A Parameterized Complexity Analysis of Combinatorial Feature Selection Problems*

Vincent Froese, René van Bevern, Rolf Niedermeier, and Manuel Sorge

Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany
{vincent.froese, rene.vanbevern, rolf.niedermeier,
manuel.sorge}@tu-berlin.de

Abstract. We examine the algorithmic tractability of NP-hard combinatorial feature selection problems in terms of parameterized complexity theory. In combinatorial feature selection, one seeks to discard dimensions from high-dimensional data such that the resulting instances fulfill a desired property. In parameterized complexity analysis, one seeks to identify relevant problem-specific quantities and tries to determine their influence on the computational complexity of the considered problem. In this paper, for various combinatorial feature selection problems, we identify parameterizations and reveal to what extent these govern computational complexity. We provide tractability as well as intractability results; for example, we show that the Distinct Vectors problem on binary points is polynomial-time solvable if each pair of points differs in at most three dimensions, whereas it is NP-hard otherwise.

1 Introduction

Feature selection in a high-dimensional data space means to choose a subset of features (that is, dimensions) such that some desirable data properties are preserved or achieved. *Combinatorial* feature selection [14, 5] is a well-motivated alternative to the more frequently studied affine feature selection: While affine feature selection combines features to reduce dimensionality, combinatorial feature selection chooses a subspace by discarding some dimensions. The advantage of the latter is that the resulting reduced feature space is easier to interpret. See Charikar et al. [5] for a more extensive discussion in favor of combinatorial feature selection. Unfortunately, combinatorial feature selection problems are typically computationally very hard to solve (NP-hard and also hard to approximate [5]), resulting in the use of heuristic approaches in practice [2, 8, 12, 13].

In this work, mainly following Charikar et al. [5], who provided classical computational hardness results (NP-hardness and inapproximability), we adopt the fresh perspective of parameterized complexity analysis. We thus refine the known picture of the computational complexity landscape of combinatorial feature selection problems. Intuitively speaking, our guiding principle is to identify

* Vincent Froese was supported by DFG, project DAMM (NI 369/13). René van Bevern and Manuel Sorge were supported by DFG, project DAPA (NI 369/12).

To appear in *Proceedings of the 38th International Symposium on Mathematical Foundations of Computer Science (MFCS'13), Klosterneuburg, Austria, August 2013.* © *Springer.*

problem-specific parameters (quantities such as number of dimensions to discard or number of dimensions to keep) and to analyze how these quantities influence the problem complexity. The point here is that in relevant applications these parameters can be small. Hence, the central question is whether the considered problems become computationally tractable in the case of small parameters.

We revisit two categories of combinatorial feature selection problems (namely dimension reduction and clustering problems) as introduced by Charikar et al. [5]. Within their framework they defined (amongst others) two problems called Distinct Vectors and Hidden Clusters. In this work, we consider Distinct Vectors and introduce a new problem called L_p-Hidden Cluster Graph which is based on Hidden Clusters. For both problems, we shed new light on the (non-)existence of provably tractable special cases.

Distinct Vectors is a dimension reduction problem defined as follows:

Distinct Vectors
Input: A multiset $S = \{x_1, \dots, x_n\} \subseteq \Sigma^d$ of n distinct points in d dimensions and $k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \dots, d\}$ of dimensions with $|K| \leq k$ such that all points in $S_{|K}$ are still distinct?

Throughout this work, $S_{|K} := \{x_{1|K}, \dots, x_{n|K}\}$ denotes the multiset of projections $x_{i|K}$ of the points in S into the dimensions in K, that is, dimensions not in K are set to zero. Distinct Vectors is NP-hard to approximate within a logarithmic factor [5]. It is also known as the Minimal Reduct problem in *rough set theory* [17] and was already earlier proven to be NP-hard [18].

In the clustering category, we assume that the input data would cluster well once some noise is removed. The representative problem for this category is Hidden Clusters [5]. The goal is to maximize the number of dimensions that allow for a clustering of the data into a predefined number of cluster centers of a given radius. Notably, the number of sought clusters has to be known in advance. This is not always realistic. Hence, we would like also to reveal clusterings in our data without knowing the number of clusters beforehand. To this end, we employ a clustering notion from graph-based data clustering: Instead of formulating a cluster as a point set within a given radius r from some center as in Hidden Clusters, we now formulate a cluster as a set of points of pairwise distance at most r. Such sets of points form cliques in a "threshold graph" that contains an edge between two points whenever their distance is at most r. The search for a clustering now essentially becomes the search for a graph whose connected components are cliques. In contrast to Hidden Clusters, this also expresses the need of points in different clusters to be dissimilar to each other.

