Technische Universität Berlin
Electrical Engineering and Computer Science
Institute of Software Engineering and Theoretical Computer Science
Algorithmics and Computational Complexity (AKT)

Algorithmic and Structural Aspects of Matrix Completion Problems

Tomohiro Koana

Thesis submitted in fulfillment of the requirements for the degree
"Master of Science" (M. Sc.) in the field of Computer Science

December 2019

Supervisor and first reviewer:	Prof. Dr. Rolf Niedermeier
Second reviewer:	Prof. Dr. Christian Komusiewicz (Philipps Universität Marburg)
Co-Supervisors:	Vincent Froese

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Die selbstständige und eigenhändige Ausfertigung versichert an Eides statt:

Berlin, den _____ _____

Datum Unterschrift

Zusammenfassung

In dieser Arbeit untersuchen wir kombinatorische Matrixvervollständigungsprobleme basierend auf den Arbeiten von Eiben et al. [Eib+19] and Ganian et al. [Gan+18]. Als Eingabe erhalten wir eine unvollständige Matrix mit n Zeilen und ℓ Spalten, in der Werte für einige Einträge unbekannt sind. Das Ziel ist es, die fehlenden Einträge so zu vervollständigen, dass die Matrix eine gewünschte Eigenschaft erfüllt. Wir betrachten das Ziel, den Radius (maximale Entfernung zu einem Vektor), den lokalen Radius (maximale Entfernung zu einem Vektor in der Matrix) oder den Durchmesser (maximale paarweise Entfernung) der vollständigen Matrix zu minimieren. Wir nennen diese Probleme Minimum Radius / Local Radius / Diameter Matrix Completion (MinRMC / MinLRMC / MinDMC). Wir untersuchen die algorithmische Komplexität dieser Matrixvervollständigungsprobleme anhand eines multivariaten Ansatzes.

Zunächst beschäftigen wir uns mit MinRMC und MinLRMC. Das Problem der Vervollständigung der Matrix ist NP-schwer, auch wenn der (lokale) Radius d der gesuchten Matrix zwei beträgt [HR15]. Für den Fall $d = 1$ entwickeln wir Polynomialzeitalgorithmen, die eine offene Frage von Hermelin und Rozenberg beantworten [HR15]. Obwohl MinRMC auch bei vollständiger Matrix NP-schwer ist, zeigen wir fixed-parameter tractability für den Parameter $d + k$, wobei die maximale Anzahl fehlender Einträge in einer Zeile ist k. Außerdem zeigen wir, dass MinLRMC fixed-parameter tractable für den Parameter k ist.

Dann studieren wir MinDMC. Wir erhalten eine vollständige Dichotomie zwischen lösbaren und unlösbaren Fällen in Bezug auf d und k für binäre Matrizen. Wir entwickeln Polynomzeitalgorithmen für den Fall $d \leq 3$, basierend auf Dezas Theorem [Dez73] aus der Theorie der extremalen Mengenlehre. Auf der negativen Seite beweisen wir die NP-Schwere für den Fall $d \geq 3$. Für den Parameter k zeigen wir, dass MinDMC für $k = 1$ polynomzeitlösbar ist und NP-schwer, wenn $k \geq 2$.

Abstract

In this thesis, we study combinatorial matrix completion problems, following the work of Eiben et al. [Eib+19] and Ganian et al. [Gan+18]. As input, we are given an *incomplete* matrix with n rows and ℓ columns, in which values for some entries are unknown. The goal is to complete the missing entries so that the matrix fulfills some desired property. We consider the objective of minimizing *radius* (maximum distance to some vector), *local radius* (maximum distance to some vector in the matrix), or *diameter* (maximum pairwise distance) of the entire matrix. We call these problems Minimum Radius/Local Radius/Diameter Matrix Completion (MinRMC/MinLRMC/MinDMC). We examine the computational complexity of these matrix completion problems via a multivariate approach.

First, we tackle MinRMC and MinLRMC. The matrix completion problem is NP-hard even if the (local) radius d of the sought matrix is two [HR15]. We provide polynomial-time algorithms for the case of $d = 1$, answering an open question of Hermelin and Rozenberg [HR15]. Although MinRMC is NP-hard even if the matrix is complete, we show fixed-parameter tractability with respect to $d + k$, where k is such that each row vector contains at most k missing entries. Meanwhile, we show that MinLRMC is fixed-parameter tractable with respect to k alone.

Then, we study MinDMC. We obtain a complete dichotomy between tractable and intractable cases in terms of d and k for binary matrices. We develop polynomial-time algorithms for the case $d \leq 3$, based on Deza's theorem [Dez73] from extremal set theory. On the negative side we prove the NP-hardness for the case $d \geq 4$. For the parameter k, we show that MinDMC is polynomial-time when $k = 1$ and NP-hard when $k \geq 2$.

Contents

Chapter 1

Introduction

It is often the case that data can be only partially measured. For the sake of an illustrative example, let us consider the well-known Netflix challenge [BL07]. Netflix users can submit ratings on a subset of provided contents to the video streaming service. Due to abundance of available titles, a typical user cannot rate all movies on Netflix. This results in an *incomplete* matrix where each entry is the rating of a user (row) on a movie (column). The matrix is incomplete, in that values for some entries are unknown. For the Netflix challenge, one would like to find which title most fits the preference of each user. Such a task can be achieved by the recovery of this incomplete matrix. In general, recommending relevant items to users is crucial in modern businesses, because it directly leads to an increase in profit. Inferring missing data points from partially observed data is quite ubiquitous, and its application includes not only recommendation systems but also computer vision [Cab+11; Cab+15; Ji+10; Luo+15], system identification [CP10], remote sensing [Sch86], and localization of IoT (Internet of Things) networks [Ngu+19], to name a few. For any of these applications, one would like to fill in the unknown values of an incomplete matrix such that a certain measure regarding the whole matrix is optimized.

The objective most frequently used in the machine learning community is arguably the rank of the completed matrix. The reasoning for minimizing the rank is based on the assumption that the matrix is structured. In the example of the Netflix challenge, there are perhaps only a few factors that contribute to user preferences. The task of completing a matrix of rank at most $r \in \mathbb{N}$ is known to be NP-hard, even when the input matrix is over the field GF(2) and $r = 3$ [Pee96]. On the positive side, it is known that the matrix of low rank can fully recovered efficiently via semidefinite relaxation under certain feasibility assumptions. [CR12; CT10; Rec11].

In this thesis, we consider the completion of incomplete matrices from a combinatorial standpoint, continuing the work initiated by Eiben et al. [Eib+19]. We assume that each entry in the matrix is from a finite set of symbols, rather than real values. We start by defining several variants of matrix completion problems in the following section.

1.1 Variants of matrix completion problems

In this section, we will formulate several variants of matrix completion problems. First, let us define the matrix completion problem in the most general form. Let Σ be an arbitrary finite set. We are given as input an $(n \times \ell)$-matrix $\mathbf{S}$ over $\Sigma \cup \{*\}$. Here the special character "$*$" denotes a *missing entry* whose value is unknown (clearly, $* \notin \Sigma$). We say that an $(n \times \ell)$-matrix $\mathbf{T}$ over Σ is a completion of $\mathbf{S}$ if $\mathbf{T}$ can be obtained by replacing each missing entry with a symbol from Σ. Our goal is to find a completion that fulfills a certain matrix property Π.

Π-Matrix Completion
Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$.
Question: Is there a completion $\mathbf{T} \in \Sigma^{n \times \ell}$ of $\mathbf{S}$ fulfilling Π?

Of our particular interest are the variants introduced by Eiben et al. [Eib+19]. In order to formulate those variants, we need to introduce some notation. For $v \in \Sigma^\ell$ and $\mathbf{T} \in \Sigma^{n \times \ell}$, we write $v \in \mathbf{T}$ if there is a row index $i \in \{1, \ldots, n\}$ with $\mathbf{T}[i] = v$. We say that the set $\{\mathbf{T}_1, \ldots, \mathbf{T}_c\}$ of matrices is a *clustering* of $\mathbf{T} \in \Sigma^{n \times \ell}$ if each matrix $\mathbf{T}_j$ is a submatrix of $\mathbf{T}$ and each row vector of $\mathbf{T}$ is a row vector of some matrix $\mathbf{T}_j$. We refer to each submatrix as a *cluster*. For vectors $u, v \in \Sigma^\ell$, we denote by $\delta(u, v)$ the Hamming distance between u and v. Let $\mathbf{T}[i]$ denote the i-th row vector of $\mathbf{T}$. Somewhat abusing notation, let $\delta(v, \mathbf{T}) := \max_{i \in \{1,\ldots,n\}} \delta(v, \mathbf{T}[i])$ and $\delta(\mathbf{T}) := \max_{i,i' \in \{1,\ldots,n\}} \delta(\mathbf{T}[i], \mathbf{T}[i'])$. Eiben et al. [Eib+19] considered the following three variants of Π-Matrix Completion albeit under different names:

Minimum Radius Clustering Matrix Completion (MinRCMC)
Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $c, d \in \mathbb{N}$.
Question: Is there a completion $\mathbf{T}$ of $\mathbf{S}$ that admits a clustering $\{\mathbf{T}_1, \ldots, \mathbf{T}_c\}$ such that there exists $v_j \in \Sigma^\ell$ with $\delta(v_j, \mathbf{T}_j) \leq d$ for each cluster $\mathbf{T}_j$?

Minimum Local Radius Clustering Matrix Completion (MinLRCMC)
Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $c, d \in \mathbb{N}$.
Question: Is there a completion $\mathbf{T}$ of $\mathbf{S}$ that admits a clustering $\{\mathbf{T}_1, \ldots, \mathbf{T}_c\}$ such that there exists $v_j \in \mathbf{T}_j$ with $\delta(v_j, \mathbf{T}_j) \leq d$ for each cluster $\mathbf{T}_j$?

Minimum Diameter Clustering Matrix Completion (MinDCMC)
Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $c, d \in \mathbb{N}$.
Question: Is there a completion $\mathbf{T}$ of $\mathbf{S}$ that admits a clustering $\{\mathbf{T}_1, \ldots, \mathbf{T}_c\}$ such that $\delta(\mathbf{T}_j) \leq d$ for each cluster $\mathbf{T}_j$?

Eiben et al. [Eib+19] proved fixed-parameter tractability for all three variants above, with respect to the combined parameter of $|\Sigma|$, c, d, and r. Here the parameter r is the minimum number of rows and columns necessary to cover all missing entries. Their fixed-parameter algorithm is based on kernelization. However, the kernel obtained is due to the so-called sunflower lemma [ER60] and of super-exponential size. Hence, its practicality is somewhat questionable. Eiben et al. [Eib+19] also proved that dropping any of c, d, or r results in parameterized intractability even if $|\Sigma| = 2$. This suggests

−	α		$>$	1
	γ	4.2		2
−		7.3		1
+	β	4.2	$>$	0

−	α	4.2	$>$	1
−	γ	4.2	$>$	2
−	β	7.3	$>$	1
+	β	4.2	$>$	0

−	β	4.2	$>$	1

−	α	4.2	$>$	1
−	γ	4.2	$>$	2
−	γ	7.3	$>$	1
+	β	4.2	$>$	0

−	γ	4.2	$>$	2

Figure 1.1: An example of MinRMC and MinLRMC. The input matrix is depicted in the left. Optimal solutions for MinRMC ($d = 2$) and MinLRMC ($d = 3$) are drawn in the middle and right, respectively. Every entry different from the corresponding entry in the solution vector is marked by gray.

	β	4.2	$>$	0
	α	4.2	$<$	1
−	α	7.3	$<$	1
+	β		$>$	0

−	β	4.2	$>$	0
+	α	4.2	$<$	1
−	α	7.3	$<$	1
+	β	7.3	$>$	0

Figure 1.2: An illustration of MinDMC with the input matrix (left) and its completion (right). Note that the missing entries in the first column must be filled by different symbols when $d = 4$.

that these clustering problems are computationally quite hard; Perhaps their hardness stems from the fact that the clustering problems are NP-hard even for complete data (note that when the input matrix $\mathbf{S}$ is complete, we have $r = 0$). The following are known hardness results for the case $r = 0$:

- MinRCMC is NP-hard even if $|\Sigma| = 2$, $c = 1$, and $r = 0$ [FL97].
- MinRCMC is NP-hard even if $|\Sigma| = 2$, $d = 1$, and $r = 0$ [Ami+14].
- MinLRCMC is NP-hard even if $\ell = 2$ and $r = 0$ [Ami+14].
- MinDCMC is NP-hard even if $c = 3$ and $r = 0$ [Eib+19].

Given the hardness of clustering problems (the case $c > 1$), we restrict our attention to the case $c = 1$ in this thesis, hoping that it will result in practically useful algorithms. Formally, we study the following variants of Π-Matrix Completion (see Figures 1.1 and 1.2 for illustrative examples):

Minimum Radius Matrix Completion (MinRMC)

Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$.

Question: Is there a completion $\mathbf{T} \in \Sigma^{n \times \ell}$ of $\mathbf{S}$ such that $\delta(v, \mathbf{T}) \leq d$ for some vector $v \in \Sigma^{\ell}$?

	1	1	1	1
1		1	1	1
1	1		1	1
1	1	1		1

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Figure 1.3: A matrix in which k is small but r is large (left) and vice versa (right). In the left matrix, each row vector has exactly one missing entry but four rows (or columns) are needed to cover all the missing entries. In the right matrix, one row is sufficient to cover all missing entries.

We remark that this variant is also known as Closest String with Wildcards [HR15] and there are some known results (see Table 1.1).

Minimum Local Radius Matrix Completion (MinLRMC)
Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$.
Question: Is there a completion $\mathbf{T} \in \Sigma^{n \times \ell}$ of $\mathbf{S}$ such that $\delta(v, \mathbf{T}) \leq d$ for some vector $v \in \mathbf{T}$?

Minimum Diameter Matrix Completion (MinDMC)
Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$.
Question: Is there a completion of $\mathbf{T} \in \Sigma^{n \times \ell}$ of $\mathbf{S}$ such that $\delta(\mathbf{T}) \leq d$?

Note that fixed-parameter tractability for the clustering variant [Eib+19] implies fixed-parameter tractability for MinRMC, MinLRMC, and MinDMC with respect to $d + r$ but its running time is at least doubly exponential in terms of $d + r$. To aim for practical algorithms, we consider an alternative parameterization k, the maximum number of missing entries in any row vector. Observe that the parameter k is incomparable to r (see Figure 1.3 for an illustration). If missing entries only appear on the diagonal of $\mathbf{S}$, then we have $k = 1$ and $r = \min(n, \ell)$. On the contrary, if missing entries only appear on the i-th row for some $i \in \{1, \ldots, n\}$ and all entries are missing in $\mathbf{S}[i]$, then we have $k = \ell$ and $r = 1$. In this thesis, we will conduct multivariate complexity analysis [Nie10] of MinRMC, MinLRMC, and MinDMC with focus on d and k.

1.2 Our contributions

In Chapter 3, we investigate the computational complexity of MinRMC. Table 1.1 summarizes known results and our results for MinRMC. We provide three main contributions for MinRMC. The first is a linear-time algorithm for the case $d = 1$. Our algorithm solves MinRMC even if the alphabet size is arbitrarily large. This answers an open question of Hermelin and Rozenberg [HR15]. The next contribution is a fixed-parameter algorithm for MinRMC with respect to $d + k$. Even though fixed-parameter tractability of MinRMC with respect to $d + k$ was claimed by Hermelin and Rozenberg

Table 1.1: Overview of previously known results and our results for MinRMC and MinLRMC. Here we use the following notation: n—number of rows, ℓ—number of columns, $|\Sigma|$—alphabet size, d—distance bound, k—maximum number of missing entries in any row vector.

Parameter	MinRMC	Reference	MinLRMC	Reference
n	$O^*(2^{2^{O(n \log n)}})$	[HR15]	$O^*(2^{O(n^2 \log n)})$	Lem 3.1
	$O^*(2^{O(n^2 \log n)})$	[KKM17]		
ℓ	$O^*(2^{\ell^2/2})$	[HR15]	$O^*(\ell^\ell)$	Cor 3.6
	$O^*(\ell^\ell)$	Cor 3.5		
$d = 1$	$O(n\ell^2)$ for $\|\Sigma\| = 2$	[HR15]	$O(n^2\ell)$	Cor 3.3
	$O(n\ell)$	Thm 3.2		
$d = 2$	NP-hard for $\|\Sigma\| = 2$	[HR15]	NP-hard for $\|\Sigma\| = 2$	trivial
k	NP-hard for $k = 0$	[FL97]	$O^*(k^k)$	Cor 3.6
$d + k$	$O^*((d+1)^{d+k})$	Thm 3.8		
$d + k + \|\Sigma\|$	$O^*(\|\Sigma\|^k \cdot d^d)$	[HR15]	$O^*(\|\Sigma\|^k)$	trivial
	$O^*(2^{4d+k} \cdot \|\Sigma\|^{d+k})$	Thm 3.11		

Table 1.2: Overview of our results for MinDMC. Here we use the following notation: n—number of rows, ℓ—number of columns, $|\Sigma|$—alphabet size, d—distance bound, k—maximum number of missing entries in any row vector.

Parameter	Result	Reference
n	$O(n\ell + 2^{2^{O(n \log n)}})$-time solvable	Theorem 4.2
$d = 1$	$O(n\ell)$-time solvable	Lemma 4.3
$d = 2$	$O(n\ell)$-time solvable when $\|\Sigma\|$ is a constant	Theorem 4.12
	$O(\|\Sigma\|^6 \cdot n^3 + n\ell)$-time solvable	Theorem 4.14
$d = 3$	$O(n\ell^4)$-time solvable when $\|\Sigma\| = 2$	Theorem 4.19
$d \geq 4$	NP-hard even for $\|\Sigma\| = 2$	Theorem 4.20
$k = 1$	$O(n^2\ell)$-time solvable	Theorem 4.21
$k = 2$	NP-hard even for $\|\Sigma\| = 2$	Theorem 4.22

[HR15], we will illustrate an error in their algorithm and provide an alternative algorithm. Lastly, we give an algorithm that is more efficient our fixed-parameter algorithm when alphabet size is sufficiently small.

We also obtain some results for MinLRMC in Chapter 3. In particular, we show that MinLRMC can be solved in polynomial time when $d = 1$. We have a dichotomy result, as MinLRMC is NP-hard for $d = 2$ (it immediately follows from the NP-hardness of MinRMC for $d = 2$, because any MinRMC instance $(\mathbf{S}, d)$ with $\mathbf{S}[i] = *^\ell$ for some $i \in [n]$ is equivalent to MinLRMC instance $(\mathbf{S}, d)$). Moreover, we show that MinLRMC is fixed-parameter tractable with respect to k. This is in sharp contrast to MinRMC, which is NP-hard even for $k = 0$. The results for MinLRMC are also summarized in Table 1.1.

We study MinDMC in Chapter 4. See Table 1.2 for an overview of our results. First, we show that MinDMC is fixed-parameter tractable with respect to the number n of rows. Whereas fixed-parameter tractability with respect to n is rather straightforward to prove, we were unable to determine the parameterized complexity of MinDMC with respect to the number ℓ of columns. In Section 4.2, we consider the parameter d. We prove that MinDMC can be solved in polynomial time when $|\Sigma| = 2$ and $d \leq 3$. Our method is based on a theorem by Deza [Dez73] from extremal set theory. To the best of our knowledge, our algorithm is the second result to prove tractability by exploiting Deza's theorem, next to Froese et al. [Fro+16]. Finally in Section 4.3, we investigate the computational complexity with respect to k. We prove that MinDMC can be solved in polynomial time when $k = 1$, even for arbitrary alphabet size. Surprisingly, we reveal that it probably is the only tractable case, when the parameter k is considered alone: Indeed, we show that MinDMC is NP-hard even for the case $|\Sigma| = 2$ and $k = 2$.

Chapter 2

Preliminaries

2.1 Basic notations

For any $m, n \in \mathbb{N}$ with $m \leq n$, we use $[m, n]$ to denote the set $\{m, \ldots, n\}$. In particular, we use $[n]$ to denote $[1, n]$. Given a predicate ϕ, we use a function $\chi[\phi]$ that is equal to 1 if ϕ evaluates to true and 0 otherwise. We sometimes use the O^* notation, which omits a factor polynomial in the input size.

Matrices. Let $\mathbf{T} \in \Sigma^{n \times \ell}$ be an $(n \times \ell)$-matrix over a finite alphabet Σ. Let $i \in [n]$ and $j \in [\ell]$. We use $\mathbf{T}[i, j]$ to denote the character in the i-th row and j-th column of $\mathbf{T}$. We use $\mathbf{T}[i, :]$ (or $\mathbf{T}[i]$ in short) to denote the *row vector* $(\mathbf{T}[i, 1], \ldots, \mathbf{T}[i, \ell])$ and $\mathbf{T}[:, j]$ to denote the *column vector* $(\mathbf{T}[1, j], \ldots, \mathbf{T}[n, j])^T$. For any subsets $I \subseteq [n]$ and $J \subseteq [m]$, we write $\mathbf{T}[I, J]$ to denote the submatrix obtained by omitting rows in $[n] \setminus I$ and $[\ell] \setminus J$ from $\mathbf{T}$. We abbreviate $\mathbf{T}[I, [\ell]]$ and $\mathbf{T}[[n], J]$ as $\mathbf{T}[I, :]$ (or $\mathbf{T}[I]$ for short) and $\mathbf{T}[:, J]$, respectively. We use a special character $*$ for a *missing* entry. A matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ that contains a missing entry is said to be a *incomplete* matrix. We say that $\mathbf{T} \in \Sigma^{n \times \ell}$ is a *completion* of $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ if either $\mathbf{S}[i, j] = *$ or $\mathbf{S}[i, j] = \mathbf{T}[i, j]$ holds for all $i \in [n]$ and $j \in [\ell]$.

We describe two preprocessing algorithms on the input matrix (see Figure 2.1 for an illustration). One is *normalization* due to Gramm, Niedermeier, and Rossmanith [GNR03]. Normalization reduces the alphabet size such that $|\Sigma| \leq n$. Let $\Sigma_j \subseteq \Sigma$ be the set of symbols from the alphabet on the j-th column of $\mathbf{S}$ for each $j \in [\ell]$. Since $|\Sigma_j| \leq n$, there is an injective function $\pi_j \colon \Sigma_j \to [n]$. We rewrite the entry of $\mathbf{S}[i, j]$ with $\pi_j(\mathbf{S}[i, j])$ for each $i \in [n]$ and $j \in [\ell]$ with $\mathbf{S}[i, j] \neq *$. Every entry in the resulting matrix is in $[n]$ or $*$ and hence we have $|\Sigma| \leq n$. The other is the removal of so-called *dirty* columns, again using the terms of Gramm, Niedermeier, and Rossmanith [GNR03]. For $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $j \in [\ell]$, we call $\mathbf{S}[:, j]$ *dirty* if the column vector contains at least two distinct symbols from the alphabet (not counting $*$). It is easy to see that a dirty column is irrelevant for all variants of matrix completion problems under consideration. Thus, all dirty columns can be removed from the input matrix. Note that normalizing the input matrix and removing dirty columns can be done in linear time and we sometimes assume that these preprocessing steps have been applied on the input matrix.

	Y	A	<		+
2.3	Y			3	−
1.7	Y	A	>		−
2.3	X	A	<	3	+
1.7	Z		>	3	+

	1	1	1
1	1		2
2	1	2	2
1	2	1	1
2	3	2	1

Figure 2.1: An illustration of preprocessing on the input matrix. After applying preprocessing the matrix in the left we obtain the matrix in the right. The columns marked by thick lines are dirty columns.

Vectors. Let $v, v'(\Sigma \cup \{*\})^\ell$ be row vectors and let $\sigma \in \Sigma \cup \{*\}$ be some character. We write $P_\sigma(v)$ to denote the set $\{j \in [\ell] \mid v[j] = \sigma\}$ of column indices where the corresponding entries are σ. We write $Q(v, v')$ to denote the set $\{j \in [\ell] \mid v[j] \neq v[j']\}$ of column indices where the corresponding entries disagree. The binary operation $v \oplus v'$ replaces the missing entries of v with the character in v' in the corresponding position, given that v' contains no missing entry. We sometimes use string notation to represent a row vector. For instance, the strings $\sigma_1\sigma_2\sigma_3$ and σ^3 represent row vectors $(\sigma_1, \sigma_2, \sigma_3)$ and (σ, σ, σ), respectively.

Hamming Distance. For $v, v \in (\Sigma \cup \{*\})^\ell$, the *Hamming distance* between row vectors $v, v' \in \Sigma^\ell$ is $\delta(v, v') := |Q(v, v')|$. When we only consider row vectors without missing entries, δ obeys the triangle inequality: $\delta(x, z) \leq d(x, y) + d(y, z)$ holds for any row vectors $x, y, z \in \Sigma^\ell$. However, δ does not follow the triangle inequality in general; for instance, $\delta(0^\ell, 1^\ell) \leq \delta(0^n, *^\ell) + \delta(*^n, 1^\ell)$ does not hold.

