$. □

Theorem 5. *There exists a* $(1+\varepsilon)$*-approximate algorithm which solves* HD2D *with don't care symbols in* $\tilde{\mathcal{O}}(n^2)$*.*

Proof. Identical to Theorem 4, but we use the algorithm from Lemma 2 instead of Lemma 1. □

The same reduction as in Theorem 4 can be applied for every HD1D solution which allows don't care symbols. Unfortunately, the most effective known algorithms for bounded HD1D rely on periodicity ([21], [33]) and inherently do not allow don't care symbols, thus, they cannot be easily generalized.

Observation 1 (Don't care padding)**.** *Every* HD2D *solution which allows don't care symbols (eg. the algorithms from Theorem 4 and Theorem 5) can be extended to also calculate the Hamming distance for occurrences of* P *which are not entirely contained in* T*. It can be done by padding the text with don't care symbols and it does not change the complexity of the solution.*

5 Proof of Theorem 1

We show an algorithm which works in time $\tilde{\mathcal{O}}(m^2 + mk^{5/4})$, assuming $2|n$ and $m < n \leq \frac{3}{2}m$. By Theorem 2, our main result follows.

We start by running the algorithm from Theorem 5 with $\varepsilon = 1$. We construct the set Q as the set of such vectors $q \in \mathbb{Z}^2$ for which the estimated value of $\mathrm{Ham}(P+q, T)$ is at most $2k$. For every $q \in \{0, \ldots, n-m\}^2 \setminus Q$ we say that $\mathrm{Ham}(P+q, T)$ equals ∞. The next step is to calculate the exact value of $\mathrm{Ham}(P+q, T)$ for every $q \in Q$.

Let us consider the case when $|Q| \leq 2m + m^2/k$. We can run the algorithm from Theorem 3 and by the fact that $\mathrm{Ham}(P+q, T) \leq 4k$ for every $q \in Q$, it will perform $\tilde{\mathcal{O}}(m^2 + mk)$ operations. We are left with the case when $|Q| > 2m + m^2/k$, in which we take advantage of the fact that some strings $P+q$ for $q \in Q$ must have a large overlap and small Hamming distance from each other, and thus P must be periodic.

5.1 Two-dimensional periodicity

In this section we introduce a range of new tools related to two-dimensional periodicity. We then select some special periods of the pattern and show how to decompose it into some regularly structured monochromatic strings.

Definition 7 (Periodicity)**.** Consider any vector $\delta \in \mathbb{Z}^2$. We say that a string S has an ℓ-period δ when

$$\mathrm{Ham}(S+\delta, S) \leq \ell.$$

Lemma 3. *For every $u, v \in Q$, the vector $u - v$ is an $8k$-period of P.*

Proof. $\mathrm{Ham}(P+u-v, P) = \mathrm{Ham}(P+u, P+v) \leq \mathrm{Ham}(P+u, T) + \mathrm{Ham}(P+v, T) \leq 4k+4k$. □

Theorem 6. *For a given $\ell \in \mathbb{Z}^+$ and a set of points $U \subseteq [\ell+1]^2$ such that $|U| > 4\ell$ there exist $s, t, s', t' \in U$ such that the following conditions hold for $w = t - s$ and $w' = t' - s'$:*

- $0 < |w||w'| \leq 22\frac{\ell^2}{|U|}$,
- $|\sin\alpha| \geq \frac{1}{2}$ *where α is the angle between w and w',*
- $w, w', -w, -w'$ *are all contained in different quadrants.*

Such w, w' can be found in $\tilde{\mathcal{O}}(|U|)$ operations.

Proof. See Section 5.4. □

We run the algorithm from Theorem 6 on the set Q (where $\ell = n - m \leq m/2$, thus $|Q| > 2m + m^2/k \geq 4\ell$). We obtain vectors $\varphi \in \mathcal{Q}_4$ and $\psi \in \mathcal{Q}_1$ which by Lemma 3 are $\mathcal{O}(k)$-periods of P. We will refer to those vectors throughout the rest of the description and we define $p = \varphi \times \psi$. Note that because $|Q| > 2m + m^2/k$, we have $p \leq |\varphi||\psi| = \mathcal{O}(\min\{m, k\})$.

Definition 8 (Lattice congruency)**.** We define $\mathcal{L} = \{ s\varphi + t\psi : s, t \in \mathbb{Z} \}$. We say that two vectors $u, v \in \mathbb{Z}^2$ are **lattice-congruent** and denote $u \equiv v$ when $u - v \in \mathcal{L}$ [Galil citation].

Lemma 4. *There exists a set of points $\Gamma \subseteq \mathbb{Z}^2$ such that $|\Gamma| = p$ and every point $u \in \mathbb{Z}^2$ is lattice-congruent to exactly one point $\gamma \in \Gamma$.*

Proof. TODO □

Definition 9 (Parquet)**.** We call a non-empty set $U \subseteq \mathbb{Z}^2$ a **parquet** when there exist some values $x_0, x_1, y_0, y_1, \varphi_0, \varphi_1, \psi_0, \psi_1 \in \mathbb{Z}$, which we will call its **signature**, such that

$$U = [x_0, x_1) \times [y_0, y_1) \cap \{ u : u \in \mathbb{Z}^2, \varphi \times u \in [\varphi_0, \varphi_1), \psi \times u \in [\psi_0, \psi_1) \}.$$

a) If additionally $x_1 - x_0 \geq |\varphi.x| + |\psi.x|$ and $y_1 - y_0 \geq |\varphi.y| + |\psi.y|$, then U is a **spacious** parquet.

b) If additionally $x_0, y_0 = -\infty$ and $x_1, y_1 = +\infty$, then U is a **simple** parquet.

Note that every simple parquet is spacious.

Definition 10 (Subparquet)**.** We call a non-empty set $V \subseteq \mathbb{Z}^2$ a **subparquet** when there exists a parquet U and a point $\gamma \in \mathbb{Z}^2$ such that

$$V = \{ u : u \in U, u \equiv \gamma \}.$$

We call V a spacious/simple subparquet when there exists U which is (correspondingly) a spacious/simple parquet. We say that V is lattice-congruent to some $v \in \mathbb{Z}^2$ (denoted as $V \equiv v$) when $v \equiv \gamma$. We similarly define the lattice congruency between two subparquets.

Theorem 7 (Subparquet convolution)**.** *For a given list of signatures of simple subparquets $U_0, \ldots, U_{\ell-1}$, list of signatures of subparquets $V_0, \ldots, V_{\ell-1}$ and a set of vectors Q we can calculate*

$$\sum_{i=0}^{\ell-1} |(U_i + q) \cap V_i|$$

for every $q \in Q$ in total time $\tilde{\mathcal{O}}(m^2 + \ell + |Q|)$, assuming that the subparquets only contain vectors of length $\mathcal{O}(m)$.

Proof. See Section 5.5. □

Definition 11 (Parquet string)**.** We call a string S a spacious/simple (sub-)parquet string when $\operatorname{dom}(S)$ is a spacious/simple (sub-)parquet.