L_p-Hidden Cluster Graph
Input: A set $S = \{x_1, \dots, x_n\} \subseteq \Sigma^d$ with $\Sigma \subseteq \mathbb{Q}$, $r \in \mathbb{Q}_0^+$, $k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \dots, d\}$ of dimensions with $|K| \geq k$ such that the graph $G_K = (V, E_K)$ with $V := S$, $E_K := \{\{x_i, x_j\} \mid x_i \neq x_j \in V,\ \mathrm{dist}_{|K}^{(p)}(x_i, x_j) \leq r\}$ is a cluster graph (that is, a union of disjoint cliques)?

Herein, $\mathrm{dist}^{(p)}_{|K}$ is a metric computing the distance between two points from Σ^d projected to the dimensions in K. We explicitly consider the distance functions induced by the L_p-norm: $\mathrm{dist}^{(p)}(x,y) := \sum_{j=1}^{d} |(x-y)_j|^p$ for $p \in \mathbb{N}$ and $\mathrm{dist}^{(\infty)}(x,y) := \max_{j \in \{1,\ldots,d\}} |(x-y)_j|$. By $(x)_j$ we denote the value of $x \in \Sigma^d$ in the j-th dimension. Note that G_K is a so-called unit ball graph.

Parameterized complexity preliminaries. The computational complexity of a parameterized problem is measured in terms of two quantities: one is the input size, the other is the *parameter* (usually a positive integer). A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is called *fixed-parameter tractable* with respect to a parameter k if it can be solved in $f(k) \cdot |x|^{O(1)}$ time, where f is a computable function only depending on k, and $|x|$ is the size of the input instance x. A *problem kernel* for a parameterized problem is a many-one self-reduction that runs in polynomial time such that the produced instances have size upper-bounded by some function exclusively depending on the parameter. Existence of a problem kernel is equivalent to fixed-parameter tractability [10, 11, 16].

A *parameterized reduction* from a parameterized problem P to another parameterized problem P' is a function that, given an instance (x,k), computes in $f(k) \cdot |x|^{O(1)}$ time an instance (x',k') (with k' only depending on k) such that (x,k) is a "yes"-instance of P if and only if (x',k') is a "yes"-instance of P'. The two basic complexity classes for showing (presumable) fixed-parameter intractability are called W[1] and W[2]; the standard assumption is that W[1]-hard and W[2]-hard problems are not fixed-parameter tractable [10, 11, 16].

Throughout this work we assume that arithmetic operations such as additions and comparisons of numbers can be done in $O(1)$ time.

Our contributions. For Distinct Vectors we prove W[2]-hardness with respect to the solution size k. In addition, we observe that it cannot be solved in $d^{o(k)} \cdot |x|^{O(1)}$ time unless W[1] = FPT (which is strongly believed not to be the case). Moreover, for Distinct Vectors restricted to a binary input alphabet, we give the following complexity dichotomy: if the maximum pairwise Hamming distance h between input points is at most three, then Distinct Vectors is polynomial-time solvable, and it is NP-complete for $h \geq 4$. The latter NP-completeness proof also implies W[1]-hardness with respect to the parameter $d-k$ ("number of dimensions to discard"). In contrast, we provide some problem kernels with respect to the combined parameters "alphabet size combined with k" and "h combined with k".

For L_p-Hidden Cluster Graph, we show that it is W[2]-hard with respect to the number t of discarded dimensions for all $p \in \mathbb{N}$, whereas it is fixed-parameter tractable with respect to t combined with the radius r. L_∞-Hidden Cluster Graph even is polynomial-time solvable in general.

Due to the lack of space, several technical details are deferred to a full version.

2 Distinct Vectors

Skowron and Rauszer [18] first proved NP-hardness for Minimal Reduct (which is equivalent to Distinct Vectors) by a reduction from Hitting Set. Charikar et al. [5] additionally showed that there is some constant c such that Distinct Vectors is not polynomial-time approximable within a factor of $c \log d$ unless P = NP. We analyze various restricted scenarios for the Distinct Vectors problem and conduct a more fine-grained computational complexity analysis which, unfortunately, yields further hardness results in most cases. More specifically, we consider the cases of (i) retaining *few* dimensions, (ii) deleting *few* dimensions, and (iii) *small* pairwise differences between points.

