Parameterized Resiliency Problems*

Jason Crampton[1], Gregory Gutin[1], Martin Koutecký[2], and Rémi Watrigant[3]

[1]Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK
[2]Technion - Israel Institute of Technology, Haifa, Israel and Charles University, Prague, Czech Republic
[3]Université de Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP, France

Abstract

We introduce an extension of decision problems called *resiliency problems*. In a resiliency problem, the goal is to decide whether an instance remains positive after any (appropriately defined) perturbation has been applied to it. To tackle these kinds of problems, some of which might be of practical interest, we introduce a notion of resiliency for Integer Linear Programs (ILP) and show how to use a result of Eisenbrand and Shmonin (Math. Oper. Res., 2008) on Parametric Linear Programming to prove that ILP Resiliency is fixed-parameter tractable (FPT) under a certain parameterization.

To demonstrate the utility of our result, we consider natural resiliency variants of several concrete problems, and prove that they are FPT under natural parameterizations. Our first results concern a four-variate problem which generalizes the Disjoint Set Cover problem and which is of interest in access control. We obtain a complete parameterized complexity classification for every possible combination of the parameters. Then, we introduce and study a resiliency variant of the Closest String problem, for which we extend an FPT result of Gramm et al. (Algorithmica, 2003). We also consider problems in the fields of scheduling and social choice. We believe that many other problems can be tackled by our framework.

Keywords: Fixed Parameter Tractability; Access Control; Parametric Integer Linear Programming; resiliency problems; social choice theory; computational biology

*The research of Crampton and Gutin was partially supported by Leverhulme Trust grant RPG-2018-161. Gutin's research was partially supported by Royal Society Wolfson Research Merit Award. Koutecký's research was supported by a Technion postdoctoral fellowship and projects 17-09142S of GA ČR. This paper is based on two papers [4, 3] published in conference proceeding of AAIM 2016 and CIAC 2017.

1 Introduction

Questions of integer linear programming (ILP) feasibility are usually answered by finding an integral assignment of variables x satisfying $Ax \leq b$. By Lenstra's theorem [28], this problem can be solved in $O^*(f(n)) := O(f(n)L^{O(1)})$ time and space, where f is a function of the number n of variables only, and L is the size of the ILP. (Subsequent research has obtained an algorithm having the above running time, with $f(n) = n^{O(n)}$, and using polynomial space, that is, space $L^{O(1)}$ [15, 22].) In the language of parameterized complexity, this means that ILP Feasibility is fixed-parameter tractable (FPT) parameterized by the number of variables. Note that there are a number of parameterized problems for which the only (known) way to prove fixed-parameter tractability is to use Lenstra's theorem[1] [6]. For more details on this topic, we refer the reader to [6, 8].

The notion of *resiliency* measures the extent to which a system can tolerate modifications to its configuration and still satisfy given criteria. An organization might, for example, wish to know whether it will still be able to continue functioning, even if some of its staff become unavailable. In the language of decision problems, we would like to know whether an instance is still positive after *any* (appropriately defined) modification to the input. Intuitively, the resiliency variant of a problem is likely to be harder than the problem itself; a naive algorithm would consider every allowed modification of the input, and then decide whether a solution exists for each modification.

In this paper we introduce a framework for dealing with resiliency problems, and study their computational complexity through the lens of fixed-parameter tractability. We define resiliency for Integer Linear Programs by considering a system $\mathcal{R}$ of linear inequalities whose variables are partitioned into vectors x and z. We denote by $\mathcal{R}_z$ all inequalities in $\mathcal{R}$ containing only variables from z. We say that $\mathcal{R}$ is z-resilient if for any integral assignment of variables z satisfying all inequalities of $\mathcal{R}_z$, there is an integral assignment of variables x such that all inequalities of $\mathcal{R}$ are satisfied. We prove in Theorem 2.3 that the obtained problem can be solved in FPT time for a suitable parameterization, using a result of Eisenbrand and Shmonin on *Parametric Integer Linear Programming* [10].[2] Our approach provides a generic framework that captures many situations. Firstly, it applies to ILP, a general and powerful model for representing many combinatorial problems. Secondly, the resiliency part of each problem can be represented as a whole ILP with its own variables and constraints, instead of, say, a simple additive term. Hence, we believe that our method can be applied to many other problems, as well as many different and intricate definitions of resiliency.