Theorem 8 (Periodic string decomposition)**.** *A given spacious/simple parquet string R with $\mathcal{O}(k)$-periods φ and ψ can be partitioned in time $\tilde{\mathcal{O}}(|\operatorname{dom}(R)|)$ into $\mathcal{O}(k)$ monochromatic spacious/simple subparquet strings, correspondingly.*

Proof. See Section 5.6. □

Since $|\varphi.x|, |\varphi.y|, |\psi.x|, |\psi.y| \leq n - m \leq m/2$, the $m \times m$ string P is a spacious parquet string and satisfies the assumptions of Theorem 8. We partition P into a set of strings $\mathcal{V}$. We then group the strings based on the single character they contain. Specifically, we construct $\mathcal{V}_\sigma = \{\, V : V \in \mathcal{V}, \mathrm{C}(V) = \sigma \,\}$ for every character $\sigma \in \Sigma$ present in P.

Theorem 9. *For a given set of monochromatic simple subparquet strings $\mathcal{S}$ we can calculate*

$$\sum_{S \in \mathcal{S}} \operatorname{Ham}(P + q, S)$$

for every $q \in Q$ in total time $\tilde{\mathcal{O}}(m^2 + \sum_{S \in \mathcal{S}} |\mathcal{V}_{\mathrm{C}(S)}|)$, assuming that the sets $\operatorname{dom}(S)$ for $S \in \mathcal{S}$ are some pairwise disjoint subsets of $\operatorname{dom}(T)$.

Proof. Let $U = \bigcup_{S \in \mathcal{S}} \operatorname{dom}(S)$. Observe that

$$\sum_{S \in \mathcal{S}} \operatorname{Ham}(P + q, S) = |(P + q) \cap U| - \sum_{S \in \mathcal{S}} \sum_{V \in \mathcal{V}_{\mathrm{C}(S)}} |\operatorname{dom}(V + q) \cap \operatorname{dom}(S)|.$$

We can calculate $|(P + q) \cap U|$ for every $q \in Q$ with a single instance of FFT (see Theorem 4) or by using prefix sums in time $\tilde{\mathcal{O}}(m^2)$. To calculate the values

$$\sum_{S \in \mathcal{S}} \sum_{V \in \mathcal{V}_{\mathrm{C}(S)}} |\operatorname{dom}(V + q) \cap \operatorname{dom}(S)|$$

we use the algorithm from Theorem 7 (where $\ell = \sum_{S \in \mathcal{S}} |\mathcal{V}_{\mathrm{C}(S)}|$). □

5.2 Text decomposition

Because the text is not necessarily periodic, we unfortunately cannot use the same approach as for the pattern. In this section we show how to decompose T using a similar, but more nuanced approach.

Definition 12 (Active text)**.** We define the **active text** $T_{\mathbf{a}}$ as the restriction of T to $\bigcup_{q \in Q} \operatorname{dom}(P) + q$. For a point $u \in \mathbb{Z}^2$ we define its **border distance** as $\min\{\, |u - v| : v \in \mathbb{Z}^2 \setminus \operatorname{dom}(T_{\mathbf{a}}) \,\}$, which we will denote as $\operatorname{BorderDis}(u)$. For a set of points $U \subseteq \mathbb{Z}^2$, we define $\operatorname{BorderDis}(U) = \max\{\, \operatorname{BorderDis}(u) : u \in U \,\}$ and for a string S, we define $\operatorname{BorderDis}(S) = \operatorname{BorderDis}(\operatorname{dom}(S))$. Note that we consider the maximum distance, not minimum.

Observation 2. $\operatorname{Ham}(P + q, T) = \operatorname{Ham}(P + q, T_{\mathbf{a}})$ *for every $q \in Q$.*

Theorem 10 (Active text decomposition)**.** *For a given positive integer $\ell \leq m$, we can partition the active text in time $\tilde{\mathcal{O}}(m^2 + \ell k)$ into a set of $\mathcal{O}(\ell k)$ monochromatic simple subparquet strings and a string F such that $\operatorname{BorderDis}(F) = \mathcal{O}(m/\ell)$.*

Proof. See Section 5.7. □

We partition $T_{\mathbf{a}}$ using the algorithm from Theorem 10 with $\ell = mk^{-3/4}$ into a set of simple subparquet strings $\mathcal{S}$ and a string F, where $|\mathcal{S}| = \mathcal{O}(mk^{1/4})$ and $\mathrm{BorderDis}(F) = \mathcal{O}(k^{3/4})$. For every $q \in Q$ we then have

$$\mathrm{Ham}(P+q, T_{\mathbf{a}}) = \mathrm{Ham}(P+q, F) + \sum_{S\in\mathcal{S}} \mathrm{Ham}(P+q, S).$$

By Theorem 9, we can calculate $\sum_{S\in\mathcal{S}} \mathrm{Ham}(P+q,S)$ for every $q \in Q$ in time $\tilde{\mathcal{O}}(m^2+mk^{5/4})$, since $\sum_{S\in\mathcal{S}} |\mathcal{V}_{C(S)}| \le |\mathcal{S}||\mathcal{V}| = \mathcal{O}(mk^{5/4})$. In the next section we will introduce Theorem 12, which states that we can calculate $\mathrm{Ham}(P+q,F)$ for every $q \in Q$ in total time $\tilde{\mathcal{O}}(m^2 + mk^{1/2}\,\mathrm{BorderDis}(F))$. By substituting $\mathrm{BorderDis}(F) = \mathcal{O}(k^{3/4})$, we get the complexity of $\tilde{\mathcal{O}}(m^2+mk^{5/4})$, which ends the main proof.

5.3 Border proximity

In this section we explore the properties of strings defined only for the points close to the border. We consider any non-empty string S, such that $\mathrm{dom}(S) \subseteq \mathrm{dom}(T_{\mathbf{a}})$ and let $d = \mathrm{BorderDis}(S)$. We define a partitioning of S into strings $S_1, \ldots, S_4$, by splitting it through the middle with a horizontal and vertical line. Specifically

- S_1 is the restriction of S to $\{n/2, \ldots, n-1\} \times \{n/2, \ldots, n-1\}$ (upper right quarter),
- S_2 is the restriction of S to $\{0, \ldots, n/2-1\} \times \{n/2, \ldots, n-1\}$ (upper left quarter),
- S_3 is the restriction of S to $\{0, \ldots, n/2-1\} \times \{0, \ldots, n/2-1\}$ (lower left quarter),
- S_4 is the restriction of S to $\{n/2, \ldots, n-1\} \times \{0, \ldots, n/2-1\}$ (lower right quarter).

We will demonstrate some characteristics of S_1, and by symmetry, generalize them to S.