We first present results for a binary input alphabet in Section 2.1 and then proceed with results for larger and unbounded alphabet size in Section 2.2.

2.1 Bounded Pairwise Hamming Distance: A Complexity Dichotomy

Throughout this subsection we focus on instances with a binary input alphabet $\Sigma = \{0, 1\}$. We further restrict our considerations to instances with points of bounded "*degree of distinctiveness*". Herein, we refer to instances where each pair of points differs in at most h dimensions. In other words, the Hamming distance of any pair of points is bounded from above by h. For example, this situation can arise for *sparse* data sets where the points mainly contain 0's. Intuitively, if the data set consists of points that are all "similar" to each other, one could hope to be able to solve the instance efficiently since there are at most h dimensions to choose from in order to distinguish two points. The following theorem, however, shows that this intuition is deceptive: when crossing a certain threshold of dissimilarity, the complexity suddenly changes.

Theorem 1. *For a binary input alphabet $\Sigma = \{0, 1\}$,* Distinct Vectors *is*

i) solvable in $O(n^3 d)$ time if the maximum pairwise Hamming distance h of the input vectors is at most three, and
ii) NP*-hard for $h \geq 4$.*

In order to prove (i), we use the following combinatorial lemma.

Lemma 2. *Let $m, n \in \mathbb{N}$ with $m > n + 1$ and let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be a family of pairwise different sets of size n each with $\forall A_i \neq A_j : |A_i \cap A_j| = n - 1$. Then, $\forall A_i \neq A_j : A_i \cap A_j = \bigcap_{k=1}^{m} A_k$.*

Now, we can sketch a proof of Theorem 1(i).

Proof (Sketch, Theorem 1(i)). We give a search tree algorithm that solves a given Distinct Vectors instance (S, k). The restriction $h = 3$ guarantees that there are not "too many" branches in the search tree to consider and, hence, that the search tree has polynomial size. For $x \in S$ and $i \in \mathbb{N}$ we define $D_x := \{j \in \{1, \ldots, d\} \mid (x)_j = 1\}$ and $S_i := \{x \in S \mid i = |D_x|\}$. Without loss of generality, we can assume that $\mathbf{0} := (0, \ldots, 0) \in S$. If this is not the case, then we can simply fix an arbitrary point $x_0 \in S$ and exchange 1's and 0's in all points in S in all dimensions where x_0 equals 1. This yields an equivalent instance with $x_0 = \mathbf{0} \in S$ in linear time.

x_1	1	1	1				
x_2	1	1		1			
x_3	1	1			1		
x_4	1	1				1	
x_5	1	1					1

Fig. 1: The points in $S_{3|D^3} \subseteq \{0,1\}^7$ represented as rows of a matrix with columns corresponding to the dimensions in D^3. Empty cells represent zero entries. Each pair of points shares a 1 in two dimensions. For more than four points there exist two dimensions in which all points equal 1. At most one of the other dimensions is not contained in a solution.

Let (S,k), $S \subseteq \{0,1\}^d$, be an instance of Distinct Vectors with $|S| = n$. The bound $h = 3$ implies that each point in S contains at most three 1's since otherwise it differs in more than three dimensions from $\mathbf{0}$. Thus, we can partition the data set $S = \{\mathbf{0}\} \uplus S_1 \uplus S_2 \uplus S_3$. Moreover, the restriction $h = 3$ also implies the following two conditions, which constitute the crucial aspects for our proof.

$$\forall x, y \in S_3 : |D_x \cap D_y| = 2, \tag{1}$$
$$\forall x, y \in S_2 : |D_x \cap D_y| = 1. \tag{2}$$

Both conditions have to be met since otherwise there exists a pair of points differing in at least four dimensions. The algorithm starts with considering the subset S_3. The points in S_3 can only be distinguished from each other by a subset of the dimensions $D^3 := \bigcup_{x \in S_3} D_x$. If $|S_3| \leq 4$, then we simply branch over all possible subsets of D^3. With a constant number of at most four distinct points in S_3, the size of D^3 is also bounded by a constant and so there are only constantly many subsets to try. If $|S_3| > 4$, then statement (1) together with Lemma 2 implies that $C^3 := \bigcap_{x \in S_3} D_x$ contains two dimensions. It follows that for each dimension $j \in D^3 \setminus C^3$ there exists exactly one point $x \in S_3$ with $(x)_j = 1$. This situation is depicted in Figure 1. In order to distinguish all points in S_3 from each other, any solution contains at least all but one dimension from $D^3 \setminus C^3$. Hence, we can try out all subsets of $D^3 \setminus C^3$ of size at least $|S_3| - 1$. Together with the four possible subsets of C^3 we end up with at most $4(n+1)$ subsets of D^3 to branch over. Similarly, we obtain that we have to branch over at most $2(n+1)$ subsets of dimensions to distinguish all points in S_2. Thus, we end up with $O(n^2)$ possible subset selections. For the set S_1 no branching is necessary. For each selection we check whether it is a solution or not. This can be done in $O(nd)$ time by sorting the data set lexicographically with radix sort and comparing successive points. Overall, we obtain a search tree algorithm with running time of $O(n^3 d)$. □