Let $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $v \in (\Sigma \cup \{*\})^\ell$. For the sake of brevity, we sometimes write $\max_{i \in [n]} \delta(u, \mathbf{S}[i])$ as $\delta(u, \mathbf{S})$ and $\max_{i,i' \in [n]} \delta(\mathbf{S}[i], \mathbf{S}[i'])$ as $\delta(\mathbf{S})$. For a subset $J \subseteq [\ell]$, we also use $\delta_J(\mathbf{S}[i], \mathbf{S}[i'])$ as a shorthand for $\delta(\mathbf{S}[i, J], \mathbf{S}[i', J])$.

2.2 Fixed-parameter tractability

A *parameterized problem* Π is a set of instances $\Sigma^* \times \mathbb{N}$, where k is called the *parameter* of the instance. A parameterized problem is *fixed-parameter tractable* if whether $(I, k) \in \Pi$ can be determined in time $f(k) \cdot |I|^{O(1)}$. Here $f \colon \mathbb{N} \to \mathbb{N}$ is an arbitrary computable function. An algorithm with running time $f(k) \cdot |I|^{O(1)}$ is said to be a *fixed-parameter algorithm*.

Here we provide a few common techniques for showing fixed-parameter tractability. For a more exhaustive list, we refer to the standard textbooks in parameterized complexity [Cyg+15; DF13; FG06; Nie06].

Search Tree Algorithms. Also referred to *branching* algorithms, search tree algorithms are probably the most common among fixed-parameter algorithms. A search tree

algorithm is a recursive procedure. In each recursion, the algorithm divides the problem into a number of subproblems, each of which the algorithm tries to solve recursively. The *search tree* is a rooted tree representation of recursions. We obtain a fixed-parameter algorithm if the algorithm spends polynomial time on each recursion and the maximum degree and the depth of the search tree is bounded by some function of the parameter k.

Kernelization. For an instance (I, k), a *problem kernel* of I is an instance (I', k') such that (i) $(I, k) \in \Pi$ if and only if $(I', k') \in \Pi$ and (ii) $|I'| + k' \leq g(k)$ for some computable function $g\colon \mathbb{N} \to \mathbb{N}$. A *kernelization* is a polynomial-time algorithm that computes a problem kernel. It is well known that a parameterized problem is fixed-parameter tractable if and only if there is a kernelization algorithm for the problem.

Integer Linear Programming. One can also prove fixed-parameter tractability via reduction to Integer Linear Programming (ILP).

Integer Linear Programming (ILP)
Input: A matrix $\mathbf{A} \in \mathbb{Z}^{n \times p}$ and $\mathbf{b} \in \mathbb{Z}^n$.
Question: Is there a vector $\mathbf{x} \in \mathbb{Z}^p$ such that $\mathbf{Ax} \leq \mathbf{b}$?

It is known that ILP is fixed-parameter tractable with respect to the number of variables.

Lemma 2.1 ([FT87; Kan87; LJ83])**.** ILP *can be solved in time* $O(p^{3p/2+o(p)} \cdot L)$*, where* p *is the number of variables and* L *is the size of* ILP*.*

Hence, a parameterized problem (I, k) is fixed-parameter tractable if there is an ILP formulation using $h(k)$ variables for some function $h\colon \mathbb{N} \to \mathbb{N}$.

2.3 Complexity of satisfiability problems

The Satisfiability problem or SAT for short asks whether there exists a satisfying truth assignment to a given Boolean formula in conjunctive normal form. Recall that a Boolean formula ϕ is in conjunctive normal form if it is a conjunction of clauses where each clause is a disjunction of literals.

k-SAT
Input: A boolean formula ϕ in conjunctive normal form, where each clause of ϕ contains at most k distinct literals.
Question: Is there a satisfying truth assignment to ϕ?

It is well-known that k-SAT is NP-complete when $k \geq 3$. In order to prove that a problem is NP-hard, one has to provide a polynomial-time reduction to a NP-hard problem. In such NP-hardness proofs, 3-SAT is one of the most commonly used problems.

The widely believed assumption that P $\neq$ NP excludes the existence of polynomial-time algorithms for NP-hard problems. The Exponential Time Hypothesis (ETH) of Impagliazzo and Paturi [IP01] is a stronger assumption, which states that 3-SAT cannot be solved in subexponential time in the number of different variables and clauses in the formula. With this assumption at hand, we can prove more fine-grained lower bounds

on the running time. For instance, there is no algorithm for Closest String (a special case of MinRMC where the input matrix is complete) with running time $d^{o(d)} \cdot (n\ell)^{O(1)}$ or $|\Sigma|^{o(d)} \cdot (n\ell)^{O(1)}$ unless the ETH breaks.

Conjecture 1 (Exponential Time Hypothesis [IP01])**.** 3-SAT *cannot be solved in time* $2^{o(n+m)} \cdot (n+m)^{O(1)}$*, where* n *and* m *denote the number of variables and clauses, respectively.*

A superset of the authors also introduced even stronger hypothesis called Strong Exponential Time Hypothesis (SETH). Assuming the SETH, one can even prove lower bounds for polynomial-time solvable problems. The maximum pairwise distance on a n-vertex graph cannot be computed in $O(n^{2-\varepsilon})$ for $\varepsilon > 0$ under SETH [RW13].

Conjecture 2 (Strong Exponential Time Hypothesis [IPZ01])**.** SAT *cannot be solved in time* $(2-\varepsilon)^n \cdot (n+m)^{O(1)}$ *for any* $\varepsilon > 0$*, where* n *and* m *denote the number of variables and clauses, respectively.*

We refer to [Cyg+15, Chapter 14] for more computational lower bounds based on the ETH and the SETH.

Despite the widely believe assumption that k-SAT has no subexponential algorithm for $k \geq 3$, there are linear-time algorithms for 2-SAT [APT79; EIS75; Kro67]. Hence we can obtain an efficient algorithm via a reduction to 2-SAT. In such a reduction, it is helpful to have an encoding of the *at-most-one* constraint. In other words, we would like to ensure that at most one of some set of literals is satisfied. Let $L = \{l_1, \ldots, l_m\}$ be the set of m literals. A naïve encoding of such a constraint is

$$\bigwedge_{1 \leq i < j \leq m} (\neg l_i \vee \neg l_j),$$

which requires $O(m^2)$ binary clauses. We employ a known trick to efficiently encode the at-most-one constraint with $O(m)$ clauses [Che10; Sin05; SW93]. The at-most-one encoding by Chen [Che10] requires $2m + 4\sqrt{m} + O(\sqrt[4]{m})$ clauses. Here we will use the simpler encoding with $3m - 1$ clauses, given by Sinz [Sin05] and Stamm-Wilbrandt [SW93] (note that both encodings have the same size asymptotically). It uses additional $m - 1$ variables $\{r_1, \ldots, r_{m-1}\}$ and it is defined as follows:

$$\begin{aligned} C_{\leq 1}(L) =& (\neg l_1 \vee r_1) \wedge (\neg l_m \vee \neg r_{m-1}) \\ & \wedge \bigwedge_{2 \leq j \leq m-1} ((\neg l_j \vee \neg r_{j-1}) \wedge (\neg l_j \vee r_j) \wedge (\neg r_{j-1} \vee r_j)) \,. \end{aligned}$$

We prove that $C_{\leq 1}$ encodes the at-most-one constraint indeed in the following lemma.

Lemma 2.2. *At most one literal in* $L = \{l_1, \ldots, l_m\}$ *is true if and only if* $C_{\leq 1}(L)$ *is satisfiable.*

Proof. ($\Rightarrow$) We can simplify $C_{\leq 1}(L)$ as follows, depending on which literal evaluates to true:

$$C_{\leq 1}(L) = \begin{cases} R & \text{if none of } l_1, \ldots, l_m \text{ is true} \\ (\neg l_1 \vee r_1) \wedge R & \text{if } l_1 \text{ is true} \\ \neg r_{k-1} \wedge r_k \wedge R & \text{if } l_k \text{ is true for } k \in \{2, \ldots, m-1\} \\ (\neg l_m \vee \neg r_m) \wedge R & \text{if } l_m \text{ is true,} \end{cases}$$

where we denote

$$R = \bigwedge_{2 \leq j \leq m-1} (\neg r_{j-1} \vee r_j).$$

Observe that a truth assignment to $r_1, \ldots, r_{m-1}$ satisfies R if $r_1, \ldots, r_{k-1}$ are false and $r_k, \ldots, r_{m-1}$ are true for some $k \in [m]$. To see why, note that the clause $(\neg r_{j-1} \vee r_j)$ is satisfied by $\neg r_{j-1}$ for $j \in [2, k]$ and by r_j for $j \in [k, m-1]$ in such a truth assignment. Hence, if l_k is set to true and all other literals $l_1, \ldots, l_{k-1}, l_{k+1}, \ldots, l_m$ are false, then assigning false to $r_1, \ldots, r_{k-1}$ and true to $r_k, \ldots, r_{m-1}$ satisfies R.

($\Leftarrow$) Assume for contradiction that there exists a truth assignment φ satisfying R in which at least two literals are true. Let $k < k' \in [m]$ be the two distinct indices satisfying $\varphi(l_k) = \varphi(l_{k'}) = 1$. Note that $\varphi(r_k) = 1$ due to the clause $(\neg l_k \vee r_k)$ and that $\varphi(r_{k'-1}) = 0$ due to the clause $(\neg l_{k'} \vee \neg r_{k'-1})$. The clause $(\neg r_{j-1} \vee r_j)$ implies that if $\varphi(r_{j-1}) = 1$, then $\varphi(r_j) = 1$ for $j \in [2, m-1]$. By using this argument repeatedly, we obtain a contradiction on the value of $\varphi(r_{k'})$. $\square$

Chapter 3

Radius Minimization

In this chapter, we investigate Minimum Radius Matrix Completion (MinRMC) and Minimum Local Radius Matrix Completion (MinLRMC). Recall that radius refers to the maximum distance to some arbitrary row vector v, whereas local radius refers to the maximum distance to some row vector v from the matrix. Note that once the vector v is found, the matrix should be completed such that each row vector matches v wherever possible. Thus, MinRMC and MinLRMC can be formulated as follows (see Figure 3.1 for an example).

Minimum Radius Matrix Completion (MinRMC)

Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$.
Question: Is there a row vector $v \in \Sigma^{\ell}$ such that $\delta(v, \mathbf{S}) \leq d$?

Minimum Local Radius Matrix Completion (MinLRMC)

Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$.
Question: Is there a row vector $v \in \Sigma^{\ell}$ such that $\delta(v, \mathbf{S}) \leq d$ and $\delta(v, \mathbf{S}[i]) = 0$ for some $i \in [n]$?

In Section 3.1, we present a linear-time algorithm for MinRMC for the case $d = 1$. It is a significant improvement over the previously known algorithm [HR15], which only works for binary alphabet and runs in super-linear time. In Section 3.2, we will give a more efficient fixed-parameter algorithm (compared to the algorithm by Hermelin and Rozenberg [HR15]) with respect to the number ℓ of columns. As a byproduct, we obtain fixed-parameter tractability for MinLRMC with respect to k.

The special case of MinRMC where $k = 0$ (recall that k denotes the maximum number of missing entries in any row vector) is known as Closest String. It has been extensively studied from a (parameterized) algorithmic perspective due to its great relevance in computational biology. Its versatile applications include universal PCP primer design [Dop+93; Lan+03; Luc+91; PH96], genetic probe design [Lan+03], antisense drug design [Den+03; Lan+03], finding unbiased consensus of a protein family [Ben+97], and motif finding [Lan+03; PMP01; PS00]. At the heart of all of these applications lies finding a sequence similar to given DNA or protein sequences.

In Section 3.3, we adapt two known algorithms for Closest String to MinRMC. The first algorithm was given by Gramm, Niedermeier, and Rossmanith [GNR03], who

−	α		>	1
	γ	4.2		2
−		7.3		1
+	β	4.2	>	0

−	α	4.2	>	1
−	γ	4.2	>	2
−	β	7.3	>	1
+	β	4.2	>	0

−	β	4.2	>	1

−	α	4.2	>	1
−	γ	4.2	>	2
−	γ	7.3	>	1
+	β	4.2	>	0

−	γ	4.2	>	2

Figure 3.1: An example of MinRMC and MinLRMC. The input matrix is depicted in the left. Optimal solutions for MinRMC ($d = 2$) and MinLRMC ($d = 3$) are drawn in the middle and right, respectively. Every entry differs from the corresponding entry in the solution vector is marked by gray.

proved fixed-parameter tractability for parameter d with a search tree algorithm running in time $O(n\ell + nd^{d+1})$. The second algorithm, which is more efficient when the alphabet size is small (which is often the case for applications in biology) with running time $O(n\ell + (16|\Sigma|)^d \cdot nd)$, was developed by Ma and Sun [MS09]. Notably we show fixed-parameter tractability with respect to $d + k$, extending the algorithm of Gramm, Niedermeier, and Rossmanith [GNR03] (the fixed-parameter tractability was claimed by Hermelin and Rozenberg [HR15, Theorem 4] but we illustrate an error in their proof). The adaptation of the algorithm by Ma and Sun [MS09] yields a more efficient algorithm algorithm in the case of small alphabet size with running time $O(n\ell + 2^{4d+k} \cdot |\Sigma|^{d+k} \cdot n(d + k))$. This beats the previously known algorithm [HR15] running in time $O(|\Sigma|^k \cdot d^d \cdot n\ell)$ whenever $|\Sigma| < d/2^{k/d+4}$.

See Table 3.1 for an overview of results for MinRMC and MinLRMC given in this section. For technical reasons, we work with the following generalization of MinRMC, in which the distance bound can be specified for each row vector.

Radius Constraint Matrix Completion (RCMC)
Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d_1, \ldots, d_n \in \mathbb{N}$.
Question: Is there a row vector $v \in \Sigma^\ell$ such that $\delta(v, \mathbf{S}[i]) \leq d_i$ for each $i \in [n]$?

Note that there is a simple polynomial-time reduction from MinLRMC to RCMC. Let $(\mathbf{S}, d)$ be an instance of MinLRMC. For each $i \in [n]$, we solve RCMC instance $(\mathbf{S}, d_1, \ldots, d_n)$ where $d_{i'} = d$ for each $i' \in [n] \setminus \{i\}$ and $d_i = 0$. We return **Yes** if and only if at least one RCMC instance is a **Yes** instance. The correctness of this procedure is trivial. It follows that if RCMC can be solved in time $O(f(n + \ell))$ for some function $f \colon \mathbb{N} \to \mathbb{N}$, then MinLRMC can be solved in time $O(n \cdot f(n+\ell))$. Since the algorithm by Knop, Koutecký, and Mnich [KKM17] solves RCMC in time $O^*(2^{O(n^2 \log n)})$, we obtain the following lemma.

Lemma 3.1. MinLRMC *can be solved in time* $O^*(2^{O(n^2 \log n)})$.

Table 3.1: Overview of previously known results and our results for MinRMC and MinLRMC. Here we use the following notation: n—number of rows, ℓ—number of columns, $|\Sigma|$—alphabet size, d—distance bound, k—maximum number of missing entries in any row vector.

Parameter	MinRMC	Reference	MinLRMC	Reference
n	$O^*(2^{2^{O(n \log n)}})$	[HR15]	$O^*(2^{O(n^2 \log n)})$	Lem 3.1
	$O^*(2^{O(n^2 \log n)})$	[KKM17]		
ℓ	$O^*(2^{\ell^2/2})$	[HR15]	$O^*(\ell^\ell)$	Cor 3.6
	$O^*(\ell^\ell)$	Cor 3.5		
$d = 1$	$O(n\ell^2)$ for $\|\Sigma\| = 2$	[HR15]	$O(n^2\ell)$	Cor 3.3
	$O(n\ell)$	Thm 3.2		
$d = 2$	NP-hard for $\|\Sigma\| = 2$	[HR15]	NP-hard for $\|\Sigma\| = 2$	trivial
k	NP-hard for $k = 0$	[FL97]	$O^*(k^k)$	Cor 3.6
$d + k$	$O^*((d+1)^{d+k})$	Thm 3.8		
$d + k + \|\Sigma\|$	$O^*(\|\Sigma\|^k \cdot d^d)$	[HR15]	$O^*(\|\Sigma\|^k)$	trivial
	$O^*(2^{4d+k} \cdot \|\Sigma\|^{d+k})$	Thm 3.11		

3.1 Linear-time algorithm for the case $d = 1$

Hermelin and Rozenberg [HR15, Theorem 6] gave a reduction from MinRMC to 2-SAT in the case of $|\Sigma| = 2$ and $d = 1$, resulting in an $O(n\ell^2)$-time algorithm. Here we will provide a more efficient reduction to 2-SAT, exploiting the compact encoding $C_{\leq 1}$ of the at-most-one constraint (see Section 2.3) in two different ways. Our algorithm solves the more general RCMC problem for arbitrary alphabet size. Moreover, its running time is linear in the input size.

Theorem 3.2. RCMC *can be solved in time* $O(n\ell)$ *when* $\max_{i\in[n]} d_i = 1$.

Proof. We reduce RCMC to 2-SAT. Let $I_d = \{i \in [n] \mid d_i = d\}$ be the row indices for which the distance bound is d for $d \in \{0, 1\}$. We define a variable $x_{j,\sigma}$ for each $j \in [\ell]$ and $\sigma \in \Sigma$. The intuition behind our reduction is that the j-th entry of the solution vector v becomes σ when $x_{j,\sigma}$ is true. We give the construction of a 2-CNF formula ϕ in three parts ϕ_1, ϕ_2, ϕ_3 (that is, $\phi = \phi_1 \wedge \phi_2 \wedge \phi_3$).

- Let $X_j = \{x_{j,\sigma} \mid \sigma \in \Sigma\}$ for each $j \in [\ell]$. The first formula will ensure that at most one character is assigned to each entry of the solution vector v:

$$\phi_1 = \bigwedge_{j\in[\ell]} C_{\leq 1}(X_j).$$

- We define the formula ϕ_2 to handle distance-0 constraints:

$$\phi_2 = \bigwedge_{i\in I_0} \bigwedge_{\substack{j\in[\ell] \\ \mathbf{S}[i,j]\neq *}} (x_{j,\mathbf{S}[i,j]}).$$

- Finally, we define the formula ϕ_3 to guarantee that the solution vector v deviates from each row vector of $\mathbf{S}[I_1]$ on at most one position.

$$\phi_3 = \bigwedge_{i \in I_1} C_{\leq 1}(\{\neg x_{j,\mathbf{S}[i,j]} \mid j \in [\ell], \mathbf{S}[i,j] \neq *\}).$$

Note that our construction uses $O(|\Sigma| \cdot \ell)$ variables and $O((n+|\Sigma|) \cdot \ell)$ clauses. We prove the correctness of the reduction.

($\Rightarrow$) Suppose that there exists a vector $v \in \Sigma^\ell$ such that $\delta(v, \mathbf{S}[i]) \leq d_i$ holds for each $i \in [n]$. For each $j \in [\ell]$ and $\sigma \in \Sigma$, we set $x_{j,\sigma} = \chi[v[j] = \sigma]$. It is easy to see that this truth assignment satisfies ϕ.

($\Leftarrow$) Suppose that there exists a satisfying truth assignment φ. Let J^* denote the column indices $j \in [\ell]$ such that $\varphi(x_{j,\sigma}) = 0$ for all $\sigma \in \Sigma$. Note that at most one variable in X_j is set to true in φ for each $j \in [\ell]$ by Lemma 2.2. It follows that for each $j \in [\ell] \setminus J^*$, there exists exactly one character $\sigma_j \in \Sigma$ satisfying $\varphi(x_{j,\sigma_j}) = 1$ and we assign $v[j] = \sigma_j$. For each $j \in J^*$, we set $v[j] = \sigma^*$ for some arbitrary character $\sigma^* \in \Sigma$. The formula ϕ_2 ensures that $\delta(v, \mathbf{S}[i]) = 0$ holds for each $i \in I_0$. Moreover, ϕ_3 ensures that there is at most one column index $j \in [\ell]$ such that $\mathbf{S}[i,j] \neq *$ and $\mathbf{S}[i,j] \neq v[j]$ for each $i \in I_1$ by Lemma 2.2. □

Note that MinRMC is NP-hard even if $|\Sigma| = 2$ and $d = 2$ [HR15]. Thus, our result implies a complete dichotomy between tractable and intractable cases regarding d.

We remark that this dichotomy also holds for MinLRMC. When $d = 1$, the reduction in the argument for Lemma 3.1 constructs n instances of RCMC with $\max_{i \in [n]} d_i = 1$. Thus, we obtain a polynomial-time algorithm for the case $d = 1$.

Corollary 3.3. MinLRMC *can be solved in time* $O(n^2\ell)$ *when* $d = 1$.

Let us also remark that MinRMC can be solved in linear time when $|\Sigma| = 2$ and $d = \ell - 1$ (the problem remains NP-hard with $d = \ell - 1$ in the case of unbounded alphabet size [LMS18]): First, we remove every row vector with at least one missing entry since it has distance at most $\ell - 1$ from any vector of length ℓ. We then remove every duplicate row vector. This can be achieved in linear time: We sort the row vectors lexicographically by linear-time sorting algorithm such as radix sort and we compare each row vector to the adjacent row vectors in the sorted order. We return **Yes** if and only if there are at most $2^\ell - 1$ row vectors left, because each distinct row vector $u \in \{0,1\}^\ell$ excludes exactly one row vector $\overline{u} \in \{0,1\}^\ell$ where $\overline{u}[j] = 1 - u[j]$ for each $j \in [\ell]$.

3.2 Parameter ℓ

Hermelin and Rozenberg [HR15, Theorem 3] showed that one can solve MinRMC in time $O(2^{\ell^2/2} \cdot n\ell)$ using a search tree algorithm. We use a more refined recursive step to obtain a better running time (see Algorithm 1). In particular we employ a trick that is used by Gramm, Niedermeier, and Rossmanith [GNR03] in order to reduce the search space to $d + 1$ subcases.

Algorithm 1 Algorithm for RCMC

Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d_1, \ldots, d_n \in \mathbb{N}$.
Task: Decide whether there exists a row vector $v \in \Sigma^\ell$ with $d(v, \mathbf{S}[i]) \leq d_i$ for all $i \in [n]$.

1: **if** $d_i < 0$ for some $i \in [n]$ **then return No**.
2: **if** $\ell - |P_*(\mathbf{S}[i])| \leq d_i$ for all $i \in [n]$ **then return Yes**.
▷ $|P_*(\mathbf{S}[i])|$ is the number of missing entries in $\mathbf{S}[i]$
3: Choose any $i \in [n]$ such that $\ell - |P_*(\mathbf{S}[i])| > d_i$.
4: Choose any $R \subseteq [\ell] \setminus P_*(\mathbf{S}[i])$ with $|R| = d_i + 1$.
5: **for all** $j \in R$ **do**
6: Let $\mathbf{S}' = \mathbf{S}[:, [\ell] \setminus \{j\}]$ and $d'_{i'} = d_i - \delta(\mathbf{S}[i, j], \mathbf{S}[i', j])$ for each $i' \in [n]$.
7: **if** recursion on $(\mathbf{S}', d'_1, \ldots, d'_n)$ returns **Yes then return Yes**.
8: **return No**.

Theorem 3.4. RCMC *can be solved in time* $O((d+1)^\ell \cdot n\ell)$ *for* $d = \max_{i \in [n]} d_i$.

Proof. We will prove that Algorithm 1 is correct by induction on ℓ. More specifically, we show that the algorithm returns **Yes** if there exists a vector $v \in \Sigma^\ell$ that satisfies $\delta(\mathbf{S}[i], v) \leq d_i$ for all $i \in [n]$. It is easy to see that the algorithm is correct for the base case $\ell = 0$, because it returns **Yes** if d_i is nonnegative for all $i \in [n]$ and **No** otherwise (Lines 1 and 2). Consider the case $\ell > 0$. The terminal conditions in Lines 1 and 2 are clearly correct. We show that branching on R is correct in Lines 4 and 5. If $v[j] \neq \mathbf{S}[i, j]$ holds for all $j \in R$, then we have a contradiction $\delta(v, \mathbf{S}[i]) \geq |R| > d_i$. Thus the branching on R leads to a correct output. Now the induction hypothesis ensures that the recursion on $\mathbf{S}[:, [\ell] \setminus \{j\}]$ (notice that it has exactly one column less) returns a desired output. This concludes that our algorithm is correct.