Lemma 5. *Assuming $d \le m/4$, there does not exist $u \in S_1$ and $v \in T_a$ such that $v.x - u.x \ge d$ and $v.y - u.y \ge d$.*

Proof. Assume the contrary. Since $u \in S_1$, we have $\mathrm{BorderDis}(u) \le d$, so there exists $w \in \mathbb{Z}^2 \setminus \mathrm{dom}(T_{\mathbf{a}})$ such that $u.x - d \le w.x \le u.x + d$ and $u.y - d \le w.y \le u.y + d$. Since $v \in T_a$, there exists $q \in Q$ such that $v \in [m]^2 + q$. We have

$$w.x \ge u.x - d \ge n/2 - m/4 \ge n - m > q.x$$

and

$$w.x \le u.x + d \le v.x \le q.x + m - 1.$$

Similarly we can show that $q.y \le w.y \le q.y + m - 1$, and thus $w \in [m]^2 + q$. Since $[m]^2 + q \subseteq \mathrm{dom}(T_{\mathbf{a}})$ and $w \notin T_{\mathbf{a}}$, we get a contradiction. □

Theorem 11. *We can calculate $\mathrm{Ham}(P+q,S)$ for every $q \in Q$ in total time $\tilde{\mathcal{O}}(m^2 + md|\Sigma|)$, where $|\Sigma|$ is the number of different characters present in both P and S.*

Proof. See Section 5.3.1. □

Theorem 12. *We can calculate $\mathrm{Ham}(P+q,S)$ for every $q \in Q$ in total time $\tilde{\mathcal{O}}(m^2 + mdk^{1/2})$.*

Proof. See Section 5.3.2. □

5.3.1 Proof of Theorem 11

We base our approach on the simple method of calculating the Hamming distance by running an instance of FFT for each unique character. We will again utilize partitioning to reduce the problem to some smaller ones and then solve them naively.

Definition 13. (width & height) For a non-empty set $U \subseteq \mathbb{Z}^2$ we define its **width** as $\max\{u.x - v.x + 1 : u, v \in U\}$ and its **height** as $\max\{u.y - v.y + 1 : u, v \in U\}$. For a non-empty string R we define the width and height as the width and height of $\mathrm{dom}(R)$.

Theorem 13. *Given two non-empty strings P and T of widths w_P, w_T and heights h_P, h_T, we can calculate $\mathrm{Ham}(P + q, T)$ for every $q \in \mathbb{Z}^2$, for which the result is non-zero, in total time $\tilde{\mathcal{O}}((|\Sigma| + 1)(w_P + w_T)(h_P + h_T))$, where $|\Sigma|$ denotes the number of different characters present in both P and T.*

Proof. We can prove it by slightly generalizing Theorem 4, although following the same method, and utilizing Observation 1. □

We will assume that $d \leq m/4$, since for $d = \Omega(m)$ we can, by Theorem 13, calculate the results in time $\tilde{\mathcal{O}}(m^2 + m^2|\Sigma|)$, which is sufficient.

Recall that $\mathrm{Ham}(P+q, S) = \mathrm{Ham}(P+q, S_1) + \cdots + \mathrm{Ham}(P+q, S_4)$. We will only show how to calculate $\mathrm{Ham}(P + q, S_1)$ for every $q \in Q$, since the other cases are symmetric.

We will take advantage of the fact that the points close to the border can overlap only with a small subset of points from the pattern when considering the occurrences fully contained in the active text. Specifically, let us consider a string P_0, defined as the restriction of P to $[m - d]^2$ and a string P_1, defined as the restriction of P to $\mathrm{dom}(P) \setminus \mathrm{dom}(P_0)$. Since the strings P_0 and P_1 partition P, we have

$$\mathrm{Ham}(P + q, S_1) = \mathrm{Ham}(P_0 + q, S_1) + \mathrm{Ham}(P_1 + q, S_1).$$

Lemma 6. $\mathrm{dom}(P_0 + q) \cap \mathrm{dom}(S_1) = \emptyset$ *for every $q \in Q$.*

Proof. Let us assume the contrary. Select any $q \in Q$ such that $\mathrm{dom}(P_0 + q) \cap \mathrm{dom}(S_1)$ contains some point u and consider the point $v = (u.x + d, u.y + d)$. Since $u \in [m - d]^2 + q$, we have $v \in [m]^2 + q \subseteq \mathrm{dom}(T_{\mathbf{a}})$, thus the points $u \in S_1$ and $v \in T_{\mathbf{a}}$ contradict Lemma 5. □

Observation 3. *P_1 can be partitioned into two strings P_2 and P_3 such that the width of P_2 and the height of P_3 are equal to d.*

By Lemma 6, $\mathrm{Ham}(P_0 + q, S_1) = 0$ for every $q \in Q$ and by Observation 3 we then have

$$\mathrm{Ham}(P + q, S_1) = \mathrm{Ham}(P_1 + q, S_1) = \mathrm{Ham}(P_2 + q, S_1) + \mathrm{Ham}(P_3 + q, S_1).$$

for some newly constructed strings P_2 and P_3, such that the width of P_2 and the height of P_3 equals d. We calculate $\mathrm{Ham}(P_2 + q, S_1)$ and $\mathrm{Ham}(P_3 + q, S_1)$ for every Q independently and sum the results. We only show how to calculate $\mathrm{Ham}(P_2 + q, S_1)$, since the other case is symmetric.

We will now partition S_1. Consider an array of strings $U_0, \ldots, U_{\lceil n/d \rceil - 1}$, where U_i is the restriction of S_1 to $\{id, \ldots, id + d - 1\} \times [n] \cap \mathrm{dom}(S_1)$. For the sake of formality (since the maximum/minimum of an empty set is undefined), let $V_0, \ldots, V_{\ell-1}$ consist of all non-empty strings U_i, given in the increasing order of i. Observe that $V_0, \ldots, V_{\ell-1}$ partition S_1 and their width is not greater than d.

For each $i \in [\ell]$ we find $h_i \in \mathbb{Z}^+$, which we define as the minimal number such that $(u.x, u.y + h_i) \notin T_{\mathbf{a}}$ for every $u \in V_i$.

Lemma 7. *The sum of all h_i is $\mathcal{O}(m)$.*

Proof. Since for $\ell < 2$, the proof is trivial, we assume $\ell \geq 2$. For every i (since h_i is minimal) there exists a point $u_i \in V_i$, such that $(u_i.x, u_i.y + h_i - 1) \in T_{\mathbf{a}}$. It can be shown that for all $i \geq 2$ we have

$$h_i \leq u_{i-2}.y - u_i.y + d,$$

since if that was not the case for some i, then the points u_{i-2} and $v = (u_i.x, u_i.y + h_i - 1)$ would contradict Lemma 5. We can conclude that

$$\sum_{i=0}^{\ell-1} h_i \leq h_0 + h_1 + \sum_{i=2}^{\ell-1} (u_{i-2}.y - u_i.y + d) = h_0 + h_1 + u_0.y + u_1.y - u_{\ell-2}.y - u_{\ell-1}.y + (\ell-2)d = \mathcal{O}(m).$$

□

Observation 4. *For every i, the height of V_i is not greater than h_i. By Lemma 7 we have $|S_1| = \mathcal{O}(md)$ and by extension $|S| = \mathcal{O}(md)$.*

For every $i \in [l]$ we construct the string L_i as the restriction of P_2 to $[m] \times [m - h_i] \cap \operatorname{dom}(P_2)$ and the string H_i as the restriction of P_2 to $\operatorname{dom}(P_2) \setminus \operatorname{dom}(L_i)$. Since L_i and H_i partition V_i, we have

$$\operatorname{Ham}(P_2 + q, S_1) = \sum_{i=0}^{\ell-1} \operatorname{Ham}(P_2 + q, V_i) = \sum_{i=0}^{\ell-1} \operatorname{Ham}(L_i + q, V_i) + \sum_{i=0}^{\ell-1} \operatorname{Ham}(H_i + q, V_i).$$