[1] Lenstra's theorem allows us to prove a mainly classification result, *i.e.* the FPT algorithm is unlikely to be efficient in practice. Nevertheless, Lenstra's theorem indicates that efficient FPT algorithms are a possibility, at least for subproblems of the problem under considerations.

[2] Note that the result of Eisenbrand and Shmonin is an improvement of an earlier result of Kannan [23]. Unfortunately, Kannan's theorem was based on an incorrect key lemma; the proof of the theorem of Eisenbrand and Shmonin does not have this problem as it uses a correct weaker version of the key lemma proved by Eisenbrand and Shmonin, see, e.g., [33].

To illustrate the fact that our approach might be useful in different situations, we apply our framework to several concrete problems.[3] Central among them is the Resiliency Disjoint Set Cover problem (RDSCP) defined in Section 3. RDSCP is a generalization of the Disjoint Set Cover problem, and has practical applications in the context of access control [5, 29], where an equivalent formulation of the problem is called the Resiliency Checking problem, also given in Section 3. Given a set U of size n and a family $\mathcal{F}$ of m subsets of U, the RDSCP asks whether after removing from $\mathcal{F}$ any subfamily of size at most s the following property is still satisfied: there are d disjoint set covers of U, each of size at most t.

Thus, RDSCP has five natural parameters n, m, s, d and t. Since the size of a non-trivial instance can be bounded by a function of m only and since in Resiliency Checking problem m is usually much larger than the other parameters [5, 29], in this paper will only focus on parameters n, s, d and t. Our main result, obtained using Theorem 2.3, is that RDSCP is FPT when parameterized by n. This, together with some additional results, allow us to determine the complexity of RDSCP (FPT, XP, W[2]-hard, para-NP-hard or para-coNP-hard) for all combinations of the four parameters. Note that by definition, a problem with several parameters $p_1, \ldots, p_\ell$ is the problem with one parameter, the sum $p_1 + \cdots + p_\ell$ [6].

We then introduce an extension of the Closest String problem, a problem arising in computational biology. Informally, Closest String asks whether there exists a string that is "sufficiently close" to each member of a set of input strings. We modify the problem so that the input strings may be unreliable – due to transcription errors, for example – and show that this resiliency variant of Closest String called Resiliency Closest String is FPT when parameterized by the number of input strings. Our resiliency result on Closest String is a generalization of a result of Gramm *et al.* for Closest String which was proved using Lenstra's theorem [17][4].

We also introduce a resiliency variant of the scheduling problem of makespan minimization on unrelated machines. We prove that this variant is FPT when parameterized by the number of machines, the number of job types and the total expected downtime, generalizing a result of Mnich and Wiese [32] provided the jobs processing times are upper-bounded by a number given in unary.

Finally, we introduce a resilient swap bribery problem, from the field of social choice, and prove that it is FPT when parameterized by the number of candidates.

The remainder of the paper is structured in the following way. Section 2 introduces ILP resiliency and proves that it is FPT under a certain parameterization. We then apply our framework to the previously mentioned problems. We establish the fixed-parameter tractability of RDSCP parameterized by n in Section 3 and use it to provide a parameterized complexity classification of the

[3] Our approach was slightly extended in [25], which allowed the authors of [25] to settle an old conjecture in social choice theory.

[4] Although not being strictly the first problem proved to be FPT using Lenstra's theorem [38], it is considered as the one which popularized this technique [6, 8].

problem. In Section 4, we introduce a resiliency variant of the Closest String Problem and prove that it is FPT. We study resiliency variants of scheduling and social choice problems in Sections 5 and 6. We conclude the paper in Section 7, where we discuss related literature.

2 ILP resiliency

Recall that questions of ILP feasibility are typically answered by finding an integral assignment of variables x satisfying $Ax \leq b$. To save space in what follows, we will always implicitly assume that integrality constraints are part of every ILP of this paper.