Now we will examine the time complexity. Note that each node in the search tree has at most $d+1$ children. Moreover, the depth of the search tree is at most ℓ because the number of columns decreases for each recursion. Since each recursion step only requires linear (that is, $O(n\ell)$) time, the overall running time is $O((d+1)^\ell n\ell)$. □

It is easy to see that any MinRMC instance with $d \geq \ell$ is a **Yes**instance. This observation yields a fixed-parameter algorithm with respect to ℓ.

Corollary 3.5. MinRMC *can be solved in time* $O(n\ell^{\ell+1})$.

We remark that this algorithm cannot be significantly improved assuming the ETH (Conjecture 1). It is known that there is no $\ell^{o(\ell)} \cdot n^{O(1)}$ time algorithm for Closest String unless the ETH fails [Cyg+15; LMS18]. The running time of our algorithm matches this lower bound and therefore there is presumably no faster algorithm (up to constants in the exponent) in terms of the parameter ℓ.

As a consequence of Corollary 3.5, we also obtain a fixed-parameter algorithm for MinLRMC with parameterization k. For each $i \in [n]$, we construct a RCMC instance, where the input matrix is $\mathbf{S}_i = \mathbf{S}[:, P_*(\mathbf{S}[i])]$ and $d_{i,i'} = d - \delta(\mathbf{S}[i], \mathbf{S}[i'])$ for each $i' \in [n]$. We return **Yes** if and only if there is a **Yes** instance $(\mathbf{S}_i, d_{i,1}, \ldots, d_{i,n})$ of RCMC. Each RCMC instance requires $O(n\ell)$ time to construct, and $O(nk^{k+1})$ time to solve, because $\mathbf{S}_i$ contains at most k columns. Thus, we obtain the following corollary.

Algorithm 2 Algorithm for NEIGHBORING STRING by Gramm, Niedermeier, and Rossmanith [GNR03]

Input: A matrix $\mathbf{T} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d_1, \ldots, d_n \in \mathbb{N}$.
Task: Decide whether there exists a row vector $v \in \Sigma^\ell$ with $d(v, \mathbf{T}[i]) \leq d_i$ for all $i \in [n]$.

1: **if** $d_i < 0$ for some $i \in [n]$ **then return No**.
2: **if** $\delta(\mathbf{T}[1], \mathbf{T}[i]) \leq d_i$ for all $i \in [n]$ **then return Yes**.
3: Choose any $i \in [n]$ such that $\delta(\mathbf{T}[1], \mathbf{T}[i]) > d_i$.
4: Choose any $Q' \subseteq Q(\mathbf{T}[1], \mathbf{T}[i])$ with $|Q'| = d_i + 1$.
5: **for all** $j \in Q'$ **do**
6: Let $\mathbf{T}' = \mathbf{T}[:, [\ell] \setminus \{j\}]$ and $d'_{i'} = d_i - \delta(\mathbf{T}[i, j], \mathbf{T}[i', j])$ for each $i' \in [n]$.
7: **if** recursion on $(\mathbf{T}', d'_1, \ldots, d'_n)$ returns **Yes then return Yes**.
8: **return No**.

Corollary 3.6. MINLRMC *can be solved in time* $O(n^2\ell + n^2 k^{k+1})$.

3.3 Parameter $d + k$

The special case of MINRMC in which the input matrix is complete is known as the CLOSEST STRING problem. It can be formulated using the matrix notation as follows.

CLOSEST STRING
Input: A matrix $\mathbf{T} \in \Sigma^{n \times \ell}$ and $d \in \mathbb{N}$.
Question: Is there a row vector $v \in \Sigma^\ell$ such that $\delta(v, \mathbf{T}) \leq d$?

The CLOSEST STRING problem is NP-hard even for binary alphabet [FL97] but several fixed-parameter algorithms are known. In this section we will generalize two algorithms given by Gramm, Niedermeier, and Rossmanith [GNR03] and Ma and Sun [MS09] to MINRMC. We will describe both algorithms briefly. In fact both algorithms solve the special case of RCMC, referred to as NEIGHBORING STRING, where the input matrix is complete.

NEIGHBORING STRING
Input: A matrix $\mathbf{T} \in \Sigma^{n \times \ell}$ and $d_1, \ldots, d_n \in \mathbb{N}$
Question: Is there a row vector $v \in \Sigma^\ell$ such that $\delta(v, \mathbf{T}[i]) \leq d_i$ for each $i \in [n]$?

The algorithm of Gramm, Niedermeier, and Rossmanith [GNR03] is given in Algorithm 2. First it determines whether the first row vector $\mathbf{T}[1]$ suffices as a solution. Once it fails, it finds another row vector $\mathbf{T}[i]$ that differs from $\mathbf{T}[1]$ on more than d_i positions. Then it branches on the column positions $Q(\mathbf{T}[1], \mathbf{T}[i])$ where $\mathbf{T}[1]$ and $\mathbf{T}[i]$ disagree.

Using a search variant of Algorithm 2, Hermelin and Rozenberg [HR15, Theorem 4] claimed that MINRMC is fixed-parameter tractable with respect to $d + k$. Here we reveal that their fixed-parameter algorithm is incorrect on some inputs. Their algorithm chooses an arbitrary row vector $\mathbf{S}[i]$ and calls the algorithm by Gramm, Niedermeier, and Rossmanith [GNR03] with input matrix $\mathbf{S}' = \mathbf{S}[:, [\ell] \setminus \{P_*(\mathbf{S}[i])\}]$. This results in a set of row vectors v satisfying $\delta(v, \mathbf{S}') \leq d$. Then the algorithm constructs an

instance of RCMC where the input matrix is $\mathbf{S}[P_*(\mathbf{S}[i])]$ and the distance bound is given by $d_{i'} = d - \delta(v, \mathbf{S}'[i'])$ for each $i' \in [n]$. The correctness is based on the erroneous assumption that the algorithm of Gramm, Niedermeier, and Rossmanith [GNR03] finds *all* row vectors v satisfying $\delta(v, \mathbf{S}') \leq d$ in time $O((d+1)^d \cdot n\ell)$. Although Gramm, Niedermeier, and Rossmanith [GNR03] noted that this is indeed the case when d is optimal, it does not always hold true. In fact, it is generally impossible to enumerate all solutions in time $O((d+1)^d \cdot n\ell)$ because there may be $\Omega(\ell^d)$ solution vectors. We use the following simple matrix to illustrate the error in the algorithm of Hermelin and Rozenberg [HR15]:

$$\mathbf{S} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ * & 0 & 0 \end{bmatrix}.$$

We show that the algorithm may output an incorrect answer for $d = 2$. If the algorithm chooses $i = 3$, then the algorithm by Gramm, Niedermeier, and Rossmanith [GNR03] returns only one row vector 00. Then the algorithm of Hermelin and Rozenberg [HR15] constructs an instance of RCMC with $\mathbf{S}' = \begin{bmatrix} 0 & 1 \end{bmatrix}^T$ and $d_1 = d_2 = 0$, resulting in **No**. However, the row vector $v = 001$ satisfies $\delta(v, \mathbf{S}) = 2$ and thus the correct output is **Yes**. To remedy this, we give a fixed-parameter algorithm for MinRMC, adapting the algorithm by Gramm, Niedermeier, and Rossmanith [GNR03].

Before presenting our algorithm, let us give an observation noted by Basavaraju et al. [Bas+18] and Gramm, Niedermeier, and Rossmanith [GNR03] for the case of no missing entries. Suppose that the input matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ contains more than nd dirty columns (recall that a column is said to be dirty if it contains at least two distinct symbols from the alphabet). Clearly we can assume that every column is dirty. For any vector $v \in \Sigma^\ell$, there exists $i \in [n]$ with $\delta(v, \mathbf{S}[i]) \geq d$ by the pigeon hole principle and hence we can immediately conclude that it is a **No** instance. It is easy to see that this argument also holds for MinRMC.

Lemma 3.7. *[GNR03, Lemma 4] Let $(\mathbf{S}, d)$ be a* MinRMC *instance, where $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$. If $\mathbf{S}$ contains more than nd dirty columns, then there is no row vector $v \in \Sigma^\ell$ with $\delta(v, \mathbf{S}) \leq d$.*

Our first algorithm is given in Algorithm 3. It generalizes Algorithm 2 and finds the solution vector even if the input matrix is incomplete. In contrast to Neighboring String, the output cannot be immediately determined even if $d_1 = 0$. We use Algorithm 1 to overcome this issue (Line 3). Algorithm 3 also considers the columns where the first row vector has missing entries (recall that $P_*(\mathbf{S}[1])$ denotes column indices j with $\mathbf{S}[1, j] = *$) in the branching step (Line 8), not only the columns where $\mathbf{S}[1]$ and $\mathbf{S}[i]$ disagree. Again we apply the trick of restricting to $d_i + 1$ subcases (Line 7). This reduces the size of the search tree significantly. We show the correctness of Algorithm 3 and analyze its running time in the proof of the following theorem.

Theorem 3.8. MinRMC *can be solved in time $O(n\ell + (d+1)^{d+k+1} n)$.*

Proof. First, we prove that our algorithm is correct by induction on $d_1 + |P_*(\mathbf{S}[1])|$. More specifically, we show that the algorithm returns **Yes** if and only if a vector $v \in \Sigma^\ell$ satisfying $\delta(\mathbf{S}[i], v) \leq d_i$ for all $i \in [n]$ exists.

Algorithm 3 Algorithm for RCMC

Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d_1, \ldots, d_n \in \mathbb{N}$.
Task: Decide whether there exists a row vector $v \in \Sigma^\ell$ with $d(v, \mathbf{S}[i]) \leq d_i$ for all $i \in [n]$.

1: **if** $d_1 = 0$ **then**
2: Let $\mathbf{S}' = \mathbf{S}[[2, n], P_*(\mathbf{S}[1])]$ and $d'_i = d_i - \delta(\mathbf{S}[1], \mathbf{S}[i])$ for each $i \in [2, n]$.
3: **return** the output of Algorithm 1 on $(\mathbf{S}', d'_2, \ldots, d'_n)$.
4: Let $R_i = (P_*(\mathbf{S}[1]) \setminus P_*(\mathbf{S}[i])) \cup Q(\mathbf{S}[1], \mathbf{S}[i])$ for each $i \in [2, n]$.
5: **if** $|R_i| \leq d_i$ for all $i \in [2, n]$ **then return Yes**.
6: Choose any $i \in [n]$ with $|R_i| > d_i$.
7: Choose any $R \subseteq R_i$ with $|R| = d_i + 1$.
8: **for all** $j \in R$ **do**
9: Let $\mathbf{S}' = \mathbf{S}[:, [\ell] \setminus \{j\}]$ and $d'_{i'} = d_i - \delta(\mathbf{S}[i, j], \mathbf{S}[i', j])$ for each $i' \in [n]$.
10: **if** recursion on $(\mathbf{S}', d'_1, \ldots, d'_n)$ returns **Yes then return Yes**.
11: **return No**.

0		0	1		1			1	0
		0			1	1	1	0	1

R_2

Figure 3.2: An illustration of R_i on 2×10 matrix. Observe that the distance between the two row vectors is zero, when restricted to columns not in R_i.

Consider the base case $d_1 + |P_*(\mathbf{S}[1])| = 0$. Since $d_1 = 0$, the algorithm terminates in Line 3. When $d_1 = 0$, any solution vector must agree with $\mathbf{S}[1]$ on each entry unless the entry is missing in $\mathbf{S}[1]$. Hence, the output in Line 3 is correct by Theorem 3.4. Consider the case $d_1 + |P_*(\mathbf{S}[1])| > 0$. Let $R_i = (P_*(\mathbf{S}[1]) \setminus P_*(\mathbf{S}[i])) \cup Q(\mathbf{S}[1], \mathbf{S}[i])$ for each $i \in [2, n]$ (see Figure 3.2 for an illustration). If $|R_i| \leq d_i$ holds for all $i \in [2, n]$, then the vector $\mathbf{S}[1] \oplus \sigma^{|P_*(\mathbf{S}[1])|}$ (the vector obtained by filling each missing entry in $\mathbf{S}[1]$ with σ) is a solution for an arbitrary character $\sigma \in \Sigma$. Hence Line 5 is correct. Suppose that there exists a solution vector $v \in \Sigma^\ell$ with $\delta(v, \mathbf{S}[i]) \leq d_i$ for all $i \in [n]$. We show that the branching in Line 8 is correct. Let R be as specified in Line 7. We claim that there exists $j \in R$ with $v[j] = \mathbf{S}[i, j]$ for every choice of R. Otherwise, $v[j] \neq \mathbf{S}[i, j]$ and $\mathbf{S}[i, j] \neq *$ holds for all $j \in R$ and we have $\delta(v, \mathbf{S}[i]) > d_i$ (a contradiction). Note that $\mathbf{S}[:, [\ell] \setminus \{j\}]$ has exactly one less missing entry if $j \in P_*(\mathbf{S}[1])$ and that $d'_1 = d_1 - 1$ in case of $j \in Q(\mathbf{S}[1], \mathbf{S}[j])$. It follows that $d_1 + |P_*(\mathbf{S}[1])|$ is strictly smaller in the recursive call (Line 10). Hence, the induction hypothesis ensures that the algorithm returns **Yes** when $v[j] = \mathbf{S}[i, j]$ holds. On the other hand, it is not hard to see that the algorithm returns **No** if there is no solution vector. This concludes the correctness proof of Algorithm 3.

We examine the time complexity. Assume without loss of generality that $k = |P_*(\mathbf{S}[1])|$ and $d = d_1$ hold initially. Consider the search tree where each node corre-

Algorithm 4 Algorithm for NEIGHBORING STRING by Ma and Sun [MS09]

Input: A matrix $\mathbf{T} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d_1, \ldots, d_n \in \mathbb{N}$.
Task: Decide whether there exists a row vector $v \in \Sigma^\ell$ with $d(v, \mathbf{T}[i]) \leq d_i$ for all $i \in [n]$.
1: **if** $\delta(\mathbf{T}[1], \mathbf{T}[i]) \leq d_i$ for all $i \in [n]$ **then return Yes** .
2: Choose any $i \in [n]$ such that $\delta(\mathbf{T}[1], \mathbf{T}[i]) > d_i$.
3: Let $Q = Q(\mathbf{T}[1], \mathbf{T}[i])$.
4: **for all** $v \in \Sigma^{|Q|}$ such that $\delta(v, \mathbf{T}[1]) \leq d_1$ and $\delta(v, \mathbf{T}[i]) \leq d_i$ **do**
5: Let $\mathbf{T}' = \mathbf{T}[:, [\ell] \setminus Q]$ and $d'_1 = \min\{d_1 - \delta(v, \mathbf{T}[1, Q]), \lceil d_1/2 \rceil - 1\}$.
6: Let $d'_{i'} = d_{i'} - d(v, \mathbf{T}[i', Q])$ for each $i' \in [2, n]$.
7: **if** recursion on $(\mathbf{T}, d'_1, \ldots, d'_n)$ returns **Yes then return Yes**.
8: **return No**.

sponds to a call on either Algorithm 1 or Algorithm 3. If $d_1 > 0$, then $d_1 + P_*(\mathbf{S}[1])$ decreases by 1 in each recursion and there are at most $d + 1$ recursive calls. Let u be some node in the search tree that invokes Algorithm 1 for the first time. We have seen in the proof of Theorem 3.4 that the subtree rooted at u is a tree of depth at most $|P_*(\mathbf{S}[1])|$, in which each node has at most $d_2 - \delta(\mathbf{S}[1], \mathbf{S}[2]) + 1 \leq d + 1$ children. Note also that u lies at depth $d + k - |P_*(\mathbf{S}[1])|$. Thus the depth of the search tree is at most $d + k$ and the search tree has size $O((d + 1)^{d+k})$. We can assume that $\ell \leq nd$ by Lemma 3.7 and hence each node requires $O(nd)$ time. This results in the overall running time $O(n\ell + (d + 1)^{d+k+1}n)$. □

Now we will provide a more efficient fixed-parameter algorithm when the alphabet size is small using Algorithm 4 of Ma and Sun [MS09]. Whereas Algorithm 2 considers each position of (a subset of) $Q(\mathbf{T}[1], \mathbf{T}[i])$ one by one, Algorithm 4 considers all vectors on $Q(\mathbf{T}[1], \mathbf{T}[i])$ in a single recursion. The following lemma justifies why d_1 can be halved (Line 5) in each iteration (the vectors u and w correspond to $\mathbf{T}[1]$ and $\mathbf{T}[i]$, respectively).

Lemma 3.9. *[MS09, Lemma 3.1] Let $u, v, w \in \Sigma^\ell$ be row vectors satisfying $\delta(u, w) > \delta(v, w)$. Then, it holds that $\delta(u[Q'], v[Q']) < \delta(u, v)/2$ for $Q' = [\ell] \setminus Q(u, w)$.*

Proof. Assume that $\delta(u[Q'], v[Q']) \geq \delta(u, v)/2$. We can rewrite the value of $\delta(u, v) + \delta(v, w)$ as follows:

$$\delta(u, v) + \delta(v, w) = \delta(u[Q'], v[Q']) + \delta(v[Q'], w[Q']) + \delta(u[Q], v[Q]) + \delta(v[Q], w[Q]),$$

where Q is a shorthand for $Q = Q(v, w)$. It follows from the definition of Q' that $u[Q'] = w[Q']$ and hence

$$\delta(u[Q'], v[Q']) = \delta(v[Q'], w[Q']). \tag{3.1}$$

We also note that $\delta(u[Q] + v[Q]) + \delta(v[Q] + w[Q]) \geq |Q| = \delta(v, w)$ because it must hold that $u[j] \neq v[j]$ or $v[j] \neq w[j]$ for each $j \in Q$. Now we have the following contradiction:

$$\delta(u, v) + \delta(v, w) \geq 2\delta(u[Q'], v[Q']) + \delta(u, w) > \delta(u, v) + \delta(v, w).$$

This concludes the proof. □

Lemma 3.9 plays a crucial role in obtaining the running time $O(n\ell + (16|\Sigma|)^d nd)$ of Ma and Sun [MS09]. However, Lemma 3.9 may not hold in the presence of missing entries (in fact, Equation (3.1) may break when at least one of u or w contains missing entries). For instance, $u = 0^\ell, v = 1^\ell, w = *^\ell$ is one counterexample. Here notice that $Q(u, w) = \emptyset$ and that $\delta(u[Q'], v[Q']) = \delta(u, v) = \ell$.

To work around this issue, let us introduce a new variant of Closest String. The problem is somewhat artificial and presumably it has no practical use. However, the problem will be useful to derive a fixed-parameter algorithm for MinRMC (Theorem 3.11). We will use a special character "$\diamond$" to denote a "dummy" character.

Neighboring String with Dummies (NSD)
Input: A matrix $\mathbf{T} \in (\Sigma \cup \{\diamond\})^{n \times \ell}$ and $d_1, \ldots, d_n \in \mathbb{N}$.
Question: Is there a row vector $v \in \Sigma^\ell$ such that $\delta(v, \mathbf{T}[i]) \leq d_i$ for each $i \in [n]$?

Note that the definition of NSD forbids dummy characters in the solution vector v. Observe that Lemma 3.9 (in particular Equation (3.1)) holds even if row vectors u, w may contain dummy characters. Now we show that NSD can be solved using Algorithm 4 as a subroutine.

Lemma 3.10. NSD *can be solved in time* $O(|\Sigma|^k \cdot n\ell + 2^{4d-3k} \cdot |\Sigma|^d \cdot nd))$, *where* k *is the minimum number of dummy characters in any row vector of* $\mathbf{T}$.

Proof. With Lemma 3.9, one can prove that Algorithm 4 solves the NSD problem if the first row vector $\mathbf{T}[1]$ contains no dummy characters by induction on d_1. Refer to [MS09, Theorem 3.2] for details. We describe how we use Algorithm 4 of Ma and Sun [MS09] to solve NSD. Let $I = (\mathbf{T}, d_1, \ldots, d_n)$ be an instance of NSD. We assume that $|P_\diamond(\mathbf{T}[1])| = k$. For each row vector u of Σ^k, we invoke Algorithm 4 with the input matrix $\mathbf{T}' = \mathbf{T}[[\ell] \setminus P_\diamond(\mathbf{T}[1])]$ and the distance bounds $d - k, d - \delta(u, \mathbf{T}[2, P_\diamond(\mathbf{T}[1])]), \ldots, d - \delta(u, \mathbf{T}[n, P_\diamond(\mathbf{T}[1])])$. Note that $\mathbf{T}'[1]$ contains no dummy character and thus the output of Algorithm 4 is correct. We return **Yes** if and only if Algorithm 4 returns **Yes** at least once. Let us prove that this solves NSD. If I is a **Yes** instance with solution vector $v \in \Sigma^\ell$, then it is easy to verify that Algorithm 4 returns **Yes** when $u = v[P_\diamond(\mathbf{T}[1])]$. On the other hand, the distance bounds in the above procedure ensure that I is a **Yes** instance if Algorithm 4 returns **Yes**.

Now we show that this procedure runs in the claimed time. Ma and Sun [MS09] proved that Algorithm 4 runs in time

$$O\left(n\ell + \binom{d_{\max} + d_{\min}}{d_{\min}} \cdot (4|\Sigma|)^{d_{\min}} \cdot n d_{\max}\right),$$

where $d_{\max} = \max_{i \in [n]} d_i$ and $d_{\min} = \min_{i \in [n]} d_i$. Note that we have $d_{\max} \leq d$ and $d_{\min} \leq d - k$ for each call of Algorithm 4. Hence it remains to show that $\binom{2d-k}{d} \in O(2^{2d-k})$. Using Stirling's approximation $\sqrt{2\pi} n^{n+1/2} e^{-n} \leq n! \leq e n^{n+1/2} e^{-n}$ that holds for all positive integers n, we obtain

$$\binom{2d-k}{d} = \frac{(2d-k)!}{d! \cdot (d-k)!} \leq c \cdot \frac{(2d-k)^{2d-k}}{d^d \cdot (d-k)^{d-k}}$$

for some constant c. We claim that the last term is upper-bounded by $c \cdot 2^{2d-k}$. We use the fact that the function $x \mapsto x \log x$ is convex over its domain $x > 0$ (note that the second derivative is given by $x \mapsto 1/x$). Since a convex function $f\colon D \to \mathbb{R}$ satisfies $(f(x) + f(y))/2 \geq f((x+y)/2)$ for any $x, y \in D$, we obtain

$$d \log d + (d-k)\log(d-k) \geq 2\left(\frac{2d-k}{2}\right) \cdot \log\left(\frac{2d-k}{2}\right).$$

It follows that $d^d \cdot (d-k)^{d-k} = 2^{d\log d + (d-k)\log(d-k)} \geq 2^{(2d-k)\log(d-k/2)} = (d-k/2)^{2d-k}$. This shows that $\binom{2d-k}{d} \in O(2^{2d-k})$. □

Finally we provide a polynomial-time reduction from MinRMC to NSD.

Theorem 3.11. MinRMC *can be solved in time* $O(n\ell + 2^{4d+k} \cdot |\Sigma|^{d+k} \cdot n(d+k))$.

Proof. Let $I = (\mathbf{S}, d)$ be an instance of MinRMC. We construct an instance $I' = (\mathbf{T}, d+k)$ of NSD where $\mathbf{T} \in (\Sigma \cup \{\diamond\})^{n \times (\ell+k)}$ and each row vector of $\mathbf{T}$ contains exactly k dummy characters. Note that such a construction yields an algorithm for MinRMC running in time $O(n(\ell+k) + |\Sigma|^k \cdot nd + 2^{4d+k} \cdot |\Sigma|^{d+k} \cdot n(d+k)) = O(n\ell + 2^{4d+k} \cdot |\Sigma|^{d+k} \cdot n(d+k))$ using Lemma 3.7. Let $\sigma \in \Sigma$ be an arbitrary character. We define the row vector $\mathbf{T}[i]$ for each $i \in [n]$ as follows: Let $\mathbf{T}[i, [\ell]] = \mathbf{S}[i] \oplus \diamond^\ell$ (in other words, the row vector $\mathbf{T}[i, [\ell]]$ is obtained from $\mathbf{S}[i]$ by replacing $*$ by $\diamond$) for the leading ℓ entries. For the remainder, let

$$\mathbf{T}[i, \ell + j] = \begin{cases} \sigma & \text{if } j \leq |P_*(\mathbf{S}[i])|, \\ \diamond & \text{otherwise,} \end{cases}$$

for each $j \in [k]$. See Figure 3.3 for an illustration. We claim that I is a **Yes**instance if and only if I' is a yesinstance.

($\Rightarrow$) Let $v \in \Sigma^\ell$ be a solution of I. We claim that the vector $v' \in \Sigma^{\ell+k}$ such that $v'[[\ell]] = v$ and $v[[\ell+1, \ell+k]] = \sigma^k$ is a solution of I'. For each $i \in [n]$, we have

$$\delta(v', \mathbf{T}[i]) = \delta(v'[[\ell]], \mathbf{T}[i, [\ell]]) + \delta(\sigma^k, \mathbf{T}[i, [\ell+1, \ell+k]]).$$

It is easy to see that the first term is at most $d + |P_*(\mathbf{S}[i])|$ and that the second term equals $k - |P_*(\mathbf{S}[i])|$. Thus we have $\delta(v', \mathbf{T}) \leq d + k$.