Lemma 8. $\operatorname{dom}(L_i + q) \cap \operatorname{dom}(V_i) = \emptyset$ *for every $q \in Q$ and $i \in [\ell]$.*

Proof. Let us assume the contrary. Select any $q \in Q$ and $i \in [\ell]$, such that $\operatorname{dom}(L_i + q) \cap \operatorname{dom}(V_i)$ contains some point u and consider the point $v = (u.x, u.y + h_i)$. Since $u \in [m] \times [m - h_i] + q$, we have $v \in [m]^2 + q \subseteq \operatorname{dom}(T_{\mathbf{a}})$, thus $v \in T_{\mathbf{a}}$, which contradicts the definition of h_i. □

By Lemma 8, for every $q \in Q$ we have $\sum_{i=0}^{\ell-1} \operatorname{Ham}(L_i + q, V_i) = 0$, thus our result is equal to $\sum_{i=0}^{\ell-1} \operatorname{Ham}(H_i + q, V_i)$. We run the algorithm from Theorem 13 for every pair of H_i and V_i and, since both H_i and V_i have widths not greater than d and heights not greater than h_i, we obtain the total complexity of $\tilde{\mathcal{O}}(\sum_{i=0}^{\ell-1}(|\Sigma| + 1)dh_i)$, which, by Lemma 7, is $\tilde{\mathcal{O}}(m^2 + md|\Sigma|)$.

5.3.2 Proof of Theorem 12

Recall the construction of the sets $\mathcal{V}_\sigma$ described in Section 5.1. We define $\sigma \in \Sigma$ to be a **frequent** character if $|\mathcal{V}_\sigma| \geq \sqrt{k}$ and if $|\mathcal{V}_\sigma| < \sqrt{k}$, we call it an **infrequent** character.

Observation 5. *The number of different frequent characters is $\mathcal{O}(\sqrt{k})$.*

We partition S into two strings F and I, based on character frequency, so that F consists of only the frequent characters and I consists of only the infrequent ones. For every $q \in Q$ we then have

$$\operatorname{Ham}(P + q, S) = \operatorname{Ham}(P + q, F) + \operatorname{Ham}(P + q, I).$$

By Observation 5 and Theorem 11, we can calculate $\operatorname{Ham}(P + q, F)$ for every $q \in Q$ in total time $\tilde{\mathcal{O}}(m^2 + mdk^{1/2})$, since $\operatorname{BorderDis}(F) \leq d$.

We partition I into $|\operatorname{dom}(I)|$ strings, one per every $u \in I$. Specifically, let I_u be the restriction of I to $\{u\}$ for every $u \in I$. We have $\operatorname{Ham}(P + q, I) = \sum_{u \in I} \operatorname{Ham}(P + q, I_u)$ for every $q \in Q$. By Definition 10, I_u are simple subarquet strings, and thus, we can by Theorem 9 calculate the results in $\tilde{\mathcal{O}}(m^2 + \sum_{u \in I} |\mathcal{V}_{I(u)}|)$. Since $I(u)$ is an infrequent character for every $u \in I$, we have $|\mathcal{V}_{I(u)}| < k^{1/2}$ for every $u \in I$. By Observation 4 we have $|\operatorname{dom}(I)| = \mathcal{O}(md)$, and thus the total complexity is $\tilde{\mathcal{O}}(m^2 + mdk^{1/2})$.

5.4 Proof of Theorem 6

First, we find any closest pair of vectors $s, t \in U$ by running the standard $\tilde{\mathcal{O}}(|U|)$ time algorithm and denote $w = t - s$. We define a partial order $\leq_w$ where $v \leq_w u$ for some $u, v \in U$ when at least one condition holds:

(a) $u = v$,

(b) $u - v$ and w belong to the same quadrant,

(c) $\alpha \in (-\frac{\pi}{6}, \frac{\pi}{6})$ where α is the angle between w and $u - v$.

We find the longest chain C and the longest antichain A using dynamic programming in $\tilde{\mathcal{O}}(|U|)$ operations. We then find any closest pair of vectors $s', t' \in A$ and denote $w' = t' - s'$. We have the following inequalities:

(i) $|U| \leq |C||A|$ (by Dilworth's theorem),

(ii) $(|C| - 1)|w| \leq (1 + \sqrt{3})\ell$ (roughly by the fact that vectors in C must be increasing in a certain direction),

(iii) $(|A| - 1)|w'| \leq 2\ell$ (by using a similar argument for vectors in A).

By considering the assumption $|U| > 4\ell$ it can be proven that $|w||w'| \leq 22\frac{\ell^2}{|U|}$ and the other conditions also hold.

5.5 Proof of Theorem 7

Throughout this section we will denote $D = \{\, u : u \in \mathcal{L}, \varphi \times u \geq 0, \psi \times u \geq 0 \,\}$, where $\mathcal{L}$ is the set introduced in Definition 8.

Lemma 9. *Given a set of subparquets $\mathcal{V}$ and a set of points Q, we can calculate*

$$\sum_{V \in \mathcal{V}} |(D + q) \cap V|$$

for every $q \in Q$ in total time $\tilde{\mathcal{O}}(n^2 + |Q| + |\mathcal{V}|)$, assuming that every $V \in \mathcal{V}$ consists of vectors of length $\mathcal{O}(n)$.

Proof. For every $u \in \mathbb{Z}^2$ let us define $\text{score}(u) = |\,\{\, V : V \in \mathcal{V}, u \in V \,\}\,|$. Observe that

$$\sum_{V \in \mathcal{V}} |(D + q) \cap V| = \sum_{u \in D + q} \text{score}(u).$$

We start by explicitly calculating the scores. We find the maximum length of a vector that some $V \in \mathcal{V}$ is defined for, which we denote ℓ. We construct the set $U \subseteq \mathbb{Z}^2$ of all vectors of length at most ℓ. By the assumption, we have $\ell = \mathcal{O}(n)$, and thus $|U| = \mathcal{O}(l^2) = \mathcal{O}(n^2)$. We observe that since all the scores are zero for points outside of U, we can only calculate them for $u \in U$.

We find the set Γ introduced in Lemma 4 and for every $\gamma \in \Gamma$ we construct $U_\gamma = U \cap (\mathcal{L} + \gamma)$. Consider any $u \in U_\gamma$ for some fixed $\gamma \in \Gamma$ and any $V \in \mathcal{V}$. We observe that if $V \not\equiv \gamma$, then $u \notin V$ and thus V does not contribute to $\text{score}(u)$. If $V \equiv \gamma$, then we can find a parquet W such that $V = W \cap (\mathcal{L} + \gamma)$ and we have $u \in V \Leftrightarrow u \in W \cap (\mathcal{L} + \gamma) \Leftrightarrow u \in W$. Thus, if we denote $\mathcal{W}_\gamma$ as the set of parquets W obtained for every $V \in \mathcal{V}$ such that $V \equiv \gamma$, then $\text{score}(u)$ for $u \in U_\gamma$ is the number of parquets $W \in \mathcal{W}_\gamma$ such that $u \in W$. We calculate $\text{score}(u)$ for every $u \in U_\gamma$ by sweeping U_γ and $\mathcal{W}_\gamma$ in time $\tilde{\mathcal{O}}(|U_\gamma| + |\mathcal{W}_\gamma|)$. We do it independently for every $\gamma \in \Gamma$, performing $\tilde{\mathcal{O}}(|U| + |\mathcal{V}|) = \tilde{\mathcal{O}}(n^2 + |\mathcal{V}|)$ operations in total.