When the pairwise Hamming distance h of the input vectors is at least four, the conditions (1) and (2) from the proof of Theorem 1(i) do not hold. Therefore, we cannot apply Lemma 2, which is crucial in that it guarantees a regular structure of the data set that makes the instance easy to solve. Instead, we can observe that, if a pair of points is allowed to take on different values in at least four dimensions, then the data set can "encode" arbitrary graphs. We exploit this to prove Theorem 1(ii), that is, that Distinct Vectors is NP-complete for $h \geq 4$. To this end, we describe a polynomial-time many-one reduction from a special variant of the Independent Set problem in graphs, which is defined as follows.

DISTANCE-3 INDEPENDENT SET
Input: An undirected graph $G = (V, E)$ and $k \in \mathbb{N}$.
Question: Is there a subset of vertices $I \subseteq V$ of size at least k such that any pair of vertices from I has distance at least three?

Here, the distance of two vertices is the number of edges contained in a shortest path between them. DISTANCE-3 INDEPENDENT SET can easily be shown to be NP-hard by a reduction from INDUCED MATCHING [3].

We are now ready to prove that DISTINCT VECTORS is NP-complete for $h \geq 4$, even if the input alphabet Σ is binary.

Proof (Theorem 1(ii)). It is easy to check that DISTINCT VECTORS is in NP. To show NP-hardness, let $(G = (V, E), k)$ with $|V| = n$ and $|E| = m$ be an instance of DISTANCE-3 INDEPENDENT SET and let Z be the $m \times n$ transposed incidence matrix of G with rows corresponding to edges and columns to vertices. The data set S of our DISTINCT VECTORS instance (S, k') is defined to contain all m row vectors of Z and the null point $\mathbf{0} = (0, \ldots, 0) \in \{0, 1\}^n$. The sought solution size is set to $k' := n - k$. Notice that each point in S contains exactly two 1's (except for $\mathbf{0}$). Thus, each pair of points differs in at most $h = 4$ dimensions. The instance (S, k') can be computed in $O(nm)$ time.

Correctness of the reduction follows by the following argument: The subset $I \subseteq V$ is a solution of (G, k) if and only if it is of size k and every edge in G has at least one endpoint in $V \setminus I$ and no vertex in $V \setminus I$ has two neighbors in I. In other words, the latter condition says that no two edges with an endpoint in I share the same endpoint in $V \setminus I$. Equivalently, for the subset K of dimensions corresponding to the vertices in $V \setminus I$, it holds that all row vectors of Z in $S_{|K}$ contain at least one 1 and no two vectors contain only a single 1 in the same dimension. This holds if and only if K is a solution for (S, k'), because S contains the null point and thus two points can only be identical in $S_{|K}$ if either they consist of 0's only or contain a single 1 in the same dimension. □

We remark that from a W[1]-hardness result for INDUCED MATCHING [15] we can infer W[1]-hardess for DISTANCE-3 INDEPENDENT SET with respect to k. Since the proof of Theorem 1(i) yields a parameterized reduction from DISTANCE-3 INDEPENDENT SET parameterized by k to DISTINCT VECTORS parameterized by the number $n - k' = k$ of dimensions to discard, we have the following:

Corollary 3. DISTINCT VECTORS *is* W[1]*-hard with respect to the number of dimensions to delete.*

2.2 Distinct Vectors with an Arbitrary Alphabet

As we have seen in Section 2.1, DISTINCT VECTORS is NP-complete and W[1]-hard with respect to the number of dimensions to be deleted even in the case of a binary alphabet when the pairwise Hamming distance of the vectors is bounded by four. Nevertheless, we note later in this section that some tractability results are achievable even for larger alphabets. First, however, we mention that HITTING

Set parameterized by the sought solution size (which is W[2]-hard, as shown by Downey and Fellows [10]) is parameterized reducible to Distinct Vectors in the case of an arbitrary alphabet size, which yields the following:

Theorem 4. *Allowing an arbitrary alphabet size,* Distinct Vectors *is $W[2]$-hard with respect to the parameter k.*

Proof. We give a parameterized reduction from Hitting Set:

Hitting Set

Input: A finite universe U, a collection $\mathcal{C}$ of subsets of U and a nonnegative integer k.