($\Leftarrow$) Let $v' \in \Sigma^\ell$ be a solution of I'. Since the row vector $\mathbf{T}[i, [\ell+1, \ell+k]]$ contains $k - |P_*(\mathbf{S})|$ dummy characters, we have $\delta(v'[[\ell]], \mathbf{T}[i, [\ell]]) \leq (d+k) - (k - |P_*(\mathbf{T}[i])|) = d + |P_*(\mathbf{T}[i])|$ for each $i \in [n]$. It follows that $\delta(v'[[\ell]], \mathbf{S}[i]) \leq d$ holds for each $i \in [n]$. □

3.4 Concluding remarks

We investigated the computational complexity of MinRMC, also known as Closest String With Wildcards. We provided a linear-time algorithm for the case of the distance bound $d = 1$, answering an open question by Hermelin and Rozenberg [HR15] whether MinRMC with $d = 1$ can be solved in polynomial time for general alphabet size. The crucial idea was to use the efficient encoding $C_{\leq 1}$ of the at-most-one constraint. We

$$\mathbf{S} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & * \\ * & * & 2 \end{bmatrix} \qquad \mathbf{T} = \begin{bmatrix} 0 & 0 & 0 & \diamond & \diamond \\ 1 & 1 & \diamond & \sigma & \diamond \\ \diamond & \diamond & 2 & \sigma & \sigma \end{bmatrix}$$

Figure 3.3: An illustration of the reduction in Theorem 3.11. Given the matrix $\mathbf{S}$ with $k = 2$ (left), our reduction constructs the matrix $\mathbf{T}$ with k additional columns (right). Note that every row vector in $\mathbf{T}$ contains exactly two dummy characters. The MinRMC instance $(\mathbf{S}, d = 1)$ is a **Yes** instance with a solution vector $v = 100$. The corresponding NSD instance $(\mathbf{T}, d + k = 3)$ is also a **Yes** instance with a solution vector $v' = 100\sigma\sigma$.

then proceeded to study the parameterized complexity of MinRMC. We revealed that the fixed-parameter algorithm proposed by Hermelin and Rozenberg [HR15] with parameterization $d + k$ is flawed, and we presented an alternative fixed-parameter algorithm (Algorithm 3) with running time $O(n\ell + (d+1)^{d+k+1}n)$. Our algorithm is a nontrivial combination of two fixed-parameter algorithms: our fixed-parameter algorithm with parameterization ℓ (Algorithm 1) and the algorithm by Gramm, Niedermeier, and Rossmanith [GNR03] for Neighboring String (Algorithm 2), a variant of MinRMC in which the input matrix is complete and the distance bound can be specified for each row vector. We also provided an algorithm that is more efficient for small alphabet size. We basically eliminated missing entries, by introducing a new variant of Neighboring String, where the input matrix may contain "dummy" characters. We were then able to take advantage of the algorithm by Ma and Sun [MS09] for Neighboring String. This is contrary to the supposition of Hermelin and Rozenberg [HR15], who hinted that techniques for Neighboring String probably cannot be applied in the presence of missing entries.

We conclude with some directions for future research:

- Can the running time of Theorem 3.11 be improved? Since Ma and Sun [MS09] proved that Closest String can be solved in $O((16|\Sigma|)^d \cdot n\ell)$ time, a plethora of efforts have been made to reduce the base in the exponential dependence in the running time [CMW12; CMW16; CW11; NS12]. A natural question is whether these results can be translated to MinRMC as well.

- Another direction would be to consider a generalization of MinRMC with *outliers*. Its input is the same as MinRMC, except that it has an additional parameter t. The task is to determine whether there are a set $I \subseteq [n]$ of row indices and a vector $v \in \Sigma^\ell$ such that $|I| \leq k$ and $\delta(v, \mathbf{S}[[n] \setminus I]) \leq d$. The case in which the input matrix is known as Closest String with Outliers [BM11; BS18]. Boucher and Ma [BM11] proved fixed-parameter tractability with respect to $d + t$. We also remark that the results of Eiben et al. [Eib+19] implies fixed parameter tractability with respect to $d + n - t$. Hence, it is interesting to study the outlier variant of MinRMC is fixed-parameter tractable with respect to $d + k + t$.

- Finally, let us mention yet another variant somewhat related to MinRMC:

MaxRMC
Input: An incomplete matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$.
Question: Is there a completion $\mathbf{T}$ of $\mathbf{S}$ and a row vector $v \in \Sigma^{\ell}$ such that $\delta(v, \mathbf{T}[i]) \geq d$ for each $i \in [n]$?

Note that the input is identical to that of MinRMC but the inequality in the radius constraint is reversed. The special case in which the input matrix is complete is referred to as Farthest String [WZ09], and fixed-tractability with respect to $|\Sigma|+d$ is known [GGN06; WZ09]. Intuitively, every missing entry can be treated as a "dummy" symbol in MaxRMC, because each missing entry should be completed by a symbol different from the one in the solution vector v in the corresponding position. We believe that the algorithm by Gramm, Niedermeier, and Rossmanith [GNR03] (Algorithm 2) can be adapted to solve Neighboring String even in the presence of dummy characters. This would imply that Neighboring String with Dummies is fixed-parameter tractable with respect to d. Is MaxRMC also fixed-parameter tractable with respect to $d(+|\Sigma|)$?

Chapter 4

Diameter Minimization

In this chapter, we study the matrix completion problem where the objective is to minimize the diameter. Here the diameter refers to the maximum pairwise Hamming distance of row vectors. Formally, it is defined as follows:

MinDMC (Minimum Diameter Matrix Completion)
Input: A matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d \in \mathbb{N}$.
Question: Is there a completion $\mathbf{T} \in \Sigma^{n \times \ell}$ of $\mathbf{S}$ such that $\delta(\mathbf{T}) \leq d$?

Note that $\delta(\mathbf{T})$ denotes the diameter of $\mathbf{T}$: $\delta(\mathbf{T}) = \max_{i,i' \in [n]} \delta(\mathbf{T}[i], \mathbf{T}[i'])$. Table 4.1 summarizes our results.

In Section 4.1, we show fixed-parameter tractability with respect to the number n of rows. In particular, we formulate MinDMC in terms of ILP (Integer Linear Programming) with $n^{O(n)}$ variables. Using the well-known algorithm by Lenstra Jr [LJ83] for ILP, we obtain a fixed-parameter algorithm (Theorem 4.2).

In Section 4.2, we investigate the computational complexity of MinDMC in terms of the diameter d of the solution matrix. We obtain a complete dichotomy for binary matrices ($|\Sigma| = 2$): We prove that MinDMC is polynomial-time solvable when $d \leq 3$ (Lemma 4.3 and Thms 4.12 and 4.19) but NP-hard when $d \geq 4$ (Theorem 4.20). For the tractable cases of $d \leq 3$, we obtain polynomial-time algorithms by making use of some combinatorial notions of set systems called Δ-*systems* (also known as *sunflowers*). For the case $d = 2$, we show that MinDMC can be solved in polynomial time even if the alphabet size is arbitrary large (Theorem 4.14). However, we leave the question open whether MinDMC with $d = 3$ can be solved in polynomial time for alphabet size greater than 2.

Finally in Section 4.3, we look into the maximum number k of missing entries over all row vectors. We present a dichotomy result regarding k: MinDMC can be solved in polynomial time when $k = 1$ (Theorem 4.21); it is NP-hard when $k = 2$, even if $|\Sigma| = 2$ (Theorem 4.22). Interestingly, our polynomial-time algorithm for the case $k = 1$ is based on a reduction to 2-SAT and the NP-hardness proof for the case $k = 2$ is due to a reduction from 3-SAT.

Before delving into our main results of this chapter, let us remark that there is a simple linear-time algorithm when $|\Sigma| + \ell$ is a constant. Note that fixed-parameter tractability with respect to $|\Sigma| + \ell$ follows from the kernelization algorithm by Eiben

Table 4.1: Overview of our results for MinDMC. Here we use the following notation: n—number of rows, ℓ—number of columns, $|\Sigma|$—alphabet size, d—distance bound, k—maximum number of missing entries in any row vector.

Parameter	Result	Reference
n	$O(n\ell + 2^{2^{O(n \log n)}})$-time solvable	Theorem 4.2
$d = 1$	$O(n\ell)$-time solvable	Lemma 4.3
$d = 2$	$O(n\ell)$-time solvable when $\|\Sigma\|$ is a constant	Theorem 4.12
	$O(\|\Sigma\|^6 \cdot n^3 + n\ell)$-time solvable	Theorem 4.14
$d = 3$	$O(n\ell^4)$-time solvable when $\|\Sigma\| = 2$	Theorem 4.19
$d \geq 4$	NP-hard even for $\|\Sigma\| = 2$	Theorem 4.20
$k = 1$	$O(n^2\ell)$-time solvable	Theorem 4.21
$k = 2$	NP-hard even for $\|\Sigma\| = 2$	Theorem 4.22

et al. [Eib+19] (for a smaller parameter in fact) but brute-forcing on the kernel requires super-linear time. Our algorithm first guesses a subset $\mathcal{V} \subseteq \Sigma^\ell$ of row vectors (note that there are $2^{|\Sigma|^\ell}$ possibilities). We then spend $|\mathcal{V}|^2 \cdot \ell$ time to confirm that $\delta(\mathcal{V}) \leq d$. If $\delta(\mathcal{V}) \leq d$, then for each $v \in \mathcal{V}$ and each $i \in [n]$, we compute in time $O(\ell)$ whether v is a completion of $\mathbf{S}[i]$. We return **Yes** if and only if there is a completion of $\mathbf{S}[i]$ in $\mathcal{V}$ for all $i \in [n]$. This gives a fixed-parameter algorithm with the following running time:

Lemma 4.1. MinDMC *can be solved in time* $O(2^{|\Sigma|^\ell} \cdot (|\Sigma|^{2\ell} \cdot \ell + |\Sigma|^\ell \cdot n\ell))$.

We leave the question open whether the parameterized complexity of MinDMC with respect to the parameter ℓ alone. It is even unclear whether MinDMC admits an XP algorithm (an algorithm with running time $n^{O(\ell)}$).

4.1 Parameter n

In this section we show that MinDMC is fixed-parameter tractable with respect to the number n of rows. Our result complements the result of Eiben et al. [Eib+19, Theorem 34], from which fixed-parameter tractability with respect to n follows for the case $|\Sigma| = 2$. Similar to their approach, we present an integer linear programming (ILP) formulation of MinDMC using $n^{O(n)}$ variables. This yields a fixed-parameter algorithm for MinDMC because ILP is fixed-parameter tractable with respect to the number of variables (Lemma 2.1). First let us define the notion of *column type* [GNR03]. We say that two column vectors $u, v \in \Sigma^\ell$ are *isomorphic* if there exists a bijective mapping $\pi\colon \Sigma \cup \{*\} \to \Sigma \cup \{*\}$ such that $\pi(*) = *$ and $v[j] = \pi(u[j])$ for each $j \in [\ell]$. Each set of isomorphic columns defines a column type. For instance, two column vectors $\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$ and $\begin{bmatrix}1 & 1 & 0\end{bmatrix}^T$ are of the same column type. It is known that the number of column types equals the Bell number $B(n) \leq n!$ when the input matrix is complete. It follows that there are $\binom{n}{i}B(n-i)$ column types with exactly i missing entries and the total

number of column types is given by

$$\sum_{i=0}^{n} \binom{n}{i} B(n-i) \leq \sum_{i=0}^{n} \frac{n!}{i! \cdot (n-i)!}(n-i)! \leq \sum_{i=0}^{n} \frac{n!}{i!} \leq en!.$$

Let T denote the set of all column types and let T_0 be the set of column types without missing entry. Let $\varphi\colon T_0 \times [n] \times [n] \to \{0,1\}$ be a function such that $\varphi(t_0, i, i') = 1$ if and only if the i-th and i'-th entries in a column vector of type t_0 have different symbols for each $t_0 \in T_0$ and $i, i' \in [n]$. We say that a column type $t \in T$ is *consistent* with a column type $t_0 \in T_0$ if one can obtain a column vector of type t_0 by filling missing entries in a column vector of type t. Let $c(t)$ denote the set of all consistent column types for each $t \in T$. Note that $c(t_0) = \{t_0\}$ if $t_0 \in T_0$. We define a nonnegative variable x_{t,t_0} for each $t \in T$ and $t_0 \in c(t)$. The variable x_{t,t_0} represents the number of columns that have type t in the input matrix and type t_0 in the solution matrix. For each $t \in T$, let n_t be the number of columns of type t in the input matrix. We require that the following constraints are fulfilled:

- The diameter of the solution matrix is at most d:

$$\sum_{t \in T} \sum_{t_0 \in c(t)} \varphi(t_0, i, i') \cdot x_{t,t_0} \leq d$$

  for each $i < i' \in [n]$.

- Every column must be completed into a column of some type in T_0:

$$\sum_{t_0 \in c(t)} x_{t,t_0} = n_t$$

  for each $t \in T$.

It is easy to see that the ILP formulation is equivalent to MinDMC. Note that we use $O(n^{2n})$ variables. Thus we obtain the following theorem with Lemma 2.1.

Theorem 4.2. MinDMC *can be solved in time* $O^*(2^{2^{O(n \log n)}})$.

4.2 Parameter d

In this section, we investigate the computational complexity of MinDMC with respect to the diameter d of the solution matrix. MinDMC can be trivially solved when $d = 1$: any MinDMC instance is a **Yes** instance if and only if there is at most one dirty column in the input matrix. Recall that a column vector is dirty if it contains at least two distinct characters from Σ.

Lemma 4.3. *Let* $I = (\mathbf{S}, d)$ *be a* MinDMC *instance, where* $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ *and* $d = 1$. *Then,* I *is a* ***Yes*** *instance if and only if* $\mathbf{S}$ *contains at most one dirty column.*

Proof. Suppose that $\mathbf{S}$ contains two dirty columns $\mathbf{T}[j], \mathbf{T}[j']$ for $j < j' \in [\ell]$. We claim that I is a **No** instance. Let $i, i' \in [n]$ be such that $\mathbf{S}[i,j] \neq *$, $\mathbf{S}[i',j] \neq *$, and $\mathbf{S}[i,j] \neq \mathbf{S}[i',j]$. Suppose that there exists a completion $\mathbf{T} \in \Sigma^{n\times\ell}$ of $\mathbf{S}$ with $\delta(\mathbf{T}) \leq 1$. Since the column vector $\mathbf{T}[:,j']$ is dirty, there exists a row index $i'' \in [n]$ with $\mathbf{T}[i'',j'] \neq \mathbf{T}[i,j']$. Note that $\mathbf{T}[i,j'] = \mathbf{T}[i',j']$. Thus, we have a contradiction because $\max(\delta(\mathbf{T}[i],\mathbf{T}[i'']), \delta(\mathbf{T}[i'],\mathbf{T}[i''])) \geq 2$. The other direction trivially follows. □

It follows that one can solve MinDMC in linear time when $d = 1$. In Section 4.2.1, we provide a linear-time algorithm for the case in which the alphabet size $|\Sigma|$ is a constant and $d = 2$. We also show that MinDMC can be solved in polynomial time for the case $d = 2$ even if the alphabet size $|\Sigma|$ is unbounded. In Section 4.2.2, we will show that MinDMC can be solved in polynomial time when $|\Sigma| = 2$ and $d = 3$ as well. On the negative side, we prove that MinDMC is NP-hard for $d = 4$ even if $|\Sigma| = 2$.

Note that one can view a binary matrix $\mathbf{T} \in \{0,1\}^{n\times\ell}$ as a family $\mathcal{F} = \{S_1, \ldots, S_n\}$ of n sets, where each set $S_i \subseteq [\ell]$ contains the column indices j such that $\mathbf{T}[i,j] = 1$. For our polynomial-time algorithms, we will exploit some notions from extremal set theory (refer to [Juk11] for an introduction), known as Δ-systems. These notions have been used to obtain efficient algorithms [Cyg+15; Eib+19; Fro+16]. They are defined as follows:

Definition 4.4 (Weak Δ-system)**.** We say that a set family $\mathcal{F} = \{S_1, \ldots, S_m\}$ is a *weak Δ-system* if there exists an integer $\lambda \in \mathbb{N}$ such that $|S_i \cap S_j| = \lambda$ for any pair of distinct sets $S_i, S_j \in \mathcal{F}$. The integer λ is called the intersection size of $\mathcal{F}$

Definition 4.5 (Strong Δ-system, Sunflower)**.** We say that a set family $\mathcal{F} = \{S_1, \ldots, S_m\}$ is a *strong Δ-system* or *sunflower* if there exists a subset $C \subseteq S_1 \cup \cdots \cup S_m$ such that $S_i \cap S_j = C$ for any pair of distinct sets $S_i, S_j \in \mathcal{F}$. We call the set C the *core* and the sets $P_i = S_i \setminus C$ the *petals* of $\mathcal{F}$.

Clearly, any strong Δ-system is a weak Δ-system. We say that a family $\mathcal{F}$ of sets is λ-uniform if $|S| = \lambda$ holds for each $S \in \mathcal{F}$. Deza [Dez74] showed that a λ-uniform weak Δ-system is a strong Δ-system when its cardinality is sufficiently large (more precisely, $|\mathcal{F}| \geq \lambda^2 - \lambda + 2$). In particular, the Deza [Dez73] also presented the tight bound for uniform weak Δ-systems in which the intersection size is exactly half of the cardinality of each set.

Lemma 4.6 ([Dez73])**.** *Let $\mathcal{F}$ be a (2μ)-uniform weak Δ-system with intersection size μ. If $|\mathcal{F}| \geq \mu^2 + \mu + 2$, then $\mathcal{F}$ is a strong Δ-system.*

We extend this result to the case in which the set size is odd.

Lemma 4.7. *Let $\mathcal{F}$ be a $(2\mu+1)$-uniform weak Δ-system with intersection size $\mu+1$. If $|\mathcal{F}| \geq \mu^2 + \mu + 3$, then $\mathcal{F}$ is a strong Δ-system.*

Proof. Let $S \in \mathcal{F}$ and let $\mathcal{F}' = \{T \triangle S \mid T \in \mathcal{F} \setminus \{S\}\}$. Here $T \triangle S$ denotes the symmetric difference $(T \setminus S) \cup (S \setminus T)$. Note that $\mathcal{F}'$ is a 2μ-uniform weak Δ-system with intersection size μ:

- For each $T \in \mathcal{F} \setminus \{S\}$, we have $|T \triangle S| = |S \setminus T| + |T \setminus S| = 2\mu$.

- We show that $|(T \triangle S) \cap (U \triangle S)| = \mu$ for each distinct $T, U \in \mathcal{F} \setminus \{S\}$. We rewrite

$$\begin{aligned}
&|(T \triangle S) \cap (U \triangle S)| \\
&= |((T \setminus S) \cup (S \setminus T)) \cap ((U \setminus S) \cup (S \setminus U))| \\
&= |((T \setminus S) \cap (U \setminus S)) \cup ((T \setminus S) \cap (S \setminus U)) \\
&\quad \cup ((S \setminus T) \cap (U \setminus S)) \cup ((S \setminus T) \cap (S \setminus U))| \\
&= |((T \cap U) \setminus S) \cup (S \setminus (T \cup U))| \\
&= |((T \cap U) \setminus S)| + |(S \setminus (T \cup U))|.
\end{aligned}$$

  Here the third equality follows from $(T \setminus S) \cap (S \cap U) = (S \setminus T) \cap (U \setminus S) = \emptyset$. Let $\kappa = |S \cap T \cap U|$. Since $|S \cap T| = |S \cap U| = \mu + 1$, it follows that $|(S \cap T) \setminus U| = |(S \cap U) \setminus T| = \mu - \kappa + 1$. Thus, we obtain

$$\begin{aligned}
|S \setminus (T \cup U))| &= |S| - |(S \cap T) \setminus U| - |(S \cap U) \setminus T| - |S \cap T \cap U| \\
&= (2\mu + 1) - (\mu - \kappa + 1) - (\mu - \kappa + 1) - \kappa = \kappa - 1.
\end{aligned}$$

  Moreover, we obtain

$$|((T \cap U) \setminus S)| = |T \cap U| - |S \cap T \cap U| = \mu - \kappa + 1.$$

  Now we have $|(T \triangle S) \cap (U \triangle S)| = |(S \setminus (T \cup U))| + |((T \cap U) \setminus S)| = \mu$.

Therefore, Lemma 4.6 tells us that $\mathcal{F}'$ is a strong Δ-system. Let C' be the core of $\mathcal{F}'$. Note that $|(T \triangle S) \cap S| = |S \setminus T| = \mu$ for each $T \in \mathcal{F} \setminus \{S\}$ or equivalently $|T' \cap S| = \mu$ for each $T' \in \mathcal{F}'$. We claim that $T' \cap S = C'$ for each $T' \in \mathcal{F}'$. Suppose not. Then, we have $C' \setminus S \neq \emptyset$ because $|T' \cap S| = \mu$. It follows that there exists an element $x \in (T' \setminus C') \cap S$ for each $T' \in \mathcal{F}'$. Since the set family $\{T' \setminus C' \mid T' \in \mathcal{F}'\}$ is pairwise disjoint, it gives us $|S| \geq \mu^2 + \mu + 2 > 2\mu + 1$, a contradiction. Thus, $\mathcal{F}$ is a sunflower with its core being $S \setminus C'$. □

4.2.1 Linear-time algorithm for the case $d = 2$

In this section, we present a linear-time algorithm for MinDMC when $d = 2$ and $|\Sigma|$ is a constant. We will prove that any binary matrix $\mathbf{T} \in \{0,1\}^{n \times \ell}$ fundamentally is a strong Δ-system (recall that any binary matrix can be viewed as a set family) when ℓ is sufficiently large. In fact, we reveal that any MinDMC instance with $\ell \geq 5$ and $d = 2$ is equivalent to the MinRMC instance on the same matrix with $d = 1$ even if the arbitrary alphabet $|\Sigma|$ is not binary. We start with two simple observations (also noted by Froese et al. [Fro+16]) on a binary matrix with diameter 2, which will be helpful in the subsequent proofs. Recall that $P_1(v)$ denotes the set $\{j \in [\ell] \mid v[j] = 1\}$ of column indices for a vector $v \in \Sigma^\ell$.

Observation 4.8. *Let $\mathbf{T} \in \{0,1\}^{n \times \ell}$ be a matrix with $\mathbf{T}[n] = 0^\ell$ and $\delta(\mathbf{T}) \leq 2$. Suppose that there exist $i_1, i_2 \in [n]$ such that $\delta(\mathbf{T}[i_1], \mathbf{T}[n]) = 1$ and $\delta(\mathbf{T}[i_2], \mathbf{T}[n]) = 2$. Then, we have $P_1(\mathbf{T}[i_1]) \subseteq P_1(\mathbf{T}[i_2])$.*

Observation 4.9. *Let $\mathbf{T} \in \{0,1\}^{n \times \ell}$ be a matrix with $\mathbf{T}[n] = 0^\ell$ and $\delta(\mathbf{T}) \leq 2$. Suppose that there exist $i_2, i_2' \in [n]$ such that $\delta(\mathbf{T}[i_2], \mathbf{T}[n]) = 2$ and $\delta(\mathbf{T}[i_2'], \mathbf{T}[n]) = 2$. Then, we have $|P_1(\mathbf{T}[i_2]) \cap P_1(\mathbf{T}[i_2'])| = 1$ if $\mathbf{T}[i_2]$ and $\mathbf{T}[i_2']$ are distinct.*

In the next lemma, we show that the number of row vectors that contain two 1's restricts the number of columns.