Now consider a query vector $q \in Q$. Let $\gamma \in \Gamma$ be such that $q \equiv \gamma$. We have already showed that the sum of scores for $u \in D + q$ is equal to the sum of scores for $u \in (D + q) \cap U$. Since $(D + q) \cap U = (D + q) \cap U_\gamma$, we see that the result is the sum of scores for such $u \in U_\gamma$, for which $\varphi \times u \geq \varphi \times q$ and $\psi \times u \geq \psi \times q$. If we denote $Q_\gamma = Q \cap (\mathcal{L} + \gamma)$, we see that we can calculate the results for all $q \in Q_\gamma$ by sweeping Q_γ and U_γ in time $\tilde{\mathcal{O}}(|Q_\gamma| + |U_\gamma|)$. We do it independently for every $\gamma \in \Gamma$, performing $\tilde{\mathcal{O}}(|Q| + |U|) = \tilde{\mathcal{O}}(n^2 + |Q|)$ operations in total. □

Lemma 10. *For any simple subparquet U we can find $w_0, \ldots, w_3 \in \mathbb{Z}^2$ such that*

$$|U \cap X| = \sum_{j=0}^{3} (-1)^j |(D + w_j) \cap X|$$

for every $X \subseteq \mathbb{Z}^2$. If U consists of vectors of length $\mathcal{O}(n)$, then $w_0, \ldots, w_3$ are of length $\mathcal{O}(n)$.

Proof. TODO □

We apply Lemma 10 to every U_i and find $w_{i,0}, \ldots, w_{i,3}$ so that we have

$$\sum_{i=0}^{\ell-1} |(U_i + q) \cap V_i| = \sum_{i=0}^{\ell-1} |U_i \cap (V_i - q)| = \sum_{i=0}^{\ell-1} \sum_{j=0}^{3} (-1)^j |(D + w_{i,j}) \cap (V_i - q)| =$$
$$= \sum_{j=0}^{3} (-1)^j \sum_{i=0}^{\ell-1} |(D + q) \cap (V_i - w_{i,j})|.$$

By Lemma 9 we can independently calculate the values $\sum_{i=0}^{\ell-1} |(D + q) \cap (V_i - w_{i,j})|$ for every j by running the algorithm for $\mathcal{V}_j = \{ V_i - w_{i,j} : i \in [\ell] \}$ and Q.

5.6 Proof of Theorem 8

Definition 14 (Lattice graph)**.** For a set $U \subseteq \mathbb{Z}^2$ we define its **lattice graph** $G_U = (U, E_U)$ where

$$E_U = \left\{ \{ u, u + \delta \} : \delta \in \{ \varphi, \psi \}, u \in U, u + \delta \in U \right\}$$

so every vector is connected with its translations by $\varphi, \psi, -\varphi, -\psi$.

Lemma 11. *If U is a spacious subparquet, then G_U is connected.*

Firstly, we partition R into a set of subparquet strings $\mathcal{S}$. For every non-empty $S \in \mathcal{S}$ we consider a lattice graph $G_{\mathrm{dom}(S)}$. If S is not monochromatic, then since $G_{\mathrm{dom}(S)}$ is connected, there must exist a pair of neighboring vectors v, w such that $S(v) \neq S(w)$. We select any such pair and partition S into spacious (or simple if S is simple) subparquet strings S' and S'' such that $v \in S'$ and $w \in S''$. For example if $v = w + \varphi$, then $S' = \{ u : u \in S, \psi \times u \leq \psi \times v \}$ and $S'' = \{ u : u \in S, \psi \times u > \psi \times v \}$. In the cases when $v = w + \delta$ for $\delta \in \{ -\varphi, \psi, -\psi \}$ the construction in similar.

We can recursively partition S' and S'' further until we obtain monochromatic strings. Because R has $\mathcal{O}(k)$-periods φ and ψ, the total number of neighbor pairs v, w such that $S(v) \neq S(w)$ is $\mathcal{O}(k)$ throughout all $S \in \mathcal{S}$. Thus the total number of recursive calls is $\mathcal{O}(k)$ and because $|\mathcal{S}| = \mathcal{O}(k)$, the total number of constructed strings is $\mathcal{O}(k)$. The algorithm can be implemented to work in time $\tilde{\mathcal{O}}(|\mathrm{dom}(R)|)$.

5.7 Proof of Theorem 10

Define

$$a_{\min} = \min\{\,\varphi \times u : u \in T\,\}, \quad a_{\max} = \max\{\,\varphi \times u : u \in T\,\},$$
$$b_{\min} = \min\{\,\psi \times u : u \in T\,\}, \quad b_{\max} = \max\{\,\psi \times u : u \in T\,\}.$$

Lemma 12. *$|\{\,a_{\min}, \ldots, a_{\max}\,\}| \le 2n|\varphi|$ and $|\{\,b_{\min}, \ldots, b_{\max}\,\}| \le 2n|\psi|$.*

Proof. There exist $u, v \in T$ such that $\varphi \times u = a_{\min}$ and $\varphi \times v = a_{\max}$. We have

$$|\{\,a_{\min}, \ldots, a_{\max}\,\}| = a_{\max} - a_{\min} + 1 = \varphi \times v - \varphi \times u + 1 =$$
$$= \varphi \times (v - u) + 1 \le |\varphi||v - u| + 1 \le |\varphi|(2n - 2) + 1 \le 2n|\varphi|$$

The proof is identical for the second inequality. □

Considering Lemma 12, we can partition the set $\{\,a_{\min}, \ldots, a_{\max}\,\}$ into ℓ sets $A_0, \ldots, A_{\ell-1}$, such that each A_i contains some subsequent integers (A_0 the smallest, $A_{\ell-1}$ the largest) and $|A_i| \le \lceil 2n|\varphi|/\ell \rceil$. Specifically, $A_i = \{\,a_{\min} + id, \ldots, a_{\min} + id + d - 1\,\}$, where $d = \lceil |A|/\ell \rceil$. We identically partition the set $\{\,b_{\min}, \ldots, b_{\max}\,\}$ into ℓ sets $B_0, \ldots, B_{\ell-1}$, such that $|B_i| \le \lceil 2n|\psi|/\ell \rceil$. Note that A_i and B_i are non-empty. For every $i, j \in [\ell]$ we construct a simple parquet (recall Definition 9)

$$U_{i,j} = \{\,u : u \in \mathbb{Z}^2, \varphi \times u \in A_i, \psi \times u \in B_j\,\}$$

and for the purpose of the analysis we also define its extension, which includes non-integer points (which we do not explicitly calculate and only use for the proof)

$$U^{\mathbb{R}}_{i,j} = \{\,u : u \in \mathbb{R}^2, \varphi \times u \in A_i, \psi \times u \in B_j\,\}.$$