In order to define resiliency for ILP we introduce another set z of variables, which can be seen as "resiliency variables". We then consider the following ILP denoted by $\mathcal{R}$:

$$\begin{aligned} Ax &\leq b && (1)\\ Cx + Dz &\leq e && (2)\\ Fz &\leq g && (3) \end{aligned}$$

The idea is that inequalities (1) and (2) represent the intrinsic structure of the problem, among which inequalities (2) represent how the resiliency variables modify the instance. Inequalities (3), finally, represent the structure of the resiliency part. The goal of ILP Resiliency is to decide whether $\mathcal{R}$ is *z-resilient*, *i.e.* whether for *any* integral assignment of variables z satisfying inequalities (3), there exists an integral assignment of variables x satisfying (1) and (2). Note that, in general, the value of x depends on z.

In $\mathcal{R}$, we will assume that all entries of matrices in the left hand sides and vectors in the right hand sides are rational numbers. The dimensions of the vectors x and z will be denoted by n and p, respectively, and the total number of rows in A and C will be denoted by m. Thus, the ILP Resiliency problem is to decide the sentence

$$\forall z \in \mathbb{Z}^p : Fz \leq g \, \exists x \in \mathbb{Z}^n : Ax \leq b \wedge Cx + Dz \leq e \ . \qquad \text{(ResILP)}$$

Our main result establishes that ILP Resiliency is FPT when parameterized by

$$\kappa(\mathcal{R}) := n + p + m, \qquad (4)$$

provided that part of the input is given in unary. To prove our main result we will use the work of Eisenbrand and Shmonin [10]. A *polyhedron* P is a set of vectors of the form $P = \{x \in \mathbb{R}^\nu : Ax \leq b\}$ for some matrix $A \in \mathbb{R}^{\mu \times \nu}$ and some vector $b \in \mathbb{R}^\mu$. A polyhedron is *rational* if $A \in \mathbb{Q}^{\mu \times \nu}$ and $b \in \mathbb{Q}^\mu$. For a rational polyhedron $Q \subseteq \mathbb{Q}^{m+p}$, define $Q/\mathbb{Z}^p := \{h \in \mathbb{Q}^m : \exists \alpha \in \mathbb{Z}^p \text{ s.t. } (h, \alpha) \in Q\}$. The Parametric Integer Linear Programming (PILP) problem takes as input a rational matrix $J \in \mathbb{Q}^{m \times n}$ and a rational polyhedron $Q \subseteq \mathbb{R}^{m+p}$, and asks whether the following expression is true:

$$\forall h \in Q/\mathbb{Z}^p \quad \exists x \in \mathbb{Z}^n : \quad Jx \leq h \ .$$

Eisenbrand and Shmonin [10, Theorem 4.2] proved that PILP is solvable in polynomial time if the number of variables $n+p$ is fixed. An interesting question arising from their result is whether this running time is a uniform or non-uniform polynomial algorithm [8, 26], and in particular for which parameters one can obtain an FPT algorithm. By looking more closely at their algorithm, one can actually obtain the following result:

Theorem 2.1 ([10, Theorem 4.2], [9])**.** PILP *can be solved in time*

$$O^*(f(n,p)(\varphi m)^{g(n,p)}),$$

where $\varphi \geq 2$ is an upper bound on the encoding length of entries of J, and f and g are some computable functions.

Complexity remark. Firstly, note that PILP belongs to the second level of the polynomial hierarchy, and is Π_2^P-complete [10]. Secondly, the polyhedron Q in Theorem 2.1 can be viewed as being defined by a system $Rh + S\alpha \leq t$, where $h \in \mathbb{R}^m$ and $\alpha \in \mathbb{Z}^p$.

Remark about the proof Theorem 2.1. The main ingredient required for the proof of Theorem 2.1 is a certain "partitioning theorem" [10, Theorem 3.3] which states that the polyhedron Q can be partitioned into t pieces with "nice" properties, that $t = O(m^{2n}\varphi^{n-1})$, where φ is a bound on the encoding length of the largest number appearing in the matrix J, and this partitioning can be carried out in time polynomial in t. The main result is then obtained by, for each piece, solving a certain mixed integer linear program whose number of variables only depends on n. The magnitude of numbers appearing in h or the definition of Q never enter the complexity bounds.

Corollary 2.2. *If all entries of J are given in unary, then* PILP *is FPT when parameterized by $n + m + p$.*

Proof. Assume that there is an upper bound $N \geq 2$ on the absolute values of entries of J and N is given in unary. Thus, the running time of the algorithm of Theorem 2.1 is $O^*(H(n,m,p)(\log N)^{g(n,p)})$, where $H(n,m,p) = f(n,p)\cdot m^{g(n,p)}$.