Lemma 4.10. *Let $\mathbf{T} \in \{0,1\}^{n\times\ell}$ be a matrix with $\mathbf{T}[n] = 0^\ell$ and $\delta(\mathbf{T}) \leq 2$, where every column is dirty. Let $n_2 = |\{i \in [n] \mid \delta(\mathbf{T}[i], \mathbf{T}[n]) = 2\}|$ be the number of row vectors that contain exactly two 1's. If $n_2 \geq 1$, then $\ell = |\bigcup_{i\in[n]} P_1(\mathbf{T}[i])| \leq n_2 + 1$.*

Proof. We prove the claim by induction over n_2. If $n_2 = 1$, then we have at most 2 columns because of Observation 4.8. Suppose that $n_2 \geq 2$. Choose an arbitrary row index $i \in [n]$ such that $\delta(\mathbf{T}[i], \mathbf{T}[n]) = 2$. Then, we have

$$\left|\bigcup_{i'\in[n]} P_1(\mathbf{T}[i'])\right| = \left|\bigcup_{i'\in[n]\setminus\{i\}} P_1(\mathbf{T}[i'])\right| + \left|P_i(\mathbf{T}[i]) \setminus \bigcup_{i'\in[n]\setminus\{i\}} P_1(\mathbf{T}[i'])\right| \leq n_2 + 1.$$

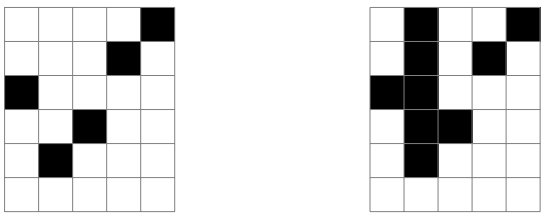
We claim that the last inequality holds. Since $\mathbf{T}[[n] \setminus \{i\}, :]$ has $n_2 - 1$ row vectors that contain exactly two 1's, the induction hypothesis gives us that $|\bigcup_{i'\in[n]\setminus\{i\}} P_1(\mathbf{T}[i'])| \leq n_2$. For the second term, observe that $|P_i(\mathbf{T}[i]) \setminus \bigcup_{i'\in[n]\setminus\{i\}} P_1(\mathbf{T}[i'])| \leq |P_i(\mathbf{T}[i]) \setminus P_i(\mathbf{T}[i''])|$ for some row index $i'' \in [n]$ with $\delta(\mathbf{T}[i''], \mathbf{T}[n]) = 2$. Such a row index i'' exists because $n_2 \geq 2$. Hence, it follows from Observation 4.9 that the second term is at most 1. □

We show that a matrix with diameter at most 2 has a radius at most 1 as long as it has sufficiently many columns.

Lemma 4.11. *Let $\mathbf{T} \in \Sigma^{n\times\ell}$ be a matrix with $\delta(\mathbf{T}) \leq 2$. Suppose that every column of $\mathbf{T}$ is dirty. If $\ell \geq 5$, then there exists a vector $v \in \{0,1\}^\ell$ such that $\delta(v, \mathbf{T}) \leq 1$.*

Proof. Let $\mathbf{T}' \in \{0,1\}^{n\times\ell}$ be a matrix such that $\mathbf{T}'[i,j] = \chi[\mathbf{T}[i,j] \neq \mathbf{T}[n,j]]$ for each $i \in [n]$ and $j \in [\ell]$. Note that $\mathbf{T}'[n] = 0^\ell$. We distinguish cases by the number of distinct row vectors of $\mathbf{T}'$ that contain exactly two 1's. Let $\mathcal{V} = \{\mathbf{T}'[i] \mid i \in [n], \delta(\mathbf{T}'[i], 0^\ell) = 2\}$ be the set of such row vectors. Note that we are immediately done if $\mathcal{V} = \emptyset$ because $\delta(\mathbf{T}[n], \mathbf{T}) \leq 1$ (see Figure 4.1a for an illustration). Note also that Lemma 4.10 excludes the case $|\mathcal{V}| \in \{1,2,3\}$ since $\ell \geq 5$. Hence, we can assume that $|\mathcal{V}| \geq 4$.

For any two arbitrary distinct row vectors $u, u' \in \mathcal{V}$ of $\mathbf{T}'$, it holds that $|P_1(u) \cap P_1(u')| = 1$ due to Observation 4.9. Hence, the set family $\{P_1(u) \mid u \in \mathcal{V}\}$ forms a 2-uniform weak Δ-system. In fact, it follows from Lemma 4.6 that this is a strong Δ-system because $\mathcal{V}$ contains at least four distinct row vectors. Let $\{j_{\mathrm{core}}\}$ denote the core of the strong Δ-system. Suppose that there exists $i_1 \in [n-1]$ such that $\mathbf{T}'[i_1, j_{\mathrm{core}}] = 0$. Since $\mathbf{T}'[i_1] \notin \mathcal{V}$ (otherwise $\mathbf{T}'[i_1, j_{\mathrm{core}}] = 1$), the row vector $\mathbf{T}'[i_1]$ contains exactly one 1. Let $P_1(\mathbf{T}'[i_1]) = \{j_1\}$. Then, it follows from Observation 4.8 that $j_1 \in P_1(u) \setminus \{j_{\mathrm{core}}\}$ and $j_1 \in P_1(u') \setminus \{j_{\mathrm{core}}\}$ for two distinct row vectors $u, u' \in \mathcal{V}$. We can rewrite it as $j_1 \in (P_1(u) \cap P_1(u')) \setminus \{j_{\mathrm{core}}\}$. However, this is a contradiction because $j_1 \in (P_1(u) \cap P_1(u')) \setminus \{j_{\mathrm{core}}\} = \emptyset$ due to Observation 4.9. Hence, we can assume that $\mathbf{T}'[i] = 0^\ell$ or $\mathbf{T}'[i, j_{\mathrm{core}}] = 1$ for each $i \in [n-1]$ (see Figure 4.1b for an illustration). We claim that $\mathbf{T}[i] = \mathbf{T}[i']$ for each $i, i' \in [n]$ with $\mathbf{T}'[i], \mathbf{T}'[i'] \in \mathcal{V}$. Let $i'' \in [n]$ be such that $P_1(\mathbf{T}[i'']) \setminus P_1(\mathbf{T}[i]) \neq \emptyset$ and $P_1(\mathbf{T}[i'']) \setminus P_1(\mathbf{T}[i']) \neq \emptyset$. Note that such i'' exists because $\ell \geq 5$. Then, we have $\delta_{[\ell]\setminus\{j_{\mathrm{core}}\}}(\mathbf{T}[i], \mathbf{T}[i'']) = \delta_{[\ell]\setminus\{j_{\mathrm{core}}\}}(\mathbf{T}[i], \mathbf{T}[i'']) = 2$. Hence it must hold that $\mathbf{T}[i, j_{\mathrm{core}}] = \mathbf{T}[i', j_{\mathrm{core}}]$ to satisfy $\delta(\mathbf{T}) \leq 2$. Let $\sigma = \mathbf{T}[i, j_{\mathrm{core}}]$ for $i \in [n]$ with $\mathbf{T}'[i] \in \mathcal{V}$. Note that each row vector $\mathbf{T}[i]$ is of one of the following types:



(a) The case $|\mathcal{V}| = 0$. (b) The case $|\mathcal{V}| \geq 4$.

Figure 4.1: Illustration of Lemma 4.11 with $n = 6$ and $j_{\mathrm{core}} = 2$. A white cell denotes 0 and a black cell denotes 1.

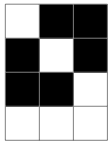


Figure 4.2: An example of a matrix with both radius and diameter 2. A white cell denotes 0 and a black cell denotes 1.

- $\mathbf{T}[i, j_{\mathrm{core}}] \neq \sigma$ and $\mathbf{T}[i, [\ell] \setminus \{j_{\mathrm{core}}\}] = \mathbf{T}[n, [\ell] \setminus \{j_{\mathrm{core}}\}]$.
- $\mathbf{T}[i, j_{\mathrm{core}}] = \sigma$ and $\delta_{[\ell]\setminus\{j_{\mathrm{core}}\}}(\mathbf{T}[i], \mathbf{T}[n]) \leq 1$.

Hence, it holds that $\delta(v, \mathbf{T}) \leq 1$ for a vector $v \in \Sigma^{\ell}$ such that $v[j_{\mathrm{core}}] = \sigma$ and $v[j] = \mathbf{T}[n, j]$ for each $j \in [\ell] \setminus \{j_{\mathrm{core}}\}$. □

Note that Lemma 4.11 does not necessarily hold when $\ell \leq 4$, as shown in Figure 4.2. We give a reduction from MinDMC to MinRMC in the next theorem.

Theorem 4.12. MinDMC *can be solved in time* $O(n\ell)$ *when* $|\Sigma|$ *is a constant and* $d = 2$.

Proof. Let $I = (\mathbf{S}, d)$ be an instance of MinDMC where $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ and $d = 2$. We use the algorithm described in Lemma 4.1 if $\ell \leq 4$. Since $|\Sigma|$ and ℓ are both constants here, this procedure completes in time $O(n\ell)$. So we can assume that $\ell \geq 5$.

We claim that I is a **Yes** instance if and only if the MinRMC instance $I' = (\mathbf{S}, 1)$ is **Yes** instance.

($\Rightarrow$) Let $\mathbf{T}$ be a completion of $\mathbf{S}$ with $\delta(\mathbf{T}) \leq 2$. Since $\ell \geq 5$, there exists a vector v such that $\delta(v, \mathbf{T}) \leq 1$ by Lemma 4.11. It follows that I' is a **Yes** instance.

($\Leftarrow$) Let v be a solution of I'. Let $\mathbf{T}$ be the matrix such that $\mathbf{T}[i] = \mathbf{S}[i] \oplus v$ for each $i \in [n]$ ($\mathbf{S}[i] \oplus v$ denotes the vector obtained by filling each missing entry in $\mathbf{S}[i]$ with the symbol in the corresponding position of v). Then, we have $\delta(v, \mathbf{T}[i]) \leq 1$ for each $i \in [n]$. By the triangle inequality, we obtain $\delta(\mathbf{T}[i], \mathbf{T}[i']) \leq \delta(v, \mathbf{T}[i]) + \delta(v, \mathbf{T}[i']) \leq 2$

Since MinRMC can be solved in linear time when $d = 1$ (Theorem 3.2), MinDMC can be solved in linear time when $|\Sigma|$ is a constant and $d = 2$. □

We extend Theorem 4.12 to the case of unbounded alphabet size. We have to show that MinDMC can be solved in polynomial time when $\ell \leq 4$.

Lemma 4.13. MinDMC *can be solved in time* $O(|\Sigma|^6 \cdot n^3)$ *when* $\ell \leq 4$.

Proof. First we verify whether a completion within diameter 1 is feasible by Lemma 4.3. Suppose that the input matrix $\mathbf{S} \in (\Sigma \cup \{*\})^{n \times \ell}$ can be completed into a matrix $\mathbf{T} \in \Sigma^{n \times \ell}$ with $\delta(\mathbf{T}) = 2$. Let $i_1, i_2 \in [n]$ be such that $\delta(\mathbf{T}[i_1], \mathbf{T}[i_2]) = 2$ and $Q(\mathbf{T}[i_1], \mathbf{T}[i_2]) = \{j_1, j_2\}$ Assume without loss generality that $j_1 = 1$ and $j_2 = 2$. Suppose that $\mathbf{T}[i_1] = \alpha_1\alpha_2\gamma_1\gamma_2$ and $\mathbf{T}[i_2] = \beta_1\beta_2\gamma_1\gamma_2$ for $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2 \in \Sigma$. Since every column of $\mathbf{T}$ is dirty, there are row indices $i \neq i' \in [n]$ such that $\mathbf{T}[i, 3] = \delta_1 \neq \gamma_1$ and $\mathbf{T}[i', 4] = \delta_2 \neq \gamma_2$. Then we have two cases: either (i) $\mathbf{T}[i] = \alpha_1\beta_2\delta_1\gamma_2$ and $\mathbf{T}[i] = \alpha_1\beta_2\gamma_1\delta_2$ or (ii) $\mathbf{T}[i] = \beta_1\alpha_2\delta_1\gamma_2$ and $\mathbf{T}[i] = \beta_1\alpha_2\gamma_1\delta_2$. Assume that case (i) holds. Then for every $i \in [n]$, the row vector $\mathbf{T}[i]$ is in the form $\sigma\beta_2\gamma_1\gamma_2$, $\alpha_1\sigma\delta_1\gamma_2$, $\alpha_1\beta_2\sigma\gamma_2$, or $\alpha_1\beta_2\gamma_1\sigma$ for an arbitrary symbol $\sigma \in \Sigma$. We can try all possible candidates for $j_1, j_2 \in [\ell]$, $i_1, i_2 \in [n]$, and $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2 \in \Sigma$ in time $O(|\Sigma|^6 \cdot n^2)$ and verify for each row vector whether it can be completed into one of the four types. We can handle case (ii) analogously. □

Since our reduction to MinRMC in the case of $\ell \geq 5$ can be applied even if the alphabet size is unbounded, we obtain the following theorem.

Theorem 4.14. MinDMC *can be solved in time* $O(|\Sigma|^6 \cdot n^3 + n\ell)$ *when* $d = 2$.

4.2.2 Polynomial-time algorithm for the case $|\Sigma| = 2$ and $d = 3$

We will prove that MinDMC can be solved in polynomial time when $|\Sigma| = 2$ and $d = 3$ in this section. The overall idea is similar to the case $d = 2$. We first show that the set family corresponding to a binary matrices with diameter at most 3 contains a strong Δ-system by Lemma 4.7 (see Lemma 4.17). We then show that each row vector (set) must fulfill certain properties due to the presence of the strong Δ-system (see Lemma 4.18). Finally in Theorem 4.19, we obtain a polynomial-time algorithm by reducing to the tractable cases of MinRMC and MinDMC. We start with an observation on a matrix whose diameter is at most 3. We remark that this observation is also noted by Froese et al. [Fro+16].

Observation 4.15. *Let* $\mathbf{T} \in \{0,1\}^{n \times \ell}$ *be a matrix with* $\mathbf{T}[n] = 0^\ell$ *and* $\delta(\mathbf{T}) \leq 3$. *Suppose that there exist row indices* $i_1, i_2, i_3, i_3' \in [n]$ *such that* $\delta(\mathbf{T}[i_1], \mathbf{T}[n]) = 1$, $\delta(\mathbf{T}[i_2], \mathbf{T}[n]) = 2$, $\delta(\mathbf{T}[i_3], \mathbf{T}[n]) = 3$, *and* $\delta(\mathbf{T}[i_3], \mathbf{T}[n]) = 3$. *Then, we have the following:*

(i) $P_1(\mathbf{T}[i_1]) \subseteq P_1(\mathbf{T}[i_3])$.

(ii) $|P_1(\mathbf{T}[i_2]) \cap P_1(\mathbf{T}[i_3])| \geq 1$.

(iii) $|P_1(\mathbf{T}[i_3]) \cap P_1(\mathbf{T}[i_3'])| \geq 2$.

With this observation, we show the following lemma analogous to Lemma 4.10. The proof proceeds in the same way using induction.

Lemma 4.16. *Let* $\mathbf{T} \in \{0,1\}^{n \times \ell}$ *be a matrix with* $\mathbf{T}[n] = 0^\ell$ *and* $\delta(\mathbf{T}) \leq 3$, *where every column is dirty. Let* $I_3 = \{i \in [n] \mid \delta(\mathbf{T}[i], \mathbf{T}[n]) = 3\}$ *be the row vectors that contain exactly three 1's and let* $n_3 = |I_3|$. *If* $n_3 \geq 1$, *then* $|\bigcup_{i \in I_3} P_1(\mathbf{T}[i])| \leq n_3 + 2$.

Proof. We prove the claim by induction over n_3. First, note that if $n_3 = 1$, then we have $|\bigcup_{i\in I_3} P_1(\mathbf{T}[i])| = 3$. Suppose that $n_3 \geq 2$. Choose an arbitrary row index $i \in I_3$. Then, we have

$$\left|\bigcup_{i'\in I_3} P_1(\mathbf{T}[i'])\right| = \left|\bigcup_{i'\in I_3\setminus\{i\}} P_1(\mathbf{T}[i'])\right| + \left|P_i(\mathbf{T}[i]) \setminus \bigcup_{i'\in I_3\setminus\{i\}} P_1(\mathbf{T}[i'])\right| \leq n_3 + 2.$$

The last inequality is due to the fact that the first term is at most $n_3 + 1$ by induction hypothesis and the second term is at most 1 by Observation 4.15 (iii). □

In order to obtain a strong Δ-system by Lemma 4.7, one has to show that a binary matrix of diameter 3 contains a row vector u such that there are at least five row vectors at distance 3 from u. In the next lemma, we show that it holds when the matrix has at least 10 columns.

Lemma 4.17. *Let $\mathbf{T} \in \{0,1\}^{n\times\ell}$ be a matrix with $\ell \geq 10$ and $\delta(\mathbf{T}) = 3$. Suppose that $\mathbf{T}$ contains distinct rows and that $\mathbf{T}[:, j]$ contains both 0 and 1 for each $j \in [\ell]$. Then, there exist row indices $i_0 \in [n]$ and $I \subseteq [n]$ such that $|I| \geq 5$ and $\delta(\mathbf{T}[i_0], \mathbf{T}[i]) = 3$ for each $i \in I$.*

Proof. Without loss of generality, assume that $\mathbf{T}[n-1] = 1110^{\ell-3}$ and $\mathbf{T}[n] = 0^\ell$. Let $I_x = \{i \in [n] \mid |P_1(\mathbf{T}[i])| = x\}$ be the row vectors in which the number of 1's is exactly x for $x \in \{1, 2, 3\}$. If $|I_3| \geq 5$, then the statement holds with $i_0 = n$. Assume that $|I_3| \leq 4$. Due to Lemma 4.16, we have $|J_3| \leq 6$ for $J_3 = \{j \in [\ell] \mid \exists i \in I_3 \colon \mathbf{T}[i, j] = 1\}$. For each $j \in [\ell] \setminus J_3$, there exists $i \in [n]$ such that $\mathbf{T}[i, j] = 1$ because every column is dirty. We claim that these row indices must be in I_2. Observation 4.15 (i) yields that there exists no $i \in I_1$ such that $P_1(\mathbf{T}[i]) \setminus J \neq \emptyset$. This implies those row indices cannot be in I_1. By definition of J_3, they cannot be in I_3 either. Hence, we see that for each $j \in [\ell] \setminus J_3$, there exists a row index $i \in I_2$ such that $\mathbf{T}[i, j] = 1$. These row indices are distinct: Any row indices $i, i' \in [n]$ such that $\mathbf{T}[i, j] = \mathbf{T}[i', j'] = 1$ for distinct column indices $j, j' \in [\ell] \setminus J_3$ must satisfy that $i \neq i'$. To see this, note that at least one of $\mathbf{T}[i, 1]$, $\mathbf{T}[i, 2]$, and $\mathbf{T}[i, 3]$ equals 1 for each $i \in I_2$ (Observation 4.15 (ii)). In fact, exactly one of $\mathbf{T}[i, 1]$, $\mathbf{T}[i, 2]$, and $\mathbf{T}[i, 3]$ equals 1 for each $i \in I_2$ such that $\mathbf{T}[i, j] = 1$ for some $j \in [\ell] \setminus J_3$ because $\mathbf{T}[i]$ contains exactly two 1's. It means that $\delta(\mathbf{T}[i], \mathbf{T}[n-1]) = 3$ for those row indices i's. Let $I_2' \subseteq I_2$ be the set of such row indices. I_2' contains at least $|[\ell] \setminus J_3| \geq 4$ distinct row indices. It is easy to verify that the statement of the lemma holds with $i_0 = n - 1$ and $I = I_2' \cup \{n\}$. □

We remark that the constraint $\ell \geq 10$ in Lemma 4.17 is tight: There exists an 8×9 binary matrix in which Lemma 4.17 does not hold (see Figure 4.3). With Lemma 4.17 at hand, we are ready to reveal the structure of a diameter-3 matrix. Intuitively speaking, a matrix $\mathbf{T}$ with $\delta(\mathbf{T}) = 3$ contains two column indices j_1 and j_2 with the following property: In the submatrix $\mathbf{T}[:, [\ell] \setminus \{j_1, j_2\}]$ obtained by removing the j_1-th and j_2-th column vectors, there exists a row vector such that each row vector differs from it on at most one column. In fact, we can see in Figure 4.4 that each row vector in the submatrix contains at most one 1. More precisely, we prove the following:

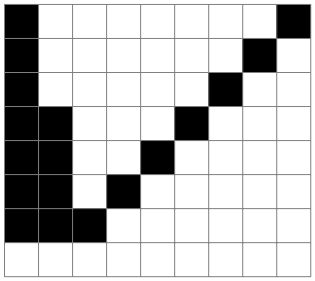


Figure 4.3: A matrix with $\ell = 9$ in which Lemma 4.17 does not hold. Observe that for each row vector, there are exactly four row at distance three.

Lemma 4.18. *Let $\mathbf{T} \in \{0,1\}^{n \times \ell}$ be a matrix with $\ell \geq 10$ and $\delta(\mathbf{T}) = 3$. Suppose that $\mathbf{T}$ contains distinct rows and dirty columns. Then, there exist a row index $i_0 \in [n]$ and distinct column indices $J = \{j_1, j_2\} \subseteq [\ell]$ such that $[n] = H_0 \cup H_1 \cup H_2 \cup H_3$ where*

- $H_0 = \{i \in [n] \mid \mathbf{T}[i, [\ell] \setminus J] = \mathbf{T}[i_0, [\ell] \setminus J]\}$,
- $H_1 = \{i \in [n] \mid \mathbf{T}[i, j_1] = \mathbf{T}[i_0, j_1], \mathbf{T}[i, j_2] \neq \mathbf{T}[i_0, j_2], \delta_{[\ell]\setminus J}(\mathbf{T}[i], \mathbf{T}[i_0]) = 1\}$,
- $H_2 = \{i \in [n] \mid \mathbf{T}[i, j_1] \neq \mathbf{T}[i_0, j_1], \mathbf{T}[i, j_2] = \mathbf{T}[i_0, j_2], \delta_{[\ell]\setminus J}(\mathbf{T}[i], \mathbf{T}[i_0]) = 1\}$,
- $H_3 = \{i \in [n] \mid \mathbf{T}[i, j_1] \neq \mathbf{T}[i_0, j_1], \mathbf{T}[i, j_2] \neq \mathbf{T}[i_0, j_2], \delta_{[\ell]\setminus J}(\mathbf{T}[i], \mathbf{T}[i_0]) = 1\}$.

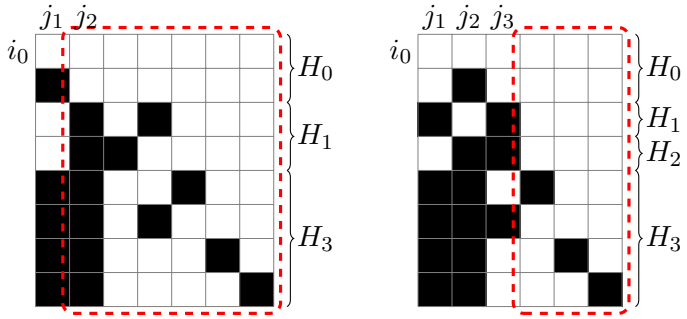
See Figure 4.4 for an illustration of the partition. Moreover, exactly one of the following holds:

(i) $|H_1| = 0$ *or* $|H_2| = 0$.

(ii) $|H_1| = 1$ *and* $|H_2| = 1$. *Furthermore, there exists a column index $j_3 \in [\ell]$ such that $\mathbf{T}[i_1, j] = \mathbf{T}[i_2, j_3] = \chi[j = j_3]$ for each $j \in [3, \ell]$ for $H_1 = \{i_1\}$ and $H_2 = \{i_2\}$.*

Proof. By Lemma 4.17, we have a row index i_0 and a set I_3 of row indices such that $|I_3| \geq 5$ and $\delta(\mathbf{T}[i_0], \mathbf{T}[i_3]) = 3$ for each $i_3 \in I_3$. Without loss of generality, assume that $\mathbf{T}[i_0] = 0^\ell$. Let $I_x = \{i \in [n] \mid |P_1(\mathbf{T}[i])| = x\}$ for $x \in \{0,1,2,3\}$. Consider the set family $\mathcal{F} = \{P_1(\mathbf{T}[i_3]) \mid i_3 \in I_3\}$. By definition, $\mathcal{F}$ is a 3-uniform set system. Moreover, $\mathcal{F}$ is a weak Δ-system with intersection size 2 (Observation 4.15 (iii)). Hence, Lemma 4.7 gives us columns indices $j_1, j_2 \in [\ell]$ such that $\mathbf{T}[i_3, j_1] = \mathbf{T}[i_3, j_2] = 1$ for each $i_3 \in I_3$. Without loss of generality, assume that $j_1 = 1$ and $j_2 = 2$. By definition, we have $I_3 \subseteq H_3$. For each $i \in I_1$, we have that exactly one of the first two entries is 1 due to Observation 4.15 (i). Hence, we have that $i \in H_0$ if $i \in I_1$, meaning that $I_1 \subseteq H_0$. Now suppose that $i \in I_2$. It follows from Observation 4.15 (ii) that at least one of the first two entries of $\mathbf{T}[i]$ is 1. If the first two entries are both 1, then we have $i \in H_0$ by definition. If exactly one of $\mathbf{T}[i,1]$ and $\mathbf{T}[i,2]$ equals one, then we have $i \in H_1$ or $i \in H_2$. This means that $I_2 \subseteq H_0 \cup H_1 \cup H_2$. Note also that $I_0 \subseteq H_0$. Since $[n] = I_0 \cup I_1 \cup I_2 \cup I_3$, we obtain $[n] = H_0 \cup H_1 \cup H_2 \cup H_3$.