Question: Is there a subset $K \subseteq U$ with $|K| \leq k$ such that K contains at least one element from each subset in $\mathcal{C}$?

Given an instance $(U, \mathcal{C}, k)$ of Hitting Set with $U = \{u_1, \dots, u_m\}$ and $\mathcal{C} = \{C_1, \dots, C_n\}$, we construct a Distinct Vectors instance (S, k') with $S := \{x_1, \dots, x_n, \mathbf{0}\} \subseteq \mathbb{N}^m$ and $k' := k$, where $\mathbf{0} = (0, \dots, 0)$ and

$$(x_i)_j := \begin{cases} i, & u_j \in C_i \\ 0, & u_j \notin C_i \end{cases} \text{ for all } i \in \{1, \dots, n\}, j \in \{1, \dots, m\}.$$

The above instance is polynomial-time computable. If $K \subseteq U$ is a solution of $(U, \mathcal{C}, k)$, then $K \cap C_i \neq \emptyset$ for all $C_i \in \mathcal{C}$ and thus for each $x_i \in S$ there is a dimension corresponding to some element in K, such that x_i equals i in this dimension and is thus different from all other points in S. Conversely, in order to distinguish any $x_i \in S$ from $\mathbf{0}$, any solution K' of (S, k') has to contain a dimension where x_i is different from 0. This implies that the subset of U corresponding to K' contains at least one element of each C_i and is thus a solution of the original instance. Finally, note that this is a parameterized reduction since $k' = k$. □

It was shown by Chen et al. [6] that, unless FPT = W[1], Hitting Set cannot be solved in $|U|^{o(k)} \cdot |x|^{O(1)}$ time. Since the reduction from Hitting Set yields an instance with $d = |U|$ dimensions and solution size k in polynomial time, it follows that Distinct Vectors cannot be solved in $d^{o(k)} \cdot |x|^{O(1)}$ time unless FPT = W[1]. On the positive side, Distinct Vectors can trivially be solved by trying out all subsets of dimensions of size k within $d^k \cdot |x|^{O(1)}$ time. Consequently, we obtain the following corollary.

Corollary 5. *If* FPT $\neq$ W[1]*, then the fastest algorithm solving* Distinct Vectors *has a running time of $d^{\Theta(k)} \cdot |x|^{O(1)}$.*

Although Theorem 4 shows that Distinct Vectors is W[2]-hard with respect to the parameter k, we can provide a problem kernel for Distinct Vectors if we additionally consider the input alphabet size $|\Sigma|$ as parameter. The size of the problem kernel is superexponential in the parameter $(k, |\Sigma|)$. Clearly, a problem kernel of polynomial size would be desirable. However, based on the complexity-theoretic assumption that the polynomial hierarchy does not collapse, polynomial-size kernels do not exist even with the additional parameter n of input points:

Theorem 6.

i) *There exists an* $O(|\Sigma|^{|\Sigma|^k+k}/|\Sigma|! \cdot \log |\Sigma|)$*-size problem kernel computable in* $O(d^2n^2)$ *time for* Distinct Vectors.
ii) *Unless* NP $\subseteq$ coNP/poly, Distinct Vectors *does not admit a polynomial-size kernel with respect to the combined parameter* $(n, |\Sigma|, k)$.

Proof (Sketch). (i) The idea is that k dimensions can distinguish at most $|\Sigma|^k$ points. Observe that every dimension partitions the data set into at most $|\Sigma|$ non-empty subsets. If any two dimensions yield the same partitioning, we can simply delete one of them. Thus, any "yes"-instance has at most $|\Sigma|^{|\Sigma|^k}/|\Sigma|!$ essentially different dimensions. Any larger instance can be discarded as "no"-instance.

(ii) The reduction from Hitting Set in the proof of Theorem 4 can easily be turned into a reduction from the closely related Set Cover. For Set Cover, Dom et al. [9] showed that there is no polynomial-size kernel, which in combination with the reduction also excludes polynomial-size kernels for Distinct Vectors. □

Besides parameterizing by the alphabet size, the maximum Hamming distance h of all pairs of points also yields tractability results. It is possible to reduce Distinct Vectors to h-Hitting Set for which problem kernels with respect to (h, k) are known [1]. These can be used to obtain problem kernels for Distinct Vectors in turn. We omit the details here and refer to a full version.