It is not difficult to use the Cauchy-Schwartz inequality to derive that for all integers r, s such that $0 \leq r \leq s$, we have $(\log s)^r \leq 2^{r^2/2}s^{o(1)}$ [6, Exercise 3.18]. Thus, $(\log N)^{g(n,p)} \leq 2^{g(n,p)^2/2}N^{o(1)}$, which concludes the proof. □

We now prove the main result of our framework, which will be applied in the next sections to concrete problems.

Theorem 2.3. ILP Resiliency *is FPT when parameterized by $\kappa(\mathcal{R})$ provided the entries of matrices A and C are given in unary.*

Proof. We will reduce ILP Resiliency to Parametric Integer Linear Programming. Given an instance A, C, D, F, b, e, g of ILP Resiliency, let us first define J and Q. The polyhedron Q is $(m+p)$-dimensional with its elements naturally divided into two parts (h, α), with h and α being m- and p-dimensional

vectors, respectively. We further subdivide the vector h as $h = (h_1, h_2)$ with h_1 and h_2 being m_1- and m_2-dimensional vectors, respectively, where $m_1+m_2 = m$. Let Q be defined by the (in)equalities

$$\begin{aligned} h^1 &= b \\ h^2 &= e - D\alpha \\ F\alpha &\leq g \ . \end{aligned}$$

Furthermore, let $Jx \leq h$ consist of the following two sets of inequalities:

$$\begin{aligned} Ax &\leq h^1 \\ Cx &\leq h^2. \end{aligned}$$

Having defined Q and J we obtain a PILP instance. We now claim that this PILP instance is a YES instance if and only if the input ILP RESILIENCY instance was a YES instance. Observe that the vector $h = (h_1, h_2)$ is fully determined by the integer vector α as $h_1 = b$ and $h_2 = e - D\alpha$, and because each α satisfies $F\alpha \leq g$, the variables α correspond exactly to the variables z of the ILP RESILIENCY problem. Thus,

$$\begin{aligned} &\forall h \in Q/\mathbb{Z}^p \, \exists x \in \mathbb{Z}^n : \, Jx \leq h \\ &\qquad \Leftrightarrow \forall \alpha \in \mathbb{Z}^p \text{ s.t. } (F\alpha \leq g) \, \exists x \in \mathbb{Z}^n : \, (Ax \leq b) \wedge Cx \leq e - D\alpha \\ &\qquad \Leftrightarrow \forall \alpha \in \mathbb{Z}^p \text{ s.t. } (F\alpha \leq g) \, \exists x \in \mathbb{Z}^n : \, (Ax \leq b) \wedge Cx + D\alpha \leq e \\ &\qquad\qquad \Leftrightarrow \forall z \in \mathbb{Z}^p \text{ s.t. } (Fz \leq g) \, \exists x \in \mathbb{Z}^n : \, (Ax \leq b) \wedge (Cx + Dz \leq e), \end{aligned}$$

where the first sentence is PILP, the second follows from the definition of Q, the third is by a simple rearrangement $Cx \leq e - D\alpha \Leftrightarrow Cx + D\alpha \leq e$, the last is obtained by renaming α as z and is exactly the definition of ILP RESILIENCY (equation (ResILP)).

Regarding time complexity, the dimension of x is n, the dimension of α is p and the number of inequalities of J is $m_1 + m_2 = m$, so applying Corollary 2.2 indeed yields the desired FPT algorithm. □

3 Resiliency Disjoint Set Cover

The SET COVER problem is one of the classical NP-hard problems [16]. Its input comprises a finite set U called the *universe*, a family[5] $\mathcal{F}$ of m subsets of U, and an integer t. It asks whether there is a subfamily $\mathcal{T} \subseteq \mathcal{F}$ of cardinality at most t such that $\cup_{X \in \mathcal{T}} X = U$ (such a subfamily is called a *set cover*). We may assume that $t \leq |U|$ since every set in a minimal set cover C must have an element of U not contained in any other set of C. A natural generalization of SET COVER

[5] We use the term *family* as a synonym of *multiset*, *i.e.* a family may have multiple copies of the same element. The operations of *union*, *intersection* and *deletion* on pairs $\mathcal{F}, \mathcal{F}'$ of families are defined in the natural way using $\max\{p, p'\}$, $\min\{p, p'\}$ and $\max\{0, p - p'\}$, where p and p' are the numbers of copies of the same element in $\mathcal{F}$ and $\mathcal{F}'$, respectively.

is the Disjoint Set Cover problem which takes an additional parameter d, and asks for the existence of d disjoint set covers, each of cardinality at most t. (Disjoint Set Cover was previously introduced in the literature [34, 35] in a different way: find the maximum number of disjoint (arbitrary sized) set covers. However, unlike our formulation, the previously introduced one does not extend the Set Cover problem, where the set cover is required to be of bounded size.)