Now we show that property (i) or (ii) must hold. Suppose that $|H_1| \geq 2$ and $|H_2| \geq 1$. Let $i_1, i_1' \in H_1$ and $i_2 \in H_2$ be distinct row indices. Then, $\delta(\mathbf{T}[i_1], \mathbf{T}[i_2]) = 4$ or $\delta(\mathbf{T}[i_1'], \mathbf{T}[i_2]) = 4$ must hold because $\mathbf{T}$ contains distinct rows. This is a contradiction,



(a) Lemma 4.18 (i) (b) Lemma 4.18 (ii)

Figure 4.4: Illustration of Lemma 4.18 with smaller ℓ. Lemma 4.18 does not hold in general for $\ell < 10$, but this example serves our purpose. Each white cell denotes 0 and each black cell denotes 1. The submatrix marked by the dashed line in each figure depicts the underlying idea of the reduction in Theorem 4.19. Observe that the diameter of the submatrix in Figure 4.4a is 2. In Figure 4.4b, observe that each row vector contains at most one 1 in the submatrix. Furthermore, note that a vector in the entire matrix starts with 110 if it contains 1 in the submatrix.

and thus we have that $|H_1| = 0$, $|H_1| = 1$, or $|H_2| = 0$. We obtain $|H_2| = 0$, $|H_2| = 1$, or $|H_1| = 0$ analogously. If $|H_1| = 0$ or $|H_2| = 0$, then property (i) is satisfied. Otherwise we have $|H_1| = |H_2| = 1$. Since $\delta(\mathbf{T}) \leq 3$, it holds that $P_1(\mathbf{T}[i_1]) = \{1, j_3\}$ and $P_1(\mathbf{T}[i_2]) = \{2, j_3\}$ for some $j_3 \in [\ell]$ (otherwise we have $\delta(\mathbf{T}[i_1], \mathbf{T}[i_2]) = 4$). □

We exploit the structure described in Lemma 4.18 to obtain a polynomial-time algorithm.

Theorem 4.19. MinDMC *can be solved in time* $O(n\ell^4)$ *and* $O(n\ell + n^5)$ *when* $|\Sigma| = 2$ *and* $d = 3$.

Proof. We first apply Theorem 4.12 to determine whether there exists a completion $\mathbf{T} \in \{0,1\}^{n \times \ell}$ of $\mathbf{S}$ such that $\delta(\mathbf{T}) \leq 2$. If so, we can conclude that I is a **Yes** instance. Otherwise it remains to determine whether there exists a solution matrix $\mathbf{T}$ with $\delta(\mathbf{T}) = 3$. We can assume that $\ell \geq 10$ by Lemma 4.1. Suppose that there exists such a matrix $\mathbf{T} \in \{0,1\}^{n \times \ell}$. Consider the matrix $\mathbf{T}'$ obtained by removing duplicate row vectors of $\mathbf{T}$. Note that $\mathbf{T}'$ meets the preconditions of Lemma 4.18, which gives two cases: Lemma 4.18 (i) and (ii). We are going to use a reduction to MinDMC with $d = 2$ to handle case (i) and RCMC (recall that RCMC is a generalization of MinRMC where the distance bound can be specified for each row vector) for case (ii).

We describe the construction. Let us define the instance I_j of MinDMC to be $(\mathbf{S}[:, [\ell] \setminus \{j\}], 2)$ for each $j \in [\ell]$ and let $\mathcal{I}_1 = \{I_j \mid j \in [\ell]\}$. This corresponds to case (i). The basic idea is that the diameter decreases to 2 once an appropriate column vector is removed (in Figure 4.4a the submatrix leaving out the first column vector has diameter 2). Now we describe the construction for case (ii). Let $j_1, j_2, j_3 \in [\ell]$ be three distinct column indices and let $J = (j_1, j_2, j_3)$ be a vector. For any such vector J and any binary vector $X = [x_1, x_2, x_3] \in \{0,1\}^3$, let us define an instance $I_{J,X} = (\mathbf{S}_{J,X}, d_1, \ldots, d_n)$ of RCMC as follows:

- $\mathbf{S}_{J,X} = \mathbf{S}[:, [\ell] \setminus \{j_1, j_2, j_3\}]$.
- For each $i \in [n]$, let

$$d_i = \begin{cases} 1 & \text{if } \delta(\mathbf{S}[i, \{j_1, j_2, j_3\}], [1 - x_1, 1 - x_2, x_3]) = 0 \\ 0 & \text{otherwise.} \end{cases}$$

We define $\mathcal{I}_2$ as those instances $I_{J,X}$ in which $\delta(\mathbf{S}[i, \{j_1, j_2, j_3\}], [1 - x_1, 1 - x_2, x_3]) \leq 2$ holds for each $i \in [n]$. In other words, we exclude instances such that $\mathbf{S}[i, \{j_1, j_2, j_3\}] = [x_1, x_2, 1-x_3]$ holds for some $i \in [n]$. The idea of the reduction is illustrated in Figure 4.4b for the case $(j_1, j_2, j_3) = (1, 2, 3)$: For any row vector that does not start with 110, the remaining entries all must be 0. Otherwise at most one of the remaining entries can become 1. We claim that I is a **Yes** instance if and only if at least one instance in $\mathcal{I}_1$ or $\mathcal{I}_2$ is a **Yes** instance.

($\Rightarrow$) By Lemma 4.18, $\mathbf{T}'$ admits a row index $i'_0 \in [n']$, column indices j'_1, j'_2, (and j'_3) $\in [\ell]$, and a partition of row indices H'_0, H'_1, H'_2, H'_3 such that Lemma 4.18 (i) or (ii) holds. We examine each case.

- Suppose that Lemma 4.18 (i) holds. We will show that at least one instance of $\mathcal{I}_1$ is a **Yes** instance. Without loss of generality, assume that $\mathbf{T}'[i'_0] = 0^\ell$, $j'_1 = 1$, $j'_2 = 2$, and $|H'_2| = 0$ (see Figure 4.4a). We claim that $\delta(\mathbf{T}'[i', [2, \ell]], 10^{\ell-2}) \leq 1$ holds for each $i' \in [n']$. If $i' \in H'_0$, then we have $\mathbf{T}'[i, [3, \ell]] = 0^{\ell-2}$. Otherwise we have $i \in H_1 \cup H_3$, which gives us $\mathbf{T}'[i, 2] = 1$ and $\delta(\mathbf{T}[i, [3, \ell-1]], 0^{\ell-2}) \leq 1$. In both cases, we obtain $\delta(\mathbf{T}'[i, [2, \ell]], 10^{\ell-2}) \leq 1$. It follows from the triangle inequality then that $\delta(\mathbf{T}'[:, [2, \ell]]) \leq 2$. It means that $\delta(\mathbf{T}[:, [2, \ell]]) \leq 2$ and thus the instance $I_1 \in \mathcal{I}_1$ is a **Yes** instance.

- Suppose that Lemma 4.18 (ii) holds. We show that $I_{J,X}$ is a **Yes** instance where $J = (j'_1, j'_2, j'_3)$ and $X = (\mathbf{T}'[i'_0, j'_1], \mathbf{T}[i'_0, j'_2], \mathbf{T}[i'_0, j'_3])$. Without loss of generality, assume that $\mathbf{T}'[i'_0] = 0^\ell$ and $(j'_1, j'_2, j'_3) = (1, 2, 3)$ (see Figure 4.4b). Observe that $\mathbf{T}'[i', [4, \ell]] = 0^{\ell-3}$ holds for each $i' \in H_0 \cup H_1 \cup H_2$. Consider a row index $i' \in H_3$. If $\mathbf{T}'[i', [3]] = 111$, then we have $\mathbf{T}'[i', [4, \ell]] = 0^{\ell-3}$ by Lemma 4.18. Otherwise we have $\mathbf{T}'[i', [3]] = 110$ and $\delta(\mathbf{T}'[i', [4]], 0^{\ell-3}) \leq 1$. It means that $I_{J,X}$ is a **Yes** instance with solution $0^{\ell-3}$.

($\Leftarrow$) There are two cases, namely the case in which a **Yes** instance contained in $\mathcal{I}_1$ and in $\mathcal{I}_2$.

- If $I_j \in \mathcal{I}_1$ is a **Yes** instance, then there exists a completion $\mathbf{T}_j \in \{0, 1\}^{n \times (\ell-1)}$ of $\mathbf{S}[:, [\ell] \setminus \{j\}]$ such that $\delta(\mathbf{T}_j) \leq 2$. Let $\mathbf{T}$ be a matrix in which

$$\mathbf{T}[i] = (\mathbf{T}_j[1], \ldots, \mathbf{T}_j[j-1], \chi[\mathbf{S}[i, j] = 1], \mathbf{T}_j[j], \ldots, \mathbf{T}_j[\ell-1])$$

for each $i \in [n]$. Because we set $\mathbf{T}[i, j] = \mathbf{S}[i, j]$ whenever $\mathbf{S}[i, j] \neq *$, the resulting matrix $\mathbf{T}$ is a completion $\mathbf{S}$. Since $\mathbf{T}$ contains one more column than $\mathbf{T}_j$, it follows that $\delta(\mathbf{T}) \leq 3$.

- Suppose that an instance of $\mathcal{I}_2$ is a **Yes** instance. Without loss of generality, assume that $\mathcal{I}_{(1,2,3),(x_1,x_2,x_3)}$ is a **Yes** instance with a solution $t' = (t'_1, \ldots, t'_{\ell-3}) \in \{0,1\}^{\ell-3}$. Let us define a vector $t = (1-x_1, 1-x_2, x_3, t'_1, \ldots, t'_{\ell-3}) \in \{0,1\}^{\ell}$. We construct a completion $\mathbf{T}$ of $\mathbf{S}$ by setting $\mathbf{T}[i] = \mathbf{S}[i] \oplus t$ or each $i \in [n]$. We will show that $\delta(\mathbf{T}[i], \mathbf{T}[i']) \leq 3$ holds for each $i, i' \in [n]$.

  We start with the following two observations: First observe that $\delta(\mathbf{T}[i], t) \leq 1$ for each row index $i \in [n]$ such that $d_i = 1$. Moreover, we have $\delta(\mathbf{T}[i], t) \leq 2$ for each $i \in [n]$ such that $d_i = 0$, which follows from the fact that $\mathbf{T}[i, [3]] \neq (x_1, x_2, 1-x_3)$ (this is because we excluded from $\mathcal{I}_2$ instances in which $\mathbf{S}[i, [3]] = (x_1, x_2, 1-x_3)$) and $\mathbf{T}[i, [4, \ell]] = t'$. Suppose that $d_i = 1$. Then, we have $\delta(\mathbf{T}[i], \mathbf{T}[i']) \leq \delta(\mathbf{T}[i], t) + \delta(\mathbf{T}[i'], t) \leq 3$ by the triangle inequality. If $d_{i'} = 0$, then we obtain $\delta(\mathbf{T}[i], \mathbf{T}[i']) \leq 3$ analogously. So assume that $d_i = d_{i'} = 0$. Since $\mathbf{T}[i, [4, \ell]] = \mathbf{T}[i', [4, \ell]] = t'$, we have that $\delta(\mathbf{T}[i], \mathbf{T}[i']) = 3$ in this case as well. Thus, we have $\delta(\mathbf{T}) \leq 3$.

Now we analyze the time complexity. It takes $O(n\ell)$ time to solve each MinDMC instance in $\mathcal{I}_1$ by Theorem 4.12. Moreover, we construct $O(\ell^3)$ RCMC instances $\mathcal{I}_2$, each of which can be solved in $O(n\ell)$ time by Theorem 3.2. Hence, MinDMC can be solved in time $O(n\ell^4)$ when $|\Sigma| = 3$ and $d = 2$. Once all columns that are not dirty are removed (which takes time $O(n\ell)$), we can assume that $\ell \leq nd$, using the argument of Lemma 3.7. Hence, MinDMC can be also solved in time $O(n\ell + n^5)$ when $|\Sigma| = 3$ and $d = 2$.. □

4.2.3 NP-hardness for the case $d = 4$

We have seen in the previous sections that MinDMC can be solved in polynomial time when $|\Sigma| = 2$ and $d \leq 3$. In this section, we show that MinDMC is NP-hard even if $|\Sigma| = 2$ and $d = 4$. Thus, we obtain a complexity dichotomy regarding d for cases where the input matrix is binary.

Theorem 4.20. MinDMC *is NP-hard even if* $|\Sigma| = 2$ *and* $d \geq 4$.

Proof. Hermelin and Rozenberg [HR15] proved that MinRMC is NP-hard even for $|\Sigma| = 2$ and $d = 2$. We take advantage of this result to give a reduction from MinRMC.

Let $I = (\mathbf{S}, d = 2)$ be an instance of MinRMC where $\mathbf{S} \in \{0, 1, *\}^{n \times \ell}$. Let $\mathbf{S}' \in \{0, 1, *\}^{(n+1) \times (\ell+2)}$ be a matrix such that

- $\mathbf{S}'[i]$ is obtained by appending 00 to $\mathbf{S}[i]$ for each $i \in [n]$.
- $\mathbf{S}[n+1]$ is obtained by appending 11 to $*^n$.

We claim that I is a **Yes** instance if and only the MinDMC instance $I' = (\mathbf{S}', 4)$ is a **Yes** instance.

($\Rightarrow$) Let $v \in \{0,1\}^{\ell}$ be a solution of I. Let $\mathbf{T}'$ be a completion of $\mathbf{S}'$ such that $\mathbf{T}'[i, [\ell]] = \mathbf{S}'[i] \oplus v$ for each $i \in [n+1]$. Note that $\delta(\mathbf{T}'[i], \mathbf{T}'[n+1]) = \delta(\mathbf{S}[i], v) + 2 \leq 4$ for each $i \in [n]$. Also note that $\delta(\mathbf{T}'[i], \mathbf{T}'[i']) = \delta(\mathbf{S}[i], \mathbf{S}[i']) \leq \delta(v, \mathbf{S}[i]) + \delta(v, \mathbf{S}[i']) \leq 4$ for each $i, i' \in [n]$ by the triangle inequality.

($\Leftarrow$) Let $\mathbf{T}'$ be a solution of I' and let $v = \mathbf{T}'[n+1, [\ell]]$. It is easy to see that $\delta(\mathbf{S}[i], v) = \delta(\mathbf{T}'[i], \mathbf{T}'[n+1]) - 2 \leq 2$ for each $i \in [n]$. □

4.3 Parameter k

We consider the maximum number k of missing entries in any row as a parameter. Observe that MinDMC becomes trivial when $k = 0$. It implies that there is no missing entry in the input matrix $\mathbf{S}$. One can verify whether $\delta(\mathbf{S}) \leq d$ in $O(n^2\ell)$ time. We show in Section 4.3.1 that MinDMC can be solved in the same running time $O(n^2\ell)$ when $k = 1$. On the other hand, MinDMC turns out to be NP-hard even if $|\Sigma| = 2$ and $k = 2$ as we will see in Section 4.3.2.

4.3.1 Polynomial-time algorithm for the case $k = 1$

We show that MinDMC can be solved in polynomial time via a reduction to 2-SAT. As in Theorem 3.2, we will use the efficient encoding $C_{\leq 1}$ of the at-most-one constraint (the definition is given in Section 2.3).

Theorem 4.21. MinDMC *can be solved in time* $O((|\Sigma| + \ell) \cdot n^2)$ *when* $k = 1$.

Proof. First, we find the distances $\delta(\mathbf{S}[i], \mathbf{S}[i'])$ for each $i, i' \in [n]$ in time $O(n^2\ell)$. We immediately return **No** if there exist row indices $i, i' \in [n]$ such that $\delta(\mathbf{S}[i], \mathbf{S}[i']) > d$. Suppose that there exists $i \in [n]$ such that $\mathbf{S}[i]$ is a row vector with no missing entry. Then, we remove the row vector $\mathbf{S}[i]$ after filling the missing entry (if any) of $\mathbf{S}[i']$ by the entry of $\mathbf{S}[i]$ in the corresponding position for each $i' \in [n]$ where $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d$. We do so as long as there is a row vector that contains no missing entry. Note that it takes $O(n\ell^2)$ time as we spend $O(n\ell)$ time for each row vector. Henceforth, we assume that $\delta(\mathbf{S}[i], \mathbf{S}[i']) \leq d$ for any $i, i' \in [n]$ and that every row vector has exactly one missing entry.

Let $p_i \in [\ell]$ be such that $\mathbf{S}[i, p_i] = *$ for each $i \in [n]$. In order to find a solution matrix, we must find values $\mathbf{T}[i, p_i]$ that satisfy the following constraints:

- $(\mathbf{T}[i, p_i] = \mathbf{T}[i', p_i]) \vee (\mathbf{T}[i, p_{i'}] = \mathbf{T}[i', p_{i'}])$ for each $i, i' \in [n]$ such that $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d - 1$ and $p_i \neq p_{i'}$.
- $(\mathbf{T}[i, p_i] = \mathbf{T}[i', p_i]) \wedge (\mathbf{T}[i, p_{i'}] = \mathbf{T}[i', p_{i'}])$ for each $i, i' \in [n]$ such that $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d$ and $p_i \neq p_{i'}$.
- $(\mathbf{T}[i, p_i] = \mathbf{T}[i', p_{i'}])$ for each $i, i' \in [n]$ such that $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d$ and $p_i = p_{i'}$.

Note that row indices $i, i' \in [n]$ with $\delta(\mathbf{S}[i], \mathbf{S}[i']) \leq d - 2$ yield no constraint. This is because, regardless of the choice of $\mathbf{T}[i, p_i]$ and $\mathbf{T}[i', p_{i'}]$, we have $\delta(\mathbf{T}[i], \mathbf{T}[i']) \leq d$. We show that these constraints can be encoded in a 2-CNF formula.

For each $i \in [n]$, we introduce a variable $x_{i,\sigma}$ for each $\sigma \in \Sigma$. The intended meaning of $x_{i,\sigma}$ is to have $\mathbf{T}[i, p_i] = \sigma$ when $x_{i,\sigma}$ is true. Let $X_i = \{x_{i,\sigma} \mid \sigma \in \Sigma\}$. We let $\phi = \phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4$:

- First, we define

$$\phi_1 = \bigwedge_{i \in [n]} C_{\leq 1}(X_i).$$

  This ensures that at most one of the variables in X_i becomes true.

- The formula ϕ_2 ensures that the number of differences between $\mathbf{S}[i]$ and $\mathbf{S}[i']$ that columns p_i and $p_{i'}$ are going to induce is at most one if $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d - 1$:

$$\phi_2 = \bigwedge_{\substack{i,i' \in [n] \\ \delta(\mathbf{S}[i],\mathbf{S}[i'])=d-1 \\ p_i \neq p_{i'}}} (x_{i,\mathbf{S}[i',p_i]} \vee x_{i',\mathbf{S}[i,p_{i'}]}).$$

- We define the formula ϕ_3 which only consists of singleton clauses:

$$\phi_3 = \bigwedge_{\substack{i,i' \in [n] \\ \delta(\mathbf{S}[i],\mathbf{S}[i'])=d \\ p_i \neq p_{i'}}} (x_{i,\mathbf{S}[i',p_i]}) \wedge (x_{i',\mathbf{S}[i,p_{i'}]}).$$

  This ensures that $\mathbf{T}[i]$ and $\mathbf{T}[i']$ match on columns p_i and $p_{i'}$ if $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d$ and $p_i \neq p_{i'}$.

- Finally, we define the formula ϕ_4, which will guarantee that $\mathbf{T}[i, p_i] = \mathbf{T}[i', p_i]$ if $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d$ and $p_i = p_{i'}$:

$$\phi_4 = \bigwedge_{\substack{i,i' \in [n] \\ \delta(\mathbf{S}[i],\mathbf{S}[i'])=d \\ p_i = p_{i'}}} \bigwedge_{\sigma \in \Sigma} (x_{i,\sigma} \vee \neg x_{i',\sigma}) \wedge (\neg x_{i,\sigma} \vee x_{i',\sigma}).$$

Altogether ϕ contains $O(|\Sigma| \cdot n)$ variables and $O(|\Sigma| \cdot n^2)$ clauses. Next, we show the correctness of the reduction.

($\Rightarrow$) Suppose that there exists a completion $\mathbf{T} \in \Sigma^{n \times \ell}$ of $\mathbf{S}$ such that $\delta(\mathbf{T}) \leq d$. For each $i \in [n]$ and each $\sigma \in \Sigma$, we set $x_{i,\sigma} = \chi[\mathbf{T}[i, p_i] = \sigma]$. Note that this assigns 1 to exactly one variable in X_i and hence satisfies ϕ_1. By construction, this truth assignment satisfies ϕ_2, ϕ_3, ϕ_4 as well.

($\Leftarrow$) Suppose that there exists a satisfying truth assignment φ. We will construct a matrix $\mathbf{T} \in \Sigma^{n \times \ell}$ such that $\delta(\mathbf{T}) \leq d$. For this construction, it is sufficient to specify the value of $\mathbf{T}[i, p_i]$ for each $i \in [n]$. Let I^* denote the row indices i such that $\varphi(x_{i,\sigma}) = 0$ for all $\sigma \in \Sigma$. Observe that for each $i \in [n]$, there exists at most one variable $x_{i,\sigma} \in X_i$ such that $\varphi(x_{i,\sigma}) = 1$ in order to satisfy ϕ_1. For each $i \in [n] \setminus I^*$, it follows that there exists exactly one $\sigma \in \Sigma$ such that $\varphi(x_{i,\sigma}) = 1$. We fix an arbitrary character $\sigma^* \in \Sigma$. If $i \in I^*$, then we set $\mathbf{T}[i, p_i] = \sigma^*$. Otherwise, we set $\mathbf{T}[i, p_i] = \sigma$ where σ is a character such that $\varphi(x_{i,\sigma}) = 1$. We claim that $\delta(\mathbf{T}) \leq d$:

- For each $i, i' \in [n]$ such that $\delta(\mathbf{S}[i], \mathbf{S}[i']) \leq d - 2$, we have

$$\delta(\mathbf{T}[i], \mathbf{T}[i']) = \delta(\mathbf{S}[i], \mathbf{S}[i']) + \delta_{\{p_i, p_{i'}\}}(\mathbf{T}[i], \mathbf{T}[i']) \leq d.$$

- Suppose that $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d - 1$. If both $i \in I^*$ and $i' \in I^*$ hold, then the clause $(x_{i,\mathbf{S}[i',p_i]} \vee x_{i,\mathbf{S}[i,p_{i'}]})$ of ϕ_2 evaluates to false in φ. Thus, either $i \notin I^*$ or $i' \notin I^*$ must hold. Without loss of generality, assume that $i \notin I^*$. It gives us that $\mathbf{T}[i, p_i] = \mathbf{S}[i', p_i] = \mathbf{T}[i', p_i]$. Hence,

$$\delta(\mathbf{T}[i], \mathbf{T}[i']) = \delta(\mathbf{S}[i], \mathbf{S}[i]) + \delta(\mathbf{T}[i, p_i], \mathbf{T}[i', p_i]) + \delta(\mathbf{T}[i, p_{i'}], \mathbf{T}[i', p_{i'}]) \leq d.$$

- Suppose that $\delta(\mathbf{S}[i], \mathbf{S}[i']) = d$. If $p_i \neq p_{i'}$, then we have $\mathbf{T}[i, p_i] = \mathbf{T}[i', p_i]$ and $\mathbf{T}[i', p_{i'}] = \mathbf{T}[i, p_{i'}]$ because ϕ_3 contains singleton clauses $(x_{i,\mathbf{S}[i',p_i]})$ and $(x_{i',\mathbf{S}[i,p_{i'}]})$. Hence, $\delta(\mathbf{T}[i], \mathbf{T}[i']) = d$.

  Now suppose that $p_i = p_{i'}$. We show that $\mathbf{T}[i, p_i] = \mathbf{T}[i', p_{i'}]$. If $i \notin I^*$, then there exists a symbol $\sigma_i \in \Sigma$ such that $\varphi(x_{i,\sigma_i}) = 1$. Since ϕ_4 contains a clause $(\neg x_{i,\sigma_i} \vee x_{i',\sigma_i})$, it follows that $\varphi(x_{i',\sigma_i}) = 1$. Thus, we have $\mathbf{T}[i, p_i] = \mathbf{T}[i', p_{i'}] = \sigma_i$. On the other hand, suppose that $i \in I^*$. If there exists $\sigma'_i \in \Sigma$ such that $\varphi(x_{i',\sigma'_i}) = 1$, then the truth assignment φ does not satisfy the clause $(x_{i,\sigma'_i} \vee \neg x_{i',\sigma'_i})$ in ϕ_4. Thus, we infer that $i' \in I^*$. By construction, this means that $\mathbf{T}[i, p_i] = \mathbf{T}[i', p_{i'}] = \sigma^*$. We have $\mathbf{T}[i, p_i] = \mathbf{T}[i', p_{i'}]$ in both cases. Hence, we have $\delta(\mathbf{T}[i], \mathbf{T}[i']) = d$.