In this subsection we have seen that Distinct Vectors can basically be regarded as a special Hitting Set problem. Interestingly, Hitting Set with respect to the solution size is W[2]-hard in general, but for constant-size alphabets, Distinct Vectors is fixed-parameter tractable (Theorem 6). Thus, the set systems induced by instances of Distinct Vectors involve a certain structure that makes them easier to solve.

3 Hidden Cluster Graph

This section investigates the complexity of Hidden Cluster Graph. It turns out that, in contrast to the Hidden Clusters problem—which is NP-hard for the radius $r = 0$ and, hence, for arbitrary metrics—the choice of the distance function has a considerable influence on the tractability of Hidden Cluster Graph.

Theorem 7.

i) L_∞-Hidden Cluster Graph *is solvable in* $O(d(n^2d + n^3))$ *time.*
ii) *For* $p \in \mathbb{N}$, L_p-Hidden Cluster Graph *is* NP*-complete and even* W[2]*-hard with respect to the parameter "maximum number* t *of allowed dimension deletions".*

Proof (Sketch). The proof of (i) is deferred to a full version of the paper. The basic idea is to insert missing edges by deleting all dimensions in which the corresponding endpoints differ more than r.

To prove (ii), first observe that L_p-Hidden Cluster Graph is contained in NP: given a solution set K, we can build the corresponding graph G_K and check whether it is a cluster graph in polynomial time. To show NP- and W[2]-hardness, we give a polynomial-time executable parameterized many-one reduction from the NP-hard and W[2]-hard Lobbying problem [7, 4] occurring in computational social choice.

Lobbying
Input: A matrix $A \in \{0,1\}^{m\times n}$ with an odd number n of columns and an integer $k > 0$.
Question: Can one modify (set to zero) at most k columns in A such that in the resulting matrix each row contains at least as many zeros as ones?

Compared with the problem definition of Bredereck et al. [4], we exchanged the roles of ones and zeros and of rows and columns. This clearly does not change the complexity. Moreover, we ask for "at least as many" instead of "more" zeros than ones per row. Since the problem is W[2]-hard with respect to k if the number of columns n is odd [7], these conditions are equivalent and our variant is also W[2]-hard. We assume that every row of A contains more ones than zeros because otherwise we could delete it from the input without changing the answer to the question.

Our reduction works as follows: Let (A, k) be an instance of Lobbying with $A \in \{0,1\}^{m\times n}$ containing m rows $a_1, \dots, a_m \in \{0,1\}^n$. We define an L_p-Hidden Cluster Graph instance (S, r, k') with

$$S := \bigcup_{1\le i\le m} \{u_i, v_i, w_i\} \subseteq \Sigma^n, \quad r := 2^{p-1}n, \quad k' := n - k.$$

The idea is to let S contain three data points u_i, v_i, and w_i for every row a_i in A such that their induced subgraph $H_i := G_{\{1,\dots,n\}}[\{u_i, v_i, w_i\}]$ is a P_3, that is, a path with three vertices. To this end, let

$$\begin{aligned} u_1 &:= \mathbf{0}, & w_1 &:= 2a_1, & v_1 &:= \frac{u_1 + w_1}{2}, \\ u_i &:= w_{i-1} + 2\boldsymbol{n}, & w_i &:= u_i + 2a_i, & v_i &:= \frac{u_i + w_i}{2}, \end{aligned}$$

for $i \in \{2, \dots, m\}$, where $\boldsymbol{x} := (x, \dots, x) \in \Sigma^n$ for $x \in \Sigma$. The above construction requires $\mathbb{N} \subseteq \Sigma$ in order to be well-defined. It is computable in $O(mn)$ time. Note that this is a parameterized reduction with respect to t since $t = n - k' = k$. Figure 2 illustrates the constructed data set. Now, for all $i = 1, \dots, m$,

$$\operatorname{dist}^{(p)}(u_i, w_i) = \sum_{j=1}^{n} 2^p \cdot |(a_i)_j|^p \ge 2^p \cdot \left(\frac{n+1}{2}\right) > r$$

and $\operatorname{dist}^{(p)}(u_i, v_i) = \operatorname{dist}^{(p)}(v_i, w_i) \le n \le r$. Since $G_{\{1,\dots,n\}}$ is defined to contain an edge between two vertices if and only if the distance of their corresponding points in S is at most r, it follows indeed that H_i is a P_3. By construction, the subgraphs H_i are independent of each other in the sense that, for every non-empty