Observe that $U^{\mathbb{R}}_{i,j}$ are non-empty and $U_{i,j}$ might be. We also define

$$X_{i,j} = \{\,u.x : u \in U^{\mathbb{R}}_{i,j}\,\}, \quad Y_{i,j} = \{\,u.y : u \in U^{\mathbb{R}}_{i,j}\,\}.$$

Lemma 13. *For every i, j and every $u, v \in U^{\mathbb{R}}_{i,j}$, we have $|u - v| = \mathcal{O}(n/\ell)$.*

Proof. Consider any $u, v \in U^{\mathbb{R}}_{i,j}$ for some i, j and denote $w = u - v$. By Lemma 12 and the fact that $\varphi \times u$ and $\varphi \times v$ belong to A_i, we have

$$|\varphi \times w| = |\varphi \times (u - v)| = |\varphi \times u - \varphi \times v| \le |A_i| - 1 = \mathcal{O}(n|\varphi|/\ell).$$

We can similarly show that $|\psi \times w| = \mathcal{O}(n|\psi|/\ell)$. Since φ and ψ are not collinear, there exist $s, t \in \mathbb{R}$, such that $w = s\varphi + t\psi$. Recall that by Theorem 6 we have $|\varphi \times \psi| \ge \frac{1}{2}|\varphi||\psi|$ (since $|\sin\alpha| \ge 1/2$), thus

$$\frac{1}{2}|t||\varphi||\psi| \le |t||\varphi \times \psi| = |\varphi \times (s\varphi + t\psi)| = |\varphi \times w| = \mathcal{O}(n|\varphi|/\ell),$$

which gives us $|t\varphi| = \mathcal{O}(n/\ell)$. We can similarly prove that $|s\psi| = \mathcal{O}(n/\ell)$ and finally

$$|w| = |s\varphi + t\psi| \le |s\varphi| + |t\psi| = \mathcal{O}(n/\ell).$$

□

Lemma 14. *For every $i \in [\ell - 1]$ and $j \in [\ell]$ we have*

$$\min X_{i,j} \leq \min X_{i+1,j}, \quad \min Y_{i,j} \leq \min Y_{i+1,j},$$
$$\max X_{i,j} \leq \max X_{i+1,j}, \quad \max Y_{i,j} \leq \max Y_{i+1,j}$$

and for every $i \in [\ell]$ and $j \in [\ell - 1]$ we have

$$\min X_{i,j} \geq \min X_{i,j+1}, \quad \min Y_{i,j} \leq \min Y_{i,j+1},$$
$$\max X_{i,j} \geq \max X_{i,j+1}, \quad \max Y_{i,j} \leq \max Y_{i,j+1}.$$

Proof. Recall that we selected $\varphi \in \mathcal{Q}_4$ and $\psi \in \mathcal{Q}_1$. We show the first inequality. There exists a point $v \in U^{\mathbb{R}}_{i+1,j}$, such that $v.x = \min X_{i+1,j}$. Select any $h \in A_i$ and consider a point $u \in \mathbb{R}^2$, such that $\varphi \times u = h$ and $\psi \times u = \psi \times v$. Observe that $u \in U^{\mathbb{R}}_{i,j}$. Let $w = v - u$. There exist s, t such that $w = s\varphi + t\psi$. We have

$$s\varphi \times \psi = \psi \times (s\varphi + t\psi) = \psi \times w = \psi \times v - \psi \times u = 0,$$

and since $\varphi \times \psi > 0$, we get $s = 0$, thus $w = t\psi$. Similarly

$$t\varphi \times \psi = \varphi \times t\psi = \varphi \times w = \varphi \times v - \varphi \times u > 0,$$

and since $\varphi \times \psi > 0$, we get $t > 0$. Because $\psi.x \geq 0$, we finally have

$$\min X_{i,j} \leq u.x = v.x - w.x = v.x - t\psi.x \leq v.x = \min X_{i+1,j}.$$

The other inequalities can be proven analogously. □

We now classify each $U_{i,j}$ into one of six types, numbered from 0 to 5. We say that $U_{i,j}$ is of the 0th type when

$$X_{i,j} \times Y_{i,j} \cap (\mathbb{Z}^2 \setminus \mathrm{dom}(T_{\mathbf{a}})) \neq \emptyset.$$

Note that even though we do not explicitly construct the sets $X_{i,j}$ and $Y_{i,j}$, we can easily check the above condition without doing so (TODO). For every $U_{i,j}$ that is not of the 0th type, we have the following classification:

a) If $\min X_{i,j} \geq n/2$ and $\min Y_{i,j} \geq n/2$, then $U_{i,j}$ is of the 1st type (in the upper right quarter).

b) If $\max X_{i,j} < n/2$ and $\min Y_{i,j} \geq n/2$, then $U_{i,j}$ is of the 2nd type (in the upper left quarter).

c) If $\max X_{i,j} < n/2$ and $\max Y_{i,j} < n/2$, then $U_{i,j}$ is of the 3rd type (in the lower left quarter).

d) If $\min X_{i,j} \geq n/2$ and $\max Y_{i,j} < n/2$, then $U_{i,j}$ is of the 4th type (in the lower right quarter).

e) If none of the above is true, then $U_{i,j}$ is of the 5th type.

Lemma 15. *If $U_{i,j}$ is of the 0th type, then* $\mathrm{BorderDis}(U_{i,j}) = \mathcal{O}(n/\ell)$.

Proof. By definition, there exists a point $f \in X_{i,j} \times Y_{i,j} \cap (\mathbb{Z}^2 \setminus \mathrm{dom}(T_{\mathbf{a}}))$. There also exists a point $v \in U^{\mathbb{R}}_{i,j}$ such that $v.x = f.x$ and a point $w \in U^{\mathbb{R}}_{i,j}$ such that $w.y = f.y$. By Lemma 13 we have

$$|v - f| = |v.y - f.y| \leq |v - w| = \mathcal{O}(n/\ell).$$

Since for every $u \in U_{i,j}$, we (by the same lemma) have $|u - v| = \mathcal{O}(n/\ell)$, we get

$$|u - f| \leq |u - v| + |v - f| = \mathcal{O}(n/\ell).$$

□

Lemma 16. *There are $\mathcal{O}(\ell)$ sets $U_{i,j}$ of the 5th type.*

Proof. See Section 5.7.1. □

We construct the set R as the sum of all $U_{i,j} \cap \mathrm{dom}(T_{\mathbf{a}})$, such that $U_{i,j}$ is of the 0th type. By Lemma 15, we have $\mathrm{BorderDis}(R) = \mathcal{O}(n/\ell)$. Note that the set R and all the sets $U_{i,j}$ of the 1-5 types form a partitioning of $\mathrm{dom}(T_{\mathbf{a}})$.

We will now merge the $\mathcal{O}(\ell^2)$ sets of the 1-5 types and construct $\mathcal{O}(\ell)$ larger ones. Since by Lemma 16, the number of sets $U_{i,j}$ of the 5th type is $\mathcal{O}(\ell)$, we will focus on the 1-4 types.