In the resiliency variant of Disjoint Set Cover studied in this paper, one is given an integer s, and asks whether after the removal of any subfamily $\mathcal{S} \subseteq \mathcal{F}$ with $|\mathcal{S}| \leq s$, one still can find d disjoint set covers, each of size at most t (and disjoint from $\mathcal{S}$). More formally, we have the following:

Resiliency Disjoint Set Cover problem (RDSCP)
Input: A universe U, a multiset $\mathcal{F}$ of subsets of U, integers s, d and t.
Question: For every $\mathcal{S} \subseteq \mathcal{F}$ such that $|\mathcal{S}| \leq s$, do there exist $\mathcal{T}_1, \ldots, \mathcal{T}_d \subseteq \mathcal{F} \setminus \mathcal{S}$ such that for every $i, j \in [d]$, we have $|\mathcal{T}_i| \leq t$, $\bigcup_{X \in \mathcal{T}_i} X = U$, and $\mathcal{T}_i \cap \mathcal{T}_j = \emptyset$ for all $j \neq i$?

In the above formulation of RDSCP and henceforth, for an integer p, $[p]$ denotes the set $\{1, 2, \ldots, p\}$.

Observe that RDSCP has five natural parameters: $n = |U|$, $m = |\mathcal{F}|$, s, d, t. The study of Resiliency Disjoint Set Cover is motivated by a problem arising in access control. As we mentioned earlier, we will only focus on the parameters n, s, d and t, this choice being motivated by our application, which we discuss in more detail in the next section.

3.1 Application of RDSCP

Access control is an important topic in computer security, and deals with the idea of enforcing a policy that specifies which users are authorized to access a given set of resources. Authorization policies are frequently augmented by additional policies, articulating concerns such as separation of duty and resiliency. The Resiliency Checking problem was introduced by Li *et al.* [29] and asks whether it is always possible to form teams of users, the members of each team collectively having access to all resources, even if some users are unavailable. It is generally assumed that the number of users of an organization (which corresponds to the parameter m) is usually much larger than the set of resources [29], hence our decision to focus on the parameters n, s, d and t.

Given a set of users V and set of resources R, an *authorization policy* is a relation $VR \subseteq V \times R$; we say $v \in V$ is *authorized* for a resource r if $(v, r) \in VR$. Given an authorization policy $VR \subseteq V \times R$, an instance of the Resiliency Checking problem is defined by a resiliency policy $\mathsf{res}(P, s, d, t)$, where $P \subseteq R$, $s \geq 0$, $d \geq 1$ and $t \geq 1$. We say that VR *satisfies* $\mathsf{res}(P, s, d, t)$ if and only if for every subset $S \subseteq V$ of at most s users, there exist d pairwise disjoint subsets of

users $V_1, \ldots, V_d$ such that for all $i \in [d]$:

$$V_i \cap S = \emptyset, \tag{5}$$

$$|V_i| \leq t \text{ and } \bigcup_{v \in V_i} \{r \in R \text{ s.t. } (v, r) \in VR\} \supseteq P. \tag{6}$$

In other words, VR satisfies $\mathsf{res}(P, s, d, t)$ if we can find d disjoint groups of users, even if up to s users are unavailable, such that each group contains no more than t users and the users in each group are collectively authorized for the resources in P.

Observe that there is an immediate reduction from Resiliency Checking problem to Resiliency Disjoint Set Cover: the elements of the universe are the resources, and the sets are in one-to-one correspondence with the users, *i.e.* a set contains all resources a given user has access to. Thus, all positive results for RDSCP imply the corresponding positive results for Resiliency Checking problem.