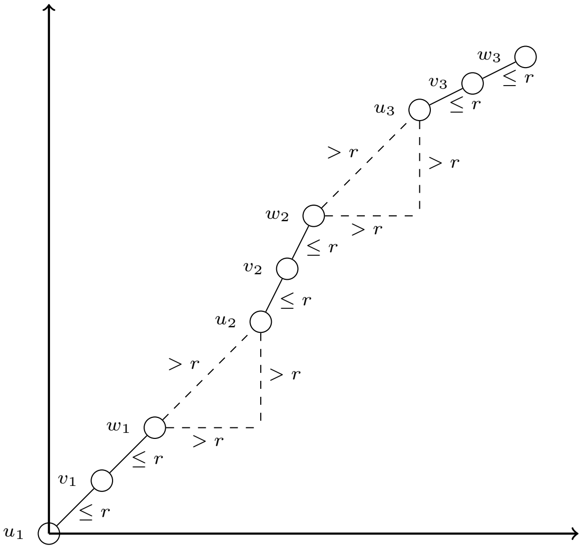

Fig. 2: A two-dimensional illustration of the constructed L_p-Hidden Cluster Graph instance: For each row a_i in the lobbying matrix A there are three points u_i, v_i, w_i in the data set S such that, for every non-empty subset of dimensions K, they induce a P_3 in G_K. This is achieved by recursively setting $v_i = u_i + a_i$, $w_i = v_i + a_i$ and choosing an appropriate radius $\|a_i\|_p^p \leq r < \|2a_i\|_p^p$. Note that the point u_{i+1} is defined such that its distance to w_i is greater than r in every dimension, which ensures that there is no edge between vertices from different P_3's for any K.

subset $K \subseteq \{1, \ldots, n\}$ of dimensions, G_K never contains an edge between any vertices from H_i and H_j for $i \neq j$. To verify this, let $1 \leq i < j \leq m$ and note that, by construction, the smallest distance between any vertices from H_i and H_j is the distance of w_i and u_j. For every non-empty subset K of dimensions, $\text{dist}^{(p)}_{|K}(u_j, w_i)$ is

$$\begin{aligned} &\sum_{l\in K} \left| \Big(w_i + (j-i)\cdot 2\boldsymbol{n} + \sum_{k=1}^{j-i-1} 2a_{i+k} \Big)_l - (w_i)_l \right|^p \\ \geq &\sum_{l\in K} 2^p |(\boldsymbol{n})_l|^p = 2^p|K| \cdot n \geq 2^p n > r. \end{aligned}$$

Thus, there cannot be an edge in G_K between vertices from H_i and H_j for any K. It follows that the only solution of this instance is the cluster graph consisting of the m disjoint triangles obtained by inserting the missing edge in each H_i. In order to insert the missing edge between u_i and w_i in every H_i, we have to find a subset of dimensions K such that

$$\text{dist}^{(p)}_{|K}(u_i, w_i) = 2^p \sum_{j\in K} |(a_i)_j|^p \leq r = 2^{p-1} n$$

holds for all $i = 1, \ldots, m$. In other words, we have to delete at most t dimensions (that is, setting entries in a_i to zero) such that for the remaining dimensions K it holds that $\sum_{j\in K} |(a_i)_j|^p \leq n/2$. Since a_i is a binary vector, this upper bound states that the modified a_i contains at least as many zeros as ones, which is exactly our Lobbying problem. So, the L_p-Hidden Cluster Graph instance is a "yes"-instance if and only if the initial Lobbying instance is a "yes"-instance. □

The reduction in the proof of Theorem 7(ii) is not only running in polynomial time but also is a polynomial parameter transformation in the sense that the number of data points n equals three times the number of rows of A, the number t of dimensions to discard equals k and the number d of dimensions equals the number of columns of A. Hence, we can transfer some problem kernel lower bound results for Lobbying [4, Theorems 3 & 4] to L_p-Hidden Cluster Graph.

Corollary 8. *Unless* NP $\subseteq$ coNP/poly, L_p-Hidden Cluster Graph *does neither admit a polynomial-size kernel with respect to* (n, t) *nor with respect to* d.