Recall that ϕ uses $O(|\Sigma| \cdot n)$ variables and $O(|\Sigma| \cdot n^2)$ clauses. Since 2-SAT can be solved in linear time [APT79], we obtain a procedure that solves MinDMC in the claimed time when $k = 1$. □

We remark that the quadratic dependence on n in the running time of the algorithm of Theorem 4.21 is probably inevitable. To conditonally prove this, we will use the Orthogonal Vectors conjecture, which states that Orthogonal Vectors cannot be solved in time $O(n^{2-\delta} \cdot \ell^c)$ for any $\delta, c > 0$.

Orthogonal Vectors
Input: Sets $\mathcal{U}, \mathcal{V}$ of row vectors in $\{0,1\}^\ell$ with $|\mathcal{U}| = |\mathcal{V}| = n$.
Question: Are there row vectors $u \in \mathcal{U}$ and $v \in \mathcal{V}$ such that $u[j] \cdot v[j] = 0$ holds for all $j \in [\ell]$?

We claim that the diameter of a matrix cannot be computed in time $O(n^{2-\delta} \cdot \ell^c)$ assuming the Orthogonal Vectors. It is widely known that the SETH implies the Orthogonal Vectors conjecture [Gao+19]. Hence, our claim implies that there is no algorithm for computing the diameter of a matrix in time $O(n^{2-\delta} \cdot \ell^c)$ for any $\delta, c > 0$, unless the SETH break. Let $u_1, \ldots, u_n, v_1, \ldots, v_n \in \{0,1\}^\ell$ be row vectors. Consider the matrix $\mathbf{T} \in \{0,1\}^{2n \times 6\ell}$ where

$$\mathbf{T}[i, [3j-2, 3j]] = \begin{cases} 001 & \text{if } j \leq \ell \text{ and } u_i[j] = 0 \\ 111 & \text{if } j \leq \ell \text{ and } u_i[j] = 1 \\ 000 & \text{otherwise} \end{cases}$$

$$\mathbf{T}[n+i, [3j-2, 3j]] = \begin{cases} 010 & \text{if } j \leq \ell \text{ and } v_i[j] = 0 \\ 111 & \text{if } j \leq \ell \text{ and } v_i[j] = 1 \\ 111 & \text{otherwise} \end{cases}$$

for each $i \in [n]$ and $j \in [2\ell]$. It is easy to see that there are $i, i' \in [n]$ such that u_i and $v_{i'}$ are orthogonal if and only if $\delta(\mathbf{T}) = 5\ell$ (see Figure 4.5 for an illustration). It follows that MinDMC cannot be solved in time $O(n^{2-\delta} \cdot \ell^c)$ for any $\delta, c > 0$ unless the SETH breaks.

$$\mathbf{T} = \left[\begin{array}{c|c|c|c} 001 & 111 & 001 & 000000000 \\ 111 & 001 & 001 & 000000000 \\ \hline 111 & 111 & 010 & 111111111 \\ 111 & 010 & 111 & 111111111 \end{array}\right]$$

Figure 4.5: An illustration of the reduction from Orthogonal Vectors, where $\mathcal{U} = \{010, 110\}$ and $\mathcal{V} = \{110, 101\}$.

4.3.2 NP-hardness for the case $k = 2$

In this section we prove that MinDMC is NP-hard even if $|\Sigma| = 2$ and $k = 2$. MinDMC asks for a matrix minimizing the pairwise Hamming distance among all row vectors. In designing a reduction for a hardness proof, it is inconvenient if one has to minimize Hamming distances of all pairs (rather than certain pairs). To lift this constraint, we introduce a matrix to increase the distance of one specific pair of row vectors relative to all other pairs.

Let $n \in \mathbb{N}$ with $n \geq 3$. First let us define a binary matrix $\mathbf{A}^n \in \{0,1\}^{n\times(2n-1)}$ as follows:

$$\mathbf{A}^n = \left[\begin{array}{ccccccccccc} 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 1 & & & & & & & & \\ 1 & 1 & 1 & & & & & & & & \\ & \vdots & & & & \mathbf{I} & & & & \mathbf{I} & \\ 1 & 1 & 1 & & & & & & & & \end{array}\right],$$

where $\mathbf{I}$ is an $(n-2)\times(n-2)$ identity matrix. Note that $\delta(\mathbf{A}^n[i], \mathbf{A}^n[i']) = 2$ if $(i, i') = (1, 2)$ and $\delta(\mathbf{A}^n[i], \mathbf{A}^n[i']) = 4$ otherwise for all $i < i' \in [n]$. We also define the matrix $\mathbf{A}^n_{i,i'}$ obtained from $\mathbf{A}^n$ by swapping the row vectors $\mathbf{A}^n[1]$ (and $\mathbf{A}^n[2]$) with $\mathbf{A}^n[i]$ (and $\mathbf{A}^n[i']$, respectively) for each $i < i' \in [n]$. Then the matrix $\mathbf{A}^n_{i,i'}$ is a matrix in which the distance between the i-th and i'-th row vectors are exactly two smaller than all other pairs. Now we use the matrix $\mathbf{A}^n_{i,i'}$ to obtain a binary matrix in which the distance of a certain pair of row vectors is exactly two greater than all others. We define $\mathbf{B}^n_{i,i'} \in \{0,1\}^{n\times\ell}$ with $\ell = (\binom{n}{2} - 1)(2n-1)$ as the matrix obtained by horizontally stacking $\binom{n}{2} - 1$ matrices $\mathbf{A}^n_{h,h'}$ for all $h < h' \in [n]$ with $(h, h') \neq (i, i')$:

$$\mathbf{B}^n_{i,i'} = \left[\mathbf{A}^n_{1,1} \cdots \mathbf{A}^n_{1,n} \cdots \mathbf{A}^n_{i,i+1} \cdots \mathbf{A}^n_{i,i'-1} \mathbf{A}^n_{i,i'+1} \cdots \mathbf{A}^n_{i,n} \cdots \mathbf{A}^n_{n-1,n}\right]$$

Observe that $\delta(\mathbf{B}^n_{i,i'}[i], \mathbf{B}^n_{i,i'}[i']) = 4\cdot(\binom{n}{2}-1) = 2n(n-1)-4$, since $\delta(\mathbf{A}^n_{i,i'}[h], \mathbf{A}^n_{i,i'}[h']) = 4$ for all $h < h' \in [n]$ with $(h, h') \neq (i, i')$. Note also that for each $h < h' \in [n]$ with $(h, h') \neq (i, i')$, we have $\delta(\mathbf{B}^n_{i,i'}[h], \mathbf{B}^n_{i,i'}[h']) = 2n(n-1)-6$ because the distance between $\mathbf{A}^n_{i,i'}[\widetilde{h}]$ and $\mathbf{A}^n_{i,i'}[\widetilde{h'}]$ is 4 for every $\widetilde{h} < \widetilde{h'} \in [n]$ except that it is smaller by two for the pair $\mathbf{A}^n_{i,i'}[i]$ and $\mathbf{A}^n_{i,i'}[i']$.

We remark that the construction of $\mathbf{B}^n_{i,i'}$ is optimal in the sense that there is no binary matrix with at least four rows where a pair of row vectors has distance one greater than all

others. It follows from an observation that the summed distance $\delta(u,v)+\delta(v,w)+\delta(w,u)$ of binary vectors $u,v,w \in \{0,1\}^\ell$ is always even. Assume without loss of generality that $v = 0^\ell$. Then, we can rewrite the distance as $\delta(u,v) = |P_1(u)| = |P_1(u) \cap P_1(u)| + |P_1(u) \setminus P_1(w)|$, $\delta(v,w) = |P_1(w)| = |P_1(w) \cap P_1(u)| + |P_1(w) \setminus P_1(u)|$, and $\delta(w,u) = |P_1(u) \triangle P_1(w)|$. The addition of these equations yields $\delta(u,v) + \delta(v,w) + \delta(w,u) = 2(|P_1(u) \cap P_1(w)| + |P_1(u) \triangle P_1(w)|)$, showing that it is an even number. Let $\mathbf{B} \in \{0,1\}^{n \times \ell}$ be a binary matrix with $n \geq 4$ such that $\delta(\mathbf{B}[1],\mathbf{B}[2]) = d+1$ and $\delta(\mathbf{B}[i],\mathbf{B}[i']) = d$ for all $i < i' \in [n]$ except $(i,i') = (1,2)$. Since the pairwise distances of $\mathbf{B}[1],\mathbf{B}[3],\mathbf{B}[4]$ are all equal to d, it follows that d is even. This however means that the sum over pairwise distances of $\mathbf{B}[1],\mathbf{B}[2],\mathbf{B}[3]$ is an odd number $3d+1$. This shows that our matrix $\mathbf{B}^n_{i,i'}$ is the best attainable.

The matrix $\mathbf{B}^n_{i,i'}$ will play a crucial role in the following NP-hardness proof of MinDMC for the case $|\Sigma| = 2$ and $k = 2$.

Theorem 4.22. MinDMC *is NP-hard even if* $|\Sigma| = 2$ *and* $k = 2$.

Proof. Our proof is based on a reduction from 3-SAT. We divide our proof into two parts as follows. We first provide a set $\mathcal{C}$ of incomplete matrices and we prescribe rules under which the matrices of $\mathcal{C}$ are completed. We prove that the given 3-CNF formula is satisfiable if and only if the matrices $\mathcal{C}$ can be completed under those rules. We then show that one can construct in polynomial time a single incomplete matrix $\mathbf{S}$ containing each matrix in $\mathcal{C}$ as a submatrix, such that $\mathbf{S}$ admits a completion of diameter at most d if and only if the completions to $\mathcal{C}$ according to the rules are feasible. We are going to exploit the matrix $\mathbf{B}^n_{i,i'}$ described above for this construction.

Part I. Let ϕ be an instance of 3-SAT with clauses $C_0,\ldots,C_{m-1}$. We assume that there are exactly three literals of distinct variables in each clause. We define the following matrix for each clause

$$\mathbf{C}_i = \begin{bmatrix} l_i^1 & & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & l_i^2 & & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & l_i^3 & & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & c_i & \end{bmatrix}.$$

Here we use l_i^1, l_i^2, l_i^3, c_i to represent two missing entries for notational purposes. Note that the matrices $\mathbf{C}_i$ are identical for all $i \in [0, m-1]$. We will prove that ϕ is satisfiable if and only if it is possible to complete matrices $\mathcal{C} = \{\mathbf{C}_i \mid i \in [0,m-1]\}$ satisfying the following constraints:

(C1) The missing entries l_i^j are filled by 00 or 11 for each $i \in [0,m-1]$ and $j \in [3]$.

(C2) The missing entries c_i are filled by 00, 01, or 10 for each $i \in [0,m-1]$.

(C3) If the missing entries c_i are filled by 00 (01, 10), then l_i^1 (l_i^2, l_i^3, respectively) are filled by 11 for each $i \in [0,m-1]$.

(C4) Let $\mathcal{Z}$ be a set such that $(i,j,i',j') \in \mathcal{Z}$ if and only if the j-th literal in C_i and the j'-th literal in $C_{i'}$ are the same variable of opposite sign for each $i < i' \in [0,m-1]$ and $j,j' \in [3]$. If $(i,j,i',j') \in \mathcal{Z}$, then either l_i^j or $l_{i'}^{j'}$ is filled by 00.

Note that one has three choices for the completion of c_i by (C2). The intuitive idea is that the completion of c_i dictates which literal in the clause C_i is satisfied. Then, one can obtain a satisfying truth assignment for ϕ, as we shall see in the following claim.

Claim 1. The formula ϕ is satisfiable if and only if the matrices $\mathcal{C}$ can be completed according to (C1) to (C4).

Proof. ($\Rightarrow$) If there exists a truth assignment φ satisfying ϕ, then at least one literal in the clause C_i evaluates to true for each $i \in [0, m-1]$. We choose an arbitrary number $l_i \in [3]$ such that the l_i-th literal of C_i is satisfied in φ for each $i \in [0, m-1]$. For each $i \in [0, m-1]$ we complete the matrix C_i as follows:

- If $l_i = 1$, then the missing entries c_i, l_i^1, l_i^2, l_i^3 are filled by 00, 11, 00, 00, respectively.
- If $l_i = 2$, then the missing entries c_i, l_i^1, l_i^2, l_i^3 are filled by 01, 00, 11, 00, respectively.
- If $l_i = 3$, then the missing entries c_i, l_i^1, l_i^2, l_i^3 are filled by 10, 00, 00, 11, respectively.

It is easy to verify that the first three constraints (C1) to (C3) are fulfilled. We claim that the last constraint (C4) is also satisfied. Suppose that there exist $i < i' \in [0, m-1]$ and $j, j' \in [3]$ such that the j-th (j'-th) literal in C_i ($C_{i'}$, respectively) is x ($\neg x$, respectively) for some variable x (or its negation), and both l_i^j and $l_{i'}^{j'}$ are filled by 11. Then it follows from our completion of $\mathcal{C}$ that $l_i = j$ and $l_{i'} = j'$, meaning that φ satisfies both x and $\neg x$ (a contradiction).

($\Leftarrow$) For each $i \in [m]$ and $j \in [3]$ where l_i^j is filled by 11, we construct a truth assignment such that the j-th literal of C_i is satisfied. No variable is given opposing truth values by such a truth assignment because of constraint (C4). It also satisfies every clause: Otherwise there exists an integer $i \in [m]$ such that all l_i^1, l_i^2, l_i^3 are completed by 00 due to (C1). Thus,we will utilize to we arrive at a contradiction because constraints (C2) and (C3) imply that at least one of l_i^1, l_i^2, l_i^3 is filled by 11. □

Part II. We provided the matrices $\mathcal{C}$ as well as the constraints on the completion of $\mathcal{C}$ in the previous part. Here we describe how one obtains the matrix $\mathbf{S}$ such that $\mathbf{S}$ can be completed within diameter d if and only if $\mathcal{C}$ can be completed fulfilling (C1) to (C4). Let us define the following matrix as a starting point:

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}'_0 & & & O \\ & \mathbf{C}'_1 & & \\ & & \ddots & \\ O & & & \mathbf{C}'_{m-1} \end{bmatrix} \in \{0, 1, *\}^{11m \times 8m}.$$

Here $\mathbf{C}'_i$ is a matrix with 8 rows and 11 columns defined as follows for each $i \in [0, m-1]$:

$$\mathbf{C}'_i = \begin{bmatrix} l_i^1 & & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & l_i^2 & & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & l_i^3 & & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & c_i & \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \in \{0, 1, *\}^{11 \times 8}.$$

Note that the first four row vectors of $\mathbf{C}'_i$ are identical to the row vectors of $\mathbf{C}_i$ and note in particular that

- $\mathbf{C}'_i[5]$ and $\mathbf{C}'_i[6]$ can be obtained by completing the missing entries in $\mathbf{C}'_i[1]$ with 01 and 10, respectively.
- $\mathbf{C}'_i[7]$ and $\mathbf{C}'_i[8]$ can be obtained by completing the missing entries in $\mathbf{C}'_i[2]$ with 01 and 10, respectively.
- $\mathbf{C}'_i[9]$ and $\mathbf{C}'_i[10]$ can be obtained by completing the missing entries in $\mathbf{C}'_i[3]$ with 01 and 10, respectively.
- $\mathbf{C}'_i[11]$ can be obtained by completing the missing entries in $\mathbf{C}'_i[4]$ with 00.

Let $n = 11m$ be the number of rows in $\mathbf{C}$. We obtain a matrix $\mathbf{S}$ by appending $\mathbf{B}^n_{h,h'}$ horizontally $c_{h,h'}$ (where $c_{h,h'}$ is to be defined) times for each $h < h' \in [n]$. Consider $h < h' \in [n]$ such that $\lfloor (h-1)/11 \rfloor = \lfloor (h'-1)/11 \rfloor$ (note that $\mathbf{C}[h]$ contains a row vector of $\mathbf{C}'_i$ for $i = \lfloor (h-1)/11 \rfloor$ and hence both $\mathbf{C}[h]$ and $\mathbf{C}[h']$ contain a row vector of $\mathbf{C}'_i$). First we define $c_{h,h'}$ for $h < h' \in [n]$ with $\lfloor (h-1)/11 \rfloor = \lfloor (h'-1)/11 \rfloor$. We do the following for each $i \in \{0, \ldots, m-1\}$ (we remark that the values of $c_{h,h'}$ seem somewhat random at first glance but it will become clearer later in Observation 4.23 that those values are deliberately chosen):

- Let $c_{11i+j,11i+j'} = 8$ for each $(j, j') \in J_1$, where $J_1 = \{(1,5), (1,6), (2,7), (2,8), (3,9), (3,10)\}$.
- Let $c_{11i+j,11i+j'} = 8$ for each $(j, j') \in J_2$, where $J_2 = \{(4,11)\}$.
- Let $c_{11i+j,11i+j'} = 5$ for each $(j, j') \in J_3$, where $J_3 = \{(1,4), (2,4), (3,4)\}$.
- Let $c_{11i+j,11i+j'} = 0$ for each $j < j' \in [11]$ with $(j, j') \notin J_1 \cup J_2 \cup J_3$.

Let us also define $c_{h,h'}$ for which $\lfloor (h-1)/11 \rfloor < \lfloor (h'-1)/11 \rfloor$ holds. For each $i < i' \in [0, m-1]$ and $j, j' \in [11]$, let

$$c_{11i+j,11i'+j'} = \begin{cases} 7 - \lfloor \delta(\mathbf{C}[11i+j], \mathbf{C}[11i'+j'])/2 \rfloor & \text{if } (i, j, i', j') \in \mathcal{Z}. \\ 0 & \text{otherwise.} \end{cases}$$

Now $c_{h,h'}$ is determined for each $h < h' \in [n]$ and $\mathbf{S}$ can be obtained from $\mathbf{C}$ by appending $\mathbf{B}^n_{h,h'}$ horizontally $c_{h,h'}$ times.

In order to obtain a MinDMC instance, it still remains to provide the diameter bound d. Let $N = 2n(n-1)-6$ (recall that $\delta(\mathbf{B}^n_{h,h'}[i], \mathbf{B}^n_{h,h'}[i'])$ equals N if $(h,h') \neq (i,i')$ and $N+2$ otherwise) and let

$$d = \sum_{h<h' \in [n']} c_{h,h'} \cdot N + 17.$$

Again the second term 17 seems arbitrary but its intention will be clear soon. Observe that the pairwise row distance $\delta(\mathbf{S}[h], \mathbf{S}[h'])$ can be rewritten as follows for each $h < h' \in [n]$:

$$\begin{aligned}\delta(\mathbf{S}[h], \mathbf{S}[h']) &= \delta(\mathbf{C}[h], \mathbf{C}[h']) + c_{h,h'} \cdot (N+2) + \sum_{\substack{i<i' \in [n'],\\ (i,i') \neq (h,h')}} c_{h,h'} \cdot N \\ &= \delta(\mathbf{C}[h], \mathbf{C}[h']) + 2c_{i,i'} + d - 17.\end{aligned}$$

Hence, we can express each pairwise row distance of $\mathbf{S}$ in terms of d by plugging in the values of $\delta(\mathbf{C}[h], \mathbf{C}[h'])$ and $c_{h,h'}$.

Observation 4.23. *All of the following hold:*

- $\delta(\mathbf{S}[11i+j], \mathbf{S}[11i+j']) = 0 + 2 \cdot 8 + d - 17 = d - 1$ *for each* $i \in [0, m-1]$ *and* $(j,j') \in J_1$ *(cf. constraint (C1)).*
- $\delta(\mathbf{S}[11i+j], \mathbf{S}[11i+j']) = 0 + 2 \cdot 8 + d - 17 = d - 1$ *for each* $i \in [0, m-1]$ *and* $(j,j') \in J_2$ *(cf. constraint (C2)).*
- $\delta(\mathbf{S}[11i+j], \mathbf{S}[11i+j']) = 4 + 2 \cdot 5 + d - 17 = d - 3$ *for each* $i \in [0, m-1]$ *and* $(j,j') \in J_3$ *(cf. constraint (C3)).*
- *It holds that*

  $$\begin{aligned}&\delta(\mathbf{S}[11i+j], \mathbf{S}[11i'+j']) \\ &= \delta(\mathbf{C}[11+j], \mathbf{C}[11i'+j']) + 2 \cdot (7 - \lfloor \delta(\mathbf{C}[11+j], \mathbf{C}[11i'+j']/2) \rfloor) + d - 17 \\ &= \begin{cases} d-3 & \text{if } \delta(\mathbf{C}[11i+j], \mathbf{C}[11i'+j']) \text{ is even} \\ d-2 & \text{if } \delta(\mathbf{C}[11i+j], \mathbf{C}[11i'+j']) \text{ is odd.} \end{cases}\end{aligned}$$

  for each $i < i' \in [0, m-1]$ *and* $j, j' \in [11]$ *with* $(i,j,i',j') \in I$ *(cf. constraint (C4)).*
- *For each* $h < h' \in [n]$ *with* $c_{h,h'} = 0$*, we have*

  $$\delta(\mathbf{S}[h], \mathbf{S}[h']) = \delta(\mathbf{C}[h], \mathbf{C}[h']) + d - 17 \leq 12 + d - 17 = d - 5.$$

  Here the first inequality is due to

  $$\delta(\mathbf{C}[h], \mathbf{C}[h']) \leq \delta(\mathbf{C}[h], 0^{8m}) + \delta(\mathbf{C}[h'], 0^{8m}) \leq 12$$

  by the triangle inequality.

We use the observations above to prove the following claim.

Claim 2. The matrices in $\mathcal{C}$ can be completed under constraints (C1) to (C4) if and only $\mathbf{S}$ can be completed within diameter d.

Proof. ($\Rightarrow$) Let $\mathbf{T}$ be the matrix where the missing entries are filled as in the completion of $\mathcal{C}$. We show that $\delta(\mathbf{T}[h], \mathbf{T}[h']) \leq d$ for each $h < h' \in [n]$. We distinguish cases as in Observation 4.23.

- Suppose that $h = 11i + j$ and $h' = 11i + j'$ for $i \in \{0, \ldots, m-1\}$ and $(j, j') \in J_1$. Note that the missing entries l_i^j in $\mathbf{S}[h]$ are filled by 00 or 11 by (C1). Also note that $\mathbf{S}[h']$ has 01 or 10 in the corresponding positions. Hence, $\delta(\mathbf{T}[h], \mathbf{T}[h']) \leq \delta(\mathbf{S}[h], \mathbf{S}[h']) + 1 = d$.

- Suppose that $h = 11i + j$ and $h' = 11i + j'$ for $i \in \{0, \ldots, m-1\}$ and $(j, j') \in J_2$. Note that the missing entries c_i in $\mathbf{S}[h]$ are filled by 00, 01, or 11 by (C2). Also note that $\mathbf{S}[h']$ has 00 in the corresponding positions. Hence, $\delta(\mathbf{T}[h], \mathbf{T}[h']) \leq \delta(\mathbf{S}[h], \mathbf{S}[h']) + 1 = d$.

- Suppose that $h = 11i + j$ and $h' = 11i + j'$ for $i \in \{0, \ldots, m-1\}$ and $(j, j') \in J_3$. Note that $\mathbf{S}[h]$ has missing entries l_i^j and $\mathbf{S}[h']$ has missing entries c_i. Let x_1x_2 be the completion of c_i for $x_1, x_2 \in \{0, 1\}$. Suppose that $\mathbf{C}_i[j, 7] = 1 - x_1$ and $\mathbf{C}_i[j, 8] = 1 - x_2$. Then, l_i^j must be filled by 11 due to constraint (C3). Since $\mathbf{S}[h']$ has 11 in the corresponding positions, it follows that $\delta(\mathbf{T}[h], \mathbf{T}[h']) = \delta(\mathbf{S}[h], \mathbf{S}[h']) + 2 = d - 1$. Hence we can assume that $\mathbf{C}_i[j, 7] = x_1$ or $\mathbf{C}_i[j, 8] = x_2$ holds. It means that $\delta(\mathbf{T}[h], \mathbf{T}[h']) \leq \delta(\mathbf{S}[h], \mathbf{S}[h']) + 3 = d$.

- Suppose that $h = 11i + j$ and $h' = 11i' + j'$ for $i < i' \in \{0, \ldots, m-1\}$ and $(i, j, i', j') \in \mathcal{Z}$. Note that $\mathbf{S}[h]$ has missing entries l_i^j and $\mathbf{S}[h']$ has missing entries $l_{i'}^{j'}$. Also note that $\mathbf{S}[h]$ and $\mathbf{S}[h']$ have 00 where the other row vector has missing entries. Since either l_i^j or $l_{i'}^{j'}$ must be completed by 00 by (C4), we have $\delta(\mathbf{T}[h], \mathbf{T}[h']) = \delta(\mathbf{S}[h], \mathbf{S}[h']) \leq d - 2 + 2 = d$.