For every $i \in [\ell]$ we construct the sets

$$V_i^1 = \bigcup_{j=0}^{\ell-1} \{\, U_{j,i} : U_{j,i} \text{ is of the 1st type} \,\}, \quad V_i^2 = \bigcup_{j=0}^{\ell-1} \{\, U_{i,j} : U_{i,j} \text{ is of the 2nd type} \,\},$$
$$V_i^3 = \bigcup_{j=0}^{\ell-1} \{\, U_{j,i} : U_{j,i} \text{ is of the 3rd type} \,\}, \quad V_i^4 = \bigcup_{j=0}^{\ell-1} \{\, U_{i,j} : U_{i,j} \text{ is of the 4th type} \,\}.$$

Consider the family $\mathcal{U}$ consisting of all the sets V_i^j for $i \in [\ell]$, $j \in \{1, \ldots, 4\}$ along with all the $U_{i,j}$ sets of the 5th type. Observe that $\mathcal{U} \cup \{R\}$ forms a partitioning of $\mathrm{dom}(T_{\mathbf{a}})$ and that by Lemma 16, $|\mathcal{U}| = \mathcal{O}(\ell)$.

Lemma 17. *$\mathcal{U}$ consists of simple parquets.*

Proof. □

Define $\mathcal{S}$.

Lemma 18. *S has $\mathcal{O}(k)$-periods φ and ψ for every $S \in \mathcal{S}$.*

Proof. □

5.7.1 Proof of Lemma 16

Every set $U_{i,j}$ of the 5th type holds at least one of the followig conditions:

1. $x_{\min} < n/2$ and $x_{\max} \geq n/2$, or
2. $y_{\min} < n/2$ and $y_{\max} \geq n/2$.

We will estimate the number of sets satisfying Condition 1. Consider the points $u_0, \ldots, u_{n-1}$, where $u_h = (n/2, h)$. Observe that every $U_{i,j}$ satisfying Condition 1 must contain some u_h.

References

[1] Amihood Amir and Gary Benson. Two-dimensional periodicity and its applications. In Greg N. Frederickson, editor, *Proceedings of the Third Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 27-29 January 1992, Orlando, Florida, USA*, pages 440–452. ACM/SIAM, 1992.

[2] Amihood Amir and Gary Benson. Two-dimensional periodicity in rectangular arrays. *SIAM J. Comput.*, 27(1):90–106, 1998.

[3] Amihood Amir, Gary Benson, and Martin Farach. An alphabet independent approach to two-dimensional pattern matching. *SIAM J. Comput.*, 23(2):313–323, 1994.

[4] Amihood Amir, Gary Benson, and Martin Farach. Optimal two-dimensional compressed matching. *J. Algorithms*, 24(2):354–379, 1997.

[5] Amihood Amir and Martin Farach. Efficient 2-dimensional approximate matching of half-rectangular figures. *Inf. Comput.*, 118(1):1–11, 1995.

[6] Amihood Amir and Gad M. Landau. Fast parallel and serial multidimensional approximate array matching. *Theor. Comput. Sci.*, 81(1):97–115, 1991.

[7] Amihood Amir, Gad M. Landau, Shoshana Marcus, and Dina Sokol. Two-dimensional maximal repetitions. *Theor. Comput. Sci.*, 812:49–61, 2020.

[8] Amihood Amir, Gad M. Landau, and Dina Sokol. Inplace 2d matching in compressed images. *J. Algorithms*, 49(2):240–261, 2003.

[9] Amihood Amir, Gad M. Landau, and Dina Sokol. Inplace run-length 2d compressed search. *Theor. Comput. Sci.*, 290(3):1361–1383, 2003.

[10] Alberto Apostolico and Valentin E. Brimkov. Fibonacci arrays and their two-dimensional repetitions. *Theor. Comput. Sci.*, 237(1-2):263–273, 2000.

[11] Alberto Apostolico and Valentin E. Brimkov. Optimal discovery of repetitions in 2d. *Discret. Appl. Math.*, 151(1-3):5–20, 2005.

[12] Alberto Apostolico, Laxmi Parida, and Simona E. Rombo. Motif patterns in 2d. *Theor. Comput. Sci.*, 390(1):40–55, 2008.

[13] Ricardo A. Baeza-Yates and Gonzalo Navarro. Fast two-dimensional approximate pattern matching. In Claudio L. Lucchesi and Arnaldo V. Moura, editors, *LATIN '98: Theoretical Informatics, Third Latin American Symposium, Campinas, Brazil, April, 20-24, 1998, Proceedings*, volume 1380 of *Lecture Notes in Computer Science*, pages 341–351. Springer, 1998.

[14] Ricardo A. Baeza-Yates and Mireille Régnier. Fast two-dimensional pattern matching. *Inf. Process. Lett.*, 45(1):51–57, 1993.

[15] Theodore P. Baker. A technique for extending rapid exact-match string matching to arrays of more than one dimension. *SIAM J. Comput.*, 7(4):533–541, 1978.

[16] Piotr Berman, Marek Karpinski, Lawrence L. Larmore, Wojciech Plandowski, and Wojciech Rytter. On the complexity of pattern matching for highly compressed two-dimensional texts. *J. Comput. Syst. Sci.*, 65(2):332–350, 2002.

[17] Richard S. Bird. Two dimensional pattern matching. *Inf. Process. Lett.*, 6(5):168–170, 1977.

[18] Itai Boneh, Dvir Fried, Shay Golan, Matan Kraus, Adrian Miclaus, and Arseny Shur. Searching 2d-strings for matching frames. *CoRR*, abs/2310.02670, 2023.

[19] Panagiotis Charalampopoulos, Jakub Radoszewski, Wojciech Rytter, Tomasz Walen, and Wiktor Zuba. The number of repetitions in 2d-strings. In Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders, editors, *28th Annual European Symposium on Algorithms, ESA 2020, September 7-9, 2020, Pisa, Italy (Virtual Conference)*, volume 173 of *LIPIcs*, pages 32:1–32:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[20] Panagiotis Charalampopoulos, Jakub Radoszewski, Wojciech Rytter, Tomasz Walen, and Wiktor Zuba. Computing covers of 2d-strings. In Pawel Gawrychowski and Tatiana Starikovskaya, editors, *32nd Annual Symposium on Combinatorial Pattern Matching, CPM 2021, July 5-7, 2021, Wrocław, Poland*, volume 191 of *LIPIcs*, pages 12:1–12:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[21] Raphaël Clifford, Allyx Fontaine, Ely Porat, Benjamin Sach, and Tatiana Starikovskaya. The k-mismatch problem revisited. *CoRR*, abs/1508.00731, 2015.

[22] Raphaël Clifford, Allyx Fontaine, Tatiana Starikovskaya, and Hjalte Wedel Vildhøj. Dynamic and approximate pattern matching in 2d. In Shunsuke Inenaga, Kunihiko Sadakane, and Tetsuya Sakai, editors, *String Processing and Information Retrieval - 23rd International Symposium, SPIRE 2016, Beppu, Japan, October 18-20, 2016, Proceedings*, volume 9954 of *Lecture Notes in Computer Science*, pages 133–144, 2016.