One easily observes that the proof of Theorem 7(ii) generates instances of L_p-Hidden Cluster Graph of unbounded diameter δ, which is defined as the maximum distance between any two vectors in S. This scenario seems not always realistic in practice since features often take on values around some expected value. And indeed, we can show that if δ and the number t of dimensions to be deleted are constant, then L_p-Hidden Cluster Graph is solvable in cubic time. To this end, observe that if $r > \delta$ in an input instance, we can immediately answer "yes", since the graph $G_{\{1,\dots,d\}}$ is then a clique and thus a cluster graph. For $r \leq \delta$, we can prove the following theorem using a search tree algorithm. For bounding the search tree size, we need the additional condition that the data set only contains integers.

Theorem 9. L_p-Hidden Cluster Graph *is* $O((2^p r)^t \cdot (n^2 d + n^3))$*-time solvable for* $p \in \mathbb{N}$ *and an alphabet* $\Sigma \subseteq \mathbb{Z}$.

Obviously, Theorem 9 does not yield an algorithm that is applicable to large data sets. Yet it shows that, despite the hardness of the problem in the general case, the development of efficient algorithms on realistic data might be possible.

4 Outlook

We conclude with some directions for future research. As to Distinct Vectors, our kernelization results in Theorem 6 (lower and upper bounds) are still far apart and ask for closing this gap. Further, it would be interesting to find improved kernels for the parameterization by Hamming distance h and number of retained dimensions k. Here, exploiting structural restrictions in context with connections to Hitting Set seems promising. Finally, we left open to generalize the polynomial-time algorithm for pairwise Hamming distance at most three from binary alphabets (see Theorem 1) to general alphabets.

As to Hidden Cluster Graph, spotting further natural and useful parameterizations is desirable.

Acknowledgements. We are grateful to anonymous MFCS referees for extensive and constructive feedback.

Bibliography

[1] R. van Bevern. Towards optimal and expressive kernelization for d-hitting set. *Algorithmica*, 2013. Online available. 8

[2] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, 1997. 1

[3] A. Brandstädt and R. Mosca. On distance-3 matchings and induced matchings. *Discrete Appl. Math.*, 159(7):509–520, 2011. 6

[4] R. Bredereck, J. Chen, S. Hartung, S. Kratsch, R. Niedermeier, and O. Suchý. A multivariate complexity analysis of lobbying in multiple referenda. In *Proc. 26th AAAI*, pages 1292–1298, 2012. 9, 11

[5] M. Charikar, V. Guruswami, R. Kumar, S. Rajagopalan, and A. Sahai. Combinatorial feature selection problems. In *Proc. 41st FOCS*, pages 631–640, 2000. 1, 2, 4

[6] J. Chen, B. Chor, M. Fellows, X. Huang, D. Juedes, I. A. Kanj, and G. Xia. Tight lower bounds for certain parameterized NP-hard problems. *Information and Computation*, 201(2):216–231, 2005. 7

[7] R. Christian, M. R. Fellows, F. Rosamond, and A. Slinko. On complexity of lobbying in multiple referenda. *Review of Economic Design*, 11(3):217–224, 2007. 9

[8] A. Dasgupta, P. Drineas, B. Harb, V. Josifovski, and M. W. Mahoney. Feature selection methods for text classification. In *Proc. 13th ACM SIGKDD*, pages 230–239, 2007. 1

[9] M. Dom, D. Lokshtanov, and S. Saurabh. Incompressibility through colors and IDs. In *Proc. 36th ICALP*, volume 5555 of *LNCS*, pages 378–389. Springer, 2009. 8

[10] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer, 1999. 3, 7

[11] J. Flum and M. Grohe. *Parameterized Complexity Theory.* Springer, 2006. 3

[12] G. Forman. An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.*, 3:1289–1305, 2003. 1

[13] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003. 1

[14] D. Koller and M. Sahami. Towards optimal feature selection. In *Proc. 13th ICML*, pages 284–292, 1996. 1

[15] H. Moser and D. M. Thilikos. Parameterized complexity of finding regular induced subgraphs. *J. Discrete Algorithms*, 7(2):181–190, 2009. 6

[16] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006. 3

[17] Z. Pawlak. *Rough Sets: Theoretical Aspects of Reasoning about Data.* Kluwer Academic, 1991. 2

[18] A. Skowron and C. Rauszer. The discernibility matrices and functions in information systems. In R. Slowinski, editor, *Intelligent Decision Support—Handbook of Applications and Advances of the Rough Sets Theory*, pages 331–362. Kluwer Academic, 1992. 2, 3