- Suppose that $h < h' \in [n]$ satisfy none of the above. Then we have $c_{h,h'} = 0$. Since every row vector of $\mathbf{S}$ contains at most two missing entries, we have $\delta(\mathbf{T}[h], \mathbf{T}[h']) \leq d - 5 + 2 \cdot 2 = d - 1$.

($\Leftarrow$) We complete the matrices in $\mathcal{C}$ in the same way as in the completion of $\mathbf{S}$. We examine each constraint (C1) to (C4)

- Suppose that the completion of l_i^1 is 01 (10) for some $i \in \{0, \ldots, m-1\}$. Then we have a contradiction because the distance between $\mathbf{S}[11i + 1]$ and $\mathbf{S}[11i + 6]$ ($\mathbf{S}[11i+5]$, respectively) is $\delta(\mathbf{S}[11i+1], S[11i+6]) + 2 = d+1$ ($\delta(\mathbf{S}[11i+1], S[11i+5]) + 2 = d + 1$, respectively). We can make analogous arguments for l_i^2 and l_i^3 as well.

- Suppose that the completion of c_i is 11 for some $i \in \{0, \ldots, m-1\}$. Then we have a contradiction because $\delta(\mathbf{S}[11i + 4], \mathbf{S}[11i + 11]) + 2 = d + 1$.

- Suppose that the completion of c_i is 00 (01, 10) and the completion of l_i^1 (l_i^2, l_i^3, respectively) is 11. Then we have a contradiction because the distance between $\mathbf{S}[11i+4]$ and $\mathbf{S}[11i+1]$ ($\mathbf{S}[11i+2]$, $\mathbf{S}[11i+3]$, respectively) is $d+1$.

- Suppose that the completion of l_i^j and $l_{i'}^{j'}$ for some $i \in \{0, \ldots, m-1\}$ and $j, j' \in [3]$ with $(i, j, i', j') \in \mathcal{Z}$. Then we have a contradiction because the distance between $\mathbf{S}[11i+j]$ and $\mathbf{S}[11i+j']$ is $\delta(\mathbf{S}[11i+j], \mathbf{S}[11i+j']) + 4 \geq d+1$.

This completes the proof of the claim. □

Hence we can conclude that ϕ is satisfiable if and only if the MinDMC instance $(\mathbf{S}, d)$ is a **Yes** instance. Moreover, the matrix $\mathbf{S}$ has $n = 11m \in O(m)$ rows and $O(m^5)$ columns and it can be constructed in polynomial time. This shows that MinDMC is NP-hard even if $|\Sigma| = 2$ and $k = 2$. □

4.4 Concluding remarks

We studied the computational complexity of the matrix completion problem, where the objective is to minimize the maximum pairwise row distance. There are several open questions to be discussed. It is known that the clustering variant Minimum Diameter Clustering Matrix Completion (see Section 1.1 for the definition) of MinDMC can be solved in polynomial time when the number c of clusters is two and the matrix is complete [GJL04]. Hence, a natural question arises whether our tractability results can be extended to this variant as well.

In Section 4.1, we provided a fixed-parameter algorithm with respect to the number n of rows, via a reduction to ILP (Integer Linear Programming). Our reduction involves exponentially many variables, and as a consequence the dependence on n in the running time is doubly exponential. One promising method for speeding up the algorithm would be to rely on the machinery of *n-fold* ILP (notably it has been used to obtain a more efficient fixed-parameter algorithm for MinRMC [KKM17]), rather than vanilla ILP.

The next open question concerns the number ℓ of columns. Although MinDMC is trivially fixed-parameter tractable with respect to ℓ for constant alphabet size $|\Sigma|$, the parameterized complexity regarding this parameter is completely unknown in the case of unbounded alphabet size. Even the existence of an algorithm with running time $n^{\ell^{O(1)}}$ (it would mean that MinDMC is XP with respect to ℓ) is open; It is possible that MinDMC is NP-hard for some constant value of ℓ.

We have some open questions regarding the parameter d and k as well. We proved that there is a linear-time algorithm solving MinDMC when $|\Sigma|$ is a constant and $d = 2$ (Theorem 4.12). In the case of unbounded alphabet size and $d = 2$, however, the running time of our algorithm is polynomial but super-linear (Theorem 4.14). A natural question is whether MinDMC with $d = 2$ can be solved in linear time even for arbitrary alphabet size. We are optimistic that more sophisticated case analysis in Lemma 4.13 would yield a linear-time algorithm. For the case of $|\Sigma| = 2$ and $d = 3$, we obtained a polynomial-time algorithm (Theorem 4.19). In both cases of $d = 2$ and $d = 3$, we obtained polynomial-time algorithms by creating *win-win* situations by using the notions of *Δ-systems*: If ℓ

is sufficiently large, then the matrix must fulfill certain properties that we can exploit to obtain an efficient algorithm. Otherwise, a simple brute-force algorithm will do, if $|\Sigma|$ is a constant. For the case $d = 3$, we showed that its boundary of the two scenarios lies between $\ell = 9$ and $\ell = 10$ (Lemma 4.18). In order to settle another question we left open—whether MinDMC is tractable for the case $|\Sigma| \geq 3$ and $d = 3$, one has to determine the (in)tractability of MinDMC for the case $\ell = 9$ and $d = 3$. This means that one has to answer the open question regarding ℓ, at least partially. Consequently, let us remark that settling fixed-parameter tractability with respect to ℓ seems to be a pressing issue.

Finally, let us pose the last open question of this chapter: Is MinDMC fixed-parameter tractable with respect to $d + k$ for binary alphabet? Note that $k \in \theta(\ell)$ in our NP-hardness proof for the case $d = 4$ (Theorem 4.20). Also note that $d \in \theta(n^5)$ at worst in the reduction of Theorem 4.22. Recall that we have made use of the matrix $\mathbf{B}^n_{i,i'}$ to adjust the pairwise row distances in the proof of Theorem 4.22. However, $\mathbf{B}^n_{i,i'}$ presumably does not help in proving the parameterized intractability, since $\delta(\mathbf{B}^n_{i,i'}) \in \theta(n^2)$. We remark that we have thus far failed to prove the NP-hardness for a constant value of k without relying on $\mathbf{B}^n_{i,i'}$. Hence, it is very well possible that MinDMC with $|\Sigma| = 2$ is fixed-parameter tractable with respect to $d+k$. One plausible way may be to aim for a win-win situation. Can we perhaps utilize the fixed-parameter algorithms for MinRMC (Theorems 3.8 and 3.11) when the input matrix is sufficiently large?

Chapter 5

Conclusion

We studied three matrix completion problems: MinRMC, MinLRMC, and MinDMC. We used a multivariate approach for computing the optimal solution. In particular, our focus was on the distance bound d and the maximum number k of missing entries in any row vector.

First, we investigated MinRMC and MinLRMC. We completely settled the (parameterized) complexity regarding the parameter d and k. MinRMC and MinLRMC turned out to be polynomial-time solvable when $d = 1$ even for arbitrary alphabet size, whereas both of these problems are NP-hard when $d = 2$ for binary alphabet [HR15]. The special case of MinRMC with $k = 0$ is known as Closest String and NP-hard even if $|\Sigma| = 2$. Despite the hardness regarding d and k, the combined parameterization of d and k yields fixed-parameter tractability for MinRMC. Meanwhile, we proved MinLRMC is fixed-parameter tractable with respect to k alone.

Then, we studied MinDMC. Our main contributions for MinDMC is dichotomy results for the parameters d and k. We provided polynomial time algorithms for the case $|\Sigma| = 2$ and $d \leq 3$ and we proved that MinDMC is NP-hard when $|\Sigma| = 2$ and $d \geq 4$. Our polynomial-time algorithm is based on the theorem on Δ-systems by Deza [Dez73]. For the parameter k, we proved that MinDMC is polynomial-time solvable when $k = 1$ but is NP-hard when $k \geq 2$.

Future research. First and foremost, let us repeat the open questions we mentioned in Sections 3.4 and 4.4:

- Can we improve the running time for Theorem 3.11 by adapting more sophisticated algorithms for Closest String?
- Consider the variant of MinRMC in which some number t of row vectors (called *outliers*) can be excluded from the radius constraint. What is the parameterized complexity of this variant with $d + k + t$?
- Is MaxRMC (see Section 3.4 for the definition) fixed-parameter tractable with respect to d?
- Is there a more efficient fixed-parameter algorithm for MinDMC with respect to the number of rows, possibly using n-fold Integer Linear Programming instead?

- What is the parameterized complexity of MinDMC with respect to the number of columns?
- Can MinDMC be solved in linear time for arbitrary alphabet size when $d = 2$?
- Is MinDMC polynomial-time solvable when $d = 3$ and $|\Sigma| \geq 3$?
- Is MinDMC fixed-parameter tractable with respect to $d + k$ when $|\Sigma| = 2$?

We will conclude the thesis with further directions for future research. Despite the fact that matrix completion problems have ubiquitous applications, the combinatorial matrix completion problems were first studied from a parameterized complexity standpoint only very recently. Hence, there is still a plenty of room for the choice of problem variants and parameters to be explored.

Note that some problem variants have been already mentioned (a variant of MinRMC with outliers and MaxRMC). Another problem of interest is the clustering variant in which we would like to minimize the sum of pairwise distances within each cluster. So far our objectives for the matrix completion are all based on the Hamming distance measures. But there are other known distance measures such as such as edit distance and rank distance, swap distance, reversal distance, and rank distance [Din03], from each of which one can derive variants of MinRMC, MinLRMC, and MinDMC. We remark that for inspiration of more matrix completion problem variants, the reader may want to refer to a survey of NP-hard string problems by Bulteau et al. [Bul+14].

Finally, let us discuss other parameterization. In this work, we considered the parameters d and k, in addition to the parameters naturally inherent to the input (the number of rows and columns, and the alphabet size). Our motivation for investigating the complexity with respect to d was based on the assumption that each row vector is close to one another. For the parameter k, we considered the cases where the input matrix is almost complete. In some applications, however, the majority of entries in the input matrix may be unknown (this seems to be the case for the Netflix challenge). To express the sparsity of observed data points, we suggest the following parameters (or their combinations): (i) the maximum number of known entries in any row vector, (ii) the maximum number of known entries in any column vector, and (iii) the minimum number of rows and columns covering all known entries. However, one cannot simply determine which parameter is "the right" parameter of the problem, and there are plenty of fair choices for the parameter, as stated by Niedermeier [Nie10]. We leave as an open question for future research, what "the right" parameters are for the matrix completion problems.

Literature

[Ami+14] A. Amir, J. Ficler, L. Roditty, and O. S. Shalom. "On the efficiency of the Hamming c-centerstring problems". In: *Symposium on Combinatorial Pattern Matching*. Springer. 2014, pp. 1–10 (cit. on p. 13).

[APT79] B. Aspvall, M. F. Plass, and R. E. Tarjan. "A linear-time algorithm for testing the truth of certain quantified Boolean formulas". In: *Information Processing Letters* 8.3 (1979), pp. 121–123 (cit. on pp. 20, 52).

[Bas+18] M. Basavaraju, F. Panolan, A. Rai, M. S. Ramanujan, and S. Saurabh. "On the kernelization complexity of string problems". In: *Theoretical Computer Science* 730 (2018), pp. 21–31 (cit. on p. 29).

[Ben+97] A. Ben-Dor, G. Lancia, J. Perone, and R. Ravi. "Banishing bias from consensus sequences". In: *8th Annual Symposium on Combinatorial Pattern Matching (CPM '97)*. Springer. 1997, pp. 247–261 (cit. on p. 23).

[BL07] J. Bennett and S. Lanning. "The Netflix prize". In: *KDD Cup and Workshop*. Vol. 2007. 2007, p. 35 (cit. on p. 11).

[BM11] C. Boucher and B. Ma. "Closest string with outliers". In: *BMC Bioinformatics* 12.S-1 (2011), S55 (cit. on p. 34).

[BS18] L. Bulteau and M. L. Schmid. "Consensus strings with small maximum distance and small distance sum". In: *43rd International Symposium on Mathematical Foundations of Computer Science, (MFCS '18)*. 2018, 1:1–1:15 (cit. on p. 34).

[Bul+14] L. Bulteau, F. Hüffner, C. Komusiewicz, and R. Niedermeier. "Multivariate algorithmics for NP-Hard string problems". In: *Bulletin of the EATCS* 114 (2014) (cit. on p. 62).

[Cab+11] R. S. Cabral, F. D. la Torre, J. P. Costeira, and A. Bernardino. "Matrix completion for multi-label image classification". In: *25th Annual Conference on Neural Information Processing Systems (NIPS '11)*. 2011, pp. 190–198 (cit. on p. 11).

[Cab+15] R. S. Cabral, F. D. la Torre, J. P. Costeira, and A. Bernardino. "Matrix completion for weakly-supervised multi-label image classification". In: *IEEE Transactions on Pattern and Analysis and Machine Intelligence* 37.1 (2015), pp. 121–135 (cit. on p. 11).

[Che10] J. Chen. “A new SAT encoding of the at-most-one constraint”. In: *9th International Workshop on Constraint Modelling and Reformulation (ModRef '10)*. 2010 (cit. on p. 20).

[CMW12] Z. Chen, B. Ma, and L. Wang. “A three-string approach to the closest string problem”. In: *Journal of Computer and System Sciences* 78.1 (2012), pp. 164–178 (cit. on p. 34).

[CMW16] Z. Chen, B. Ma, and L. Wang. “Randomized fixed-parameter algorithms for the Closest String Problem”. In: *Algorithmica* 74.1 (2016), pp. 466–484 (cit. on p. 34).

[CP10] E. J. Candès and Y. Plan. “Matrix completion with noise”. In: *Proceedings of the IEEE* 98.6 (2010), pp. 925–936 (cit. on p. 11).

[CR12] E. J. Candès and B. Recht. “Exact matrix completion via convex optimization”. In: *Communications of the ACM* 55.6 (2012), pp. 111–119 (cit. on p. 11).

[CT10] E. J. Candès and T. Tao. “The power of convex relaxation: near-optimal matrix completion”. In: *IEEE Transactions on Information Theory* 56.5 (2010), pp. 2053–2080 (cit. on p. 11).

[CW11] Z. Chen and L. Wang. “Fast exact algorithms for the closest string and substring problems with application to the planted (ℓ, d)-Motif Model”. In: *IEEE/ACM Transactions Computational Biology and Bioinformatics* 8.5 (2011), pp. 1400–1410 (cit. on p. 34).

[Cyg+15] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015 (cit. on pp. 18, 20, 27, 40).

[Den+03] X. Deng, G. Li, Z. Li, B. Ma, and L. Wang. “Genetic design of drugs without side-effects”. In: *SIAM Journal on Computing* 32.4 (2003), pp. 1073–1090 (cit. on p. 23).

[Dez73] M. Deza. “Une propriété extrémale des plans projectifs finis dans une classe de codes équidistants”. In: *Discrete Mathematics* 6.4 (1973), pp. 343–352 (cit. on pp. 5, 7, 16, 40, 61).

[Dez74] M. Deza. “Solution d'un problème de Erdős-Lovász”. In: *Journal of Combinatorial Theory, Series B* 16.4 (1974), pp. 166–167 (cit. on p. 40).

[DF13] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Vol. 4. Springer, 2013 (cit. on p. 18).

[Din03] L. P. Dinu. “On the classification and aggregation of hierarchies with different constitutive elements”. In: *Fundamenta Informaticae* 55.1 (2003), pp. 39–50 (cit. on p. 62).

[Dop+93] J. Dopazo, A. Rodríguez, J. Sáiz, and F. Sobrino. “Design of primers for PCR amplification of highly variable genomes”. In: *Bioinformatics* 9.2 (1993), pp. 123–125 (cit. on p. 23).

[Eib+19] E. Eiben, R. Ganian, I. Kanj, S. Ordyniak, and S. Szeider. "On clustering incomplete data". In: *CoRR* abs/1911.01465 (2019). URL: http://arxiv.org/abs/1911.01465 (cit. on pp. 5, 7, 11–14, 34, 37, 38, 40).

[EIS75] S. Even, A. Itai, and A. Shamir. "On the complexity of time table and multi-commodity flow problems". In: *16th Annual Symposium on Foundations of Computer Science (FOCS '75)*. IEEE. 1975, pp. 184–193 (cit. on p. 20).

[ER60] P. Erdős and R. Rado. "Intersection theorems for systems of sets". In: *Journal of the London Mathematical Society* 1.1 (1960), pp. 85–90 (cit. on p. 12).

[FG06] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006 (cit. on p. 18).

[FL97] M. Frances and A. Litman. "On covering problems of codes". In: *Theory of Computer Systems* 30.2 (1997), pp. 113–119 (cit. on pp. 13, 15, 25, 28).

[Fro+16] V. Froese, R. van Bevern, R. Niedermeier, and M. Sorge. "Exploiting hidden structure in selecting dimensions that distinguish vectors". In: *Journal of Computer and System Sciences* 82.3 (2016), pp. 521–535 (cit. on pp. 16, 40, 41, 44).

[FT87] A. Frank and É. Tardos. "An application of simultaneous Diophantine approximation in combinatorial optimization". In: *Combinatorica* 7.1 (1987), pp. 49–65 (cit. on p. 19).

[Gan+18] R. Ganian, I. A. Kanj, S. Ordyniak, and S. Szeider. "Parameterized algorithms for the matrix completion problem". In: *35th International Conference on Machine Learning, (ICML '18)*. 2018, pp. 1642–1651 (cit. on pp. 5, 7).

[Gao+19] J. Gao, R. Impagliazzo, A. Kolokolova, and R. Williams. "Completeness for first-order properties on sparse structures with algorithmic applications". In: *ACM Transactions on Algorithms* 15.2 (2019), 23:1–23:35 (cit. on p. 52).

[GGN06] J. Gramm, J. Guo, and R. Niedermeier. "Parameterized intractability of distinguishing substring selection". In: *Theory of Computing Systems* 39.4 (2006), pp. 545–560 (cit. on p. 35).

[GJL04] L. Gąsieniec, J. Jansson, and A. Lingas. "Approximation algorithms for Hamming clustering problems". In: *Journal of Discrete Algorithms* 2.2 (2004), pp. 289–301 (cit. on p. 59).

[GNR03] J. Gramm, R. Niedermeier, and P. Rossmanith. "Fixed-parameter algorithms for closest string and related problems". In: *Algorithmica* 37.1 (2003), pp. 25–42 (cit. on pp. 17, 23, 24, 26, 28, 29, 34, 35, 38).

[HR15] D. Hermelin and L. Rozenberg. "Parameterized complexity analysis for the closest string with wildcards problem". In: *Theoretical Computer Science* 600 (2015), pp. 11–18 (cit. on pp. 5, 7, 14, 15, 23–26, 28, 29, 33, 34, 49, 61).

[IP01] R. Impagliazzo and R. Paturi. "On the complexity of k-SAT". In: *Journal of Computer and System Sciences* 62.2 (2001), pp. 367–375 (cit. on pp. 19, 20).

[IPZ01] R. Impagliazzo, R. Paturi, and F. Zane. "Which problems have strongly exponential complexity?" In: *Journal of Computer and System Sciences* 63.4 (2001), pp. 512–530 (cit. on p. 20).

[Ji+10] H. Ji, C. Liu, Z. Shen, and Y. Xu. "Robust video denoising using low rank matrix completion". In: *23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR '10)*. 2010, pp. 1791–1798 (cit. on p. 11).

[Juk11] S. Jukna. *Extremal Combinatorics: With Applications in Computer Science*. Springer Science & Business Media, 2011 (cit. on p. 40).

[Kan87] R. Kannan. "Minkowski's convex body theorem and integer programming". In: *Mathematics of Operations Research* 12.3 (1987), pp. 415–440 (cit. on p. 19).

[KKM17] D. Knop, M. Koutecký, and M. Mnich. "Combinatorial n-fold integer programming and applications". In: *25th Annual European Symposium on Algorithms, (ESA '17)*. 2017, 54:1–54:14 (cit. on pp. 15, 24, 25, 59).

[Kro67] M. R. Krom. "The decision problem for a class of first-order formulas in which all disjunctions are binary". In: *Mathematical Logic Quarterly* 13.1-2 (1967), pp. 15–20 (cit. on p. 20).

[Lan+03] J. K. Lanctôt, M. Li, B. Ma, S. Wang, and L. Zhang. "Distinguishing string selection problems". In: *Information and Computation* 185.1 (2003), pp. 41–55 (cit. on p. 23).

[LJ83] H. W. Lenstra Jr. "Integer programming with a fixed number of variables". In: *Mathematics of Operations Research* 8.4 (1983), pp. 538–548 (cit. on pp. 19, 37).

[LMS18] D. Lokshtanov, D. Marx, and S. Saurabh. "Slightly superexponential parameterized problems". In: *SIAM Journal on Computing* 47.3 (2018), pp. 675–702 (cit. on pp. 26, 27).

[Luc+91] K Lucas, M Busch, S Mössinger, and J. Thompson. "An improved microcomputer program for finding gene- or gene family-specific oligonucleotides suitable as primers for polymerase chain reactions or as probes". In: *Bioinformatics* 7.4 (1991), pp. 525–529 (cit. on p. 23).

[Luo+15] Y. Luo, T. Liu, D. Tao, and C. Xu. "Multiview matrix completion for multilabel image classification". In: *IEEE Transactions on Image Processing* 24.8 (2015), pp. 2355–2368 (cit. on p. 11).

[MS09] B. Ma and X. Sun. "More efficient algorithms for closest string and substring problems". In: *SIAM Journal on Computing* 39.4 (2009), pp. 1432–1443 (cit. on pp. 24, 28, 31, 32, 34).

[Ngu+19] L. T. Nguyen, J. Kim, S. Kim, and B. Shim. "Localization of IoT Networks via low-rank matrix completion". In: *IEEE Transactions on Communications* 67.8 (2019), pp. 5833–5847 (cit. on p. 11).

[Nie06] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006 (cit. on p. 18).

[Nie10] R. Niedermeier. "Reflections on multivariate algorithmics and problem parameterization". In: *27th International Symposium on Theoretical Aspects of Computer Science, (STACS '10)*. 2010, pp. 17–32 (cit. on pp. 14, 62).

[NS12] N. Nishimura and N. Simjour. "Enumerating neighbour and closest strings". In: *7th International Symposium on Parameterized and Exact Computation, (IPEC '12)*. Springer. 2012, pp. 252–263 (cit. on p. 34).

[Pee96] R. Peeters. "Orthogonal representations over finite fields and the chromatic number of graphs". In: *Combinatorica* 16.3 (1996), pp. 417–431 (cit. on p. 11).

[PH96] V. Proutski and E. C. Holmes. "Primer master: a new program for the design and analysis of PCR primers". In: *Bioinformatics* 12.3 (1996), pp. 253–255 (cit. on p. 23).

[PMP01] G. Pavesi, G. Mauri, and G. Pesole. "An algorithm for finding signals of unknown length in DNA sequences". In: *International Conference on Intelligent Systems for Molecular Biology (ISMB '01)*. 2001, pp. 207–214 (cit. on p. 23).

[PS00] P. A. Pevzner and S. Sze. "Combinatorial approaches to finding subtle signals in DNA sequences". In: *8th International Conference on Intelligent Systems for Molecular Biology (ISMB '00)*. 2000, pp. 269–278 (cit. on p. 23).

[Rec11] B. Recht. "A simpler approach to matrix completion". In: *Journal of Machine Learning Research* 12 (2011), pp. 3413–3430 (cit. on p. 11).

[RW13] L. Roditty and V. V. Williams. "Fast approximation algorithms for the diameter and radius of sparse graphs". In: *Symposium on Theory of Computing Conference (STOC '13)*. 2013, pp. 515–524 (cit. on p. 20).

[Sch86] R. Schmidt. "Multiple emitter location and signal parameter estimation". In: *IEEE transactions on antennas and propagation* 34.3 (1986), pp. 276–280 (cit. on p. 11).

[Sin05] C. Sinz. "Towards an optimal CNF encoding of boolean cardinality constraints". In: *International Conference on Principles and Practice of Constraint Programming*. Springer. 2005, pp. 827–831 (cit. on p. 20).

[SW93] H. Stamm-Wilbrandt. *Programming in propositional logic or reductions: Back to the roots (satisfiability)*. Technical Report, Universität Bonn, 1993 (cit. on p. 20).

[WZ09] L. Wang and B. Zhu. "Efficient algorithms for the closest string and distinguishing string selection problems". In: *3rd International Frontiers in Algorithmics Workshop (FAW '09)*. Springer. 2009, pp. 261–270 (cit. on p. 35).