3.2 The Parameterized Complexity Classification of RDSCP

As we observed earlier, RDSCP contains four natural parameters – n, s, d and t. In the next two subsections we prove the results leading to the classification of the parameterized complexity of RDSCP shown in Fig. 1. The FPT results follow from Theorem 3.2 which is proved in Section 3.2.1. The remaining results are obtained in Section 3.2.2.

3.2.1 Fixed-Parameter Tractability of RDSCP

In this subsection we prove that RDSCP is FPT parameterized by n. We first introduce some notation. In the following, $U, \mathcal{F}, s, d, t$ will denote an input of RDSCP, as defined previously. For all $N \subseteq U$, let $\mathcal{F}_N = \{X \in \mathcal{F} : X = N\}$. Notice that we may have $\mathcal{F}_N = \emptyset$ for some $N \subseteq U$.

Roughly speaking, the idea is that in order to construct several disjoint set covers, it is sufficient to know how many sets were picked from $\mathcal{F}_N$, for every $N \subseteq U$. (Observe that we may assume that a single set cover does not contain more than one set from each $\mathcal{F}_N$.) We first define the set of all possible set covers of the instance:

$$\mathcal{C} = \left\{ \{N_1, \ldots, N_b\} : b \leq t \wedge (N_i \subseteq U, i \in [b]) \wedge \bigcup_{i=1}^{b} N_i = U \right\}.$$

Then, for any $N \subseteq U$, we denote the set of set covers involving N by $\mathcal{C}_N$, *i.e.*

$$\mathcal{C}_N = \{c = \{N_1, \ldots, N_{b_c}\} \in \mathcal{C} : N = N_i \text{ for some } i \in [b_c]\}.$$

Observe that since we assume $t \leq n$, we have $|\mathcal{C}| = O(2^{n^2})$.

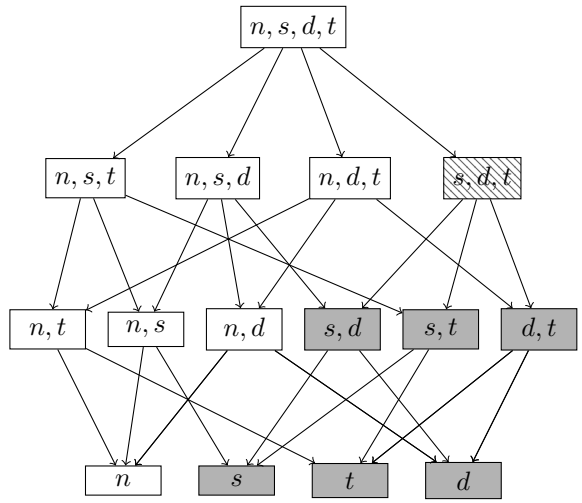

Figure 1: Schema of the parameterized complexity of RDSCP. An arrow $A \longrightarrow B$ means that A is a larger parameter than B, in the sense that the existence of an FPT algorithm parameterized by B implies the existence of an FPT algorithm parameterized by A, and, conversely, any negative result parameterized by A implies the same negative result parameterized by B (The arrows are provided only for parameters belonging to consecutive layers.)

We define an ILP $\mathcal{L}$ over the set of variables $x = (x_c : c \in \mathcal{C})$ and $z = (z_N : N \subseteq U)$, with the following system of inequalities:

$$\sum_{c \in \mathcal{C}} x_c \geq d \tag{7}$$

$$\sum_{N \subseteq U} z_N \leq s \tag{8}$$

$$\sum_{c \in \mathcal{C}_N} x_c \leq |\mathcal{F}_N| - z_N \quad \text{for every } N \subseteq U \tag{9}$$

$$0 \leq z_N \leq |\mathcal{F}_N| \quad \text{for every } N \subseteq U \tag{10}$$

$$0 \leq x_c \leq d \quad \text{for every } c \in \mathcal{C} \tag{11}$$

Recall the definition of $\kappa(\bullet)$ in (4). Observe that $\kappa(\mathcal{L})$ is upper bounded by a function of n only. The idea behind this model is to represent a family $\mathcal{S}$ of at most s sets by variables z (by deciding how many sets to take from each family $\mathcal{F}_N$, $N \subseteq U$), and to represent the disjoint set covers by variables x (by deciding how many set covers will be equal to $c \in \mathcal{C}$). Then, inequalities (9) will ensure that the set covers do not intersect with the chosen family $\mathcal{S}$. However, while we would be able to solve $\mathcal{L}$ in FPT time parameterized by n by using, *e.g.*, Lenstra's ILP Theorem, the reader might realize that doing so would not solve RDSCP directly. Nevertheless, the following result establishes the crucial link

between the resiliency of $\mathcal{L}$ and RDSCP.