[23] Richard Cole, Maxime Crochemore, Zvi Galil, Leszek Gasieniec, Ramesh Hariharan, S. Muthukrishnan, Kunsoo Park, and Wojciech Rytter. Optimally fast parallel algorithms for preprocessing and pattern matching in one and two dimensions. In *34th Annual Symposium on Foundations of Computer Science, Palo Alto, California, USA, 3-5 November 1993*, pages 248–258. IEEE Computer Society, 1993.

[24] Richard Cole, Zvi Galil, Ramesh Hariharan, S. Muthukrishnan, and Kunsoo Park. Parallel two dimensional witness computation. *Inf. Comput.*, 188(1):20–67, 2004.

[25] Maxime Crochemore, Leszek Gasieniec, Ramesh Hariharan, S. Muthukrishnan, and Wojciech Rytter. A constant time optimal parallel algorithm for two-dimensional pattern matching. *SIAM J. Comput.*, 27(3):668–681, 1998.

[26] Maxime Crochemore, Leszek Gasieniec, Wojciech Plandowski, and Wojciech Rytter. Two-dimensional pattern matching in linear time and small space. In Ernst W. Mayr and Claude Puech, editors, *STACS 95, 12th Annual Symposium on Theoretical Aspects of Computer Science, Munich, Germany, March 2-4, 1995, Proceedings*, volume 900 of *Lecture Notes in Computer Science*, pages 181–192. Springer, 1995.

[27] Maxime Crochemore, Costas S. Iliopoulos, and Maureen Korda. Two-dimensional prefix string matching and covering on square matrices. *Algorithmica*, 20(4):353–373, 1998.

[28] Kimmo Fredriksson, Gonzalo Navarro, and Esko Ukkonen. Optimal exact and fast approximate two dimensional pattern matching allowing rotations. In Alberto Apostolico and Masayuki Takeda, editors, *Combinatorial Pattern Matching, 13th Annual Symposium, CPM 2002, Fukuoka, Japan, July 3-5, 2002, Proceedings*, volume 2373 of *Lecture Notes in Computer Science*, pages 235–248. Springer, 2002.

[29] Zvi Galil and Kunsoo Park. Alphabet-independent two-dimensional witness computation. *SIAM J. Comput.*, 25(5):907–935, 1996.

[30] Guilhem Gamard and Gwénaël Richomme. Coverability and multi-scale coverability on infinite pictures. *J. Comput. Syst. Sci.*, 104:258–277, 2019.

[31] Guilhem Gamard, Gwénaël Richomme, Jeffrey O. Shallit, and Taylor J. Smith. Periodicity in rectangular arrays. *Inf. Process. Lett.*, 118:58–63, 2017.

[32] Pawel Gawrychowski, Samah Ghazawi, and Gad M. Landau. Lower bounds for the number of repetitions in 2d strings. In Thierry Lecroq and Hélène Touzet, editors, *String Processing and Information Retrieval - 28th International Symposium, SPIRE 2021, Lille, France, October 4-6, 2021, Proceedings*, volume 12944 of *Lecture Notes in Computer Science*, pages 179–192. Springer, 2021.

[33] Pawel Gawrychowski and Przemyslaw Uznanski. Optimal trade-offs for pattern matching with k mismatches. *CoRR*, abs/1704.01311, 2017.

[34] Juha Kärkkäinen and Esko Ukkonen. Two- and higher-dimensional pattern matching in optimal expected time. *SIAM J. Comput.*, 29(2):571–589, 1999.

[35] Howard J. Karloff. Fast algorithms for approximately counting mismatches. *Inf. Process. Lett.*, 48(2):53–60, 1993.

[36] Richard M. Karp, Raymond E. Miller, and Arnold L. Rosenberg. Rapid identification of repeated patterns in strings, trees and arrays. In Patrick C. Fischer, H. Paul Zeiger, Jeffrey D. Ullman, and Arnold L. Rosenberg, editors, *Proceedings of the 4th Annual ACM Symposium on Theory of Computing, May 1-3, 1972, Denver, Colorado, USA*, pages 125–136. ACM, 1972.

[37] Richard M. Karp and Michael O. Rabin. Efficient randomized pattern-matching algorithms. *IBM J. Res. Dev.*, 31(2):249–260, 1987.

[38] Donald E. Knuth, James H. Morris Jr., and Vaughan R. Pratt. Fast pattern matching in strings. *SIAM J. Comput.*, 6(2):323–350, 1977.

[39] Kamala Krithivasan and R. Sitalakshmi. Efficient two-dimensional pattern matching in the presence of errors. *Inf. Sci.*, 43(3):169–184, 1987.

[40] Kalpana Mahalingam and Palak Pandoh. On the maximum number of distinct palindromic sub-arrays. In Carlos Martín-Vide, Alexander Okhotin, and Dana Shapira, editors, *Language and Automata Theory and Applications - 13th International Conference, LATA 2019, St. Petersburg, Russia, March 26-29, 2019, Proceedings*, volume 11417 of *Lecture Notes in Computer Science*, pages 434–446. Springer, 2019.

[41] Shoshana Marcus and Dina Sokol. 2d lyndon words and applications. *Algorithmica*, 77(1):116–133, 2017.

[42] Filippo Mignosi, Antonio Restivo, and Pedro V. Silva. On fine and wilf's theorem for bidimensional words. *Theor. Comput. Sci.*, 292(1):245–262, 2003.

[43] Shoshana Neuburger and Dina Sokol. Succinct 2d dictionary matching. *Algorithmica*, 65(3):662–684, 2013.

[44] Kunsoo Park. Analysis of two-dimensional approximate pattern matching algorithms. *Theor. Comput. Sci.*, 201(1-2):263–273, 1998.

[45] Jakub Radoszewski, Wojciech Rytter, Juliusz Straszynski, Tomasz Walen, and Wiktor Zuba. Rectangular tile covers of 2d-strings. In Hideo Bannai and Jan Holub, editors, *33rd Annual Symposium on Combinatorial Pattern Matching, CPM 2022, June 27-29, 2022, Prague, Czech Republic*, volume 223 of *LIPIcs*, pages 23:1–23:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[46] Sanjay Ranka and Todd Heywood. Two-dimensional pattern matching with k mismatches. *Pattern Recognit.*, 24(1):31–40, 1991.

[47] Wojciech Rytter. Compressed and fully compressed pattern matching in one and two dimensions. *Proc. IEEE*, 88(11):1769–1778, 2000.

[48] Jorma Tarhio. A sublinear algorithm for two-dimensional string matching. *Pattern Recognit. Lett.*, 17(8):833–838, 1996.

[49] Rui Feng Zhu, Masayuki Nakajima, and Takeshi Agui. The extension of the aho-corasick algorithm to multiple rectangular patterns of different sizes and n-dimensional patterns and text. In *Proceedings of IAPR Workshop on Computer Vision - Special Hardware and Industrial Applications, MVA 1988, Tokyo, Japan, October 12-14, 1988*, pages 185–188, 1988.

[50] Rui Feng Zhu and Tadao Takaoka. A technique for two-dimensional pattern matching. *Commun. ACM*, 32(9):1110–1120, 1989.