Lemma 3.1. *The* RDSCP *instance is positive if and only if $\mathcal{L}$ is z-resilient.*

Proof. Let us denote by $\mathcal{L}_z$ the ILP consisting only of inequalities involving variables z, *i.e.* inequalities (8) and (10). Suppose first that the instance is positive (*i.e.*, there exist d disjoint set covers, each of size at most t and disjoint from $\mathcal{S}$, for any $\mathcal{S} \subseteq \mathcal{F}$ of size at most s), and let σ_z be an integral assignment for z such that σ_z satisfies $\mathcal{L}_z$.

Let $\mathcal{S} \subseteq \mathcal{F}$ be any familly constructed by picking, in an arbitrary manner, $\sigma_z(z_N)$ sets in $\mathcal{F}_N$, for each $N \subseteq U$. Since $\sigma_z(z_N) \leq \min\{s, |\mathcal{F}_N|\}$, this construction is indeed possible. And since $\mathcal{S}$ is a family of at most s sets, there exist d disjoint set covers $V = \{\mathcal{T}_1, \ldots, \mathcal{T}_d\}$ such that $\left(\bigcup_{i\in[d]} \mathcal{T}_i\right) \cap \mathcal{S} = \emptyset$. Then, for each $c \in \mathcal{C}$, let $\sigma_x(x_c)$ be the number of set covers of U being equal to c. Clearly we have $\sigma_x(x_c) \in \{0, \ldots, d\}$ and $\sum_{c\in\mathcal{C}} \sigma_x(x_c) = d$, and thus inequalities (7) and (11) are satisfied. Then, for all $N \subseteq U$, we may assume w.l.o.g. that $|\mathcal{T}_i \cap \mathcal{F}_N| \leq 1$ for all $i \in \{1, \ldots, d\}$. Hence $\sum_{c\in\mathcal{C}_N} \sigma_x(x_c)$ equals $\left|\left(\bigcup_{i\in[d]} \mathcal{T}_i\right) \cap \mathcal{F}_N\right|$, which is the number of sets of $\mathcal{F}_N$ involved in some set covers of U. Since $\left(\bigcup_{i\in[d]} \mathcal{T}_i\right) \cap \mathcal{S} = \emptyset$, we have $\sum_{c\in\mathcal{C}_N} \sigma_x(x_c) \leq |\mathcal{F}_N| - \sigma_z(z_N)$, and thus inequalities (9) are also satisfied for every $N \subseteq U$. Consequently, $\sigma_x \cup \sigma_z$ satisfies $\mathcal{L}$.

Conversely, let us assume that $\mathcal{L}$ is z-resilient, and let us show that the RDSCP is positive. To this end, let $\mathcal{S} \subseteq \mathcal{F}$, $|\mathcal{S}| \leq s$. For each $N \subseteq U$, define $\sigma_z(z_N) = |\mathcal{S} \cap \mathcal{F}_N|$, which is thus an integral assignment of variables z satisfying $\mathcal{L}_Z$. Hence, there exists a valid assignment σ_x such that σ_z and σ_x satisfy $\mathcal{L}$. Then, for $c = \{N_1, \ldots, N_b\} \in \mathcal{C}$, $b \leq t$, consider a family of sets $\mathcal{T}_c$, where each $\mathcal{T} \in \mathcal{T}_c$ consists of a set chosen arbitrarily in $\mathcal{S}_{N_i}$ for each $i \in [b]$. By definition of $\mathcal{C}$, each $\mathcal{T}$ is a set cover of size at most t. It is possible to construct the sets $\{T \in \mathcal{T}_c : c \in \mathcal{C}\}$ so that they satisfy the following properties:

- they are all set covers of size at most t (by construction of the set $\mathcal{C}$)
- they are d many of them (guaranteed by inequalities (7))
- they are all pairwise non-intersecting, and have an empty intersection with S (guaranteed by inequalities (9)).

In other words, the instance is positive. □

Since, as we observed earlier, $\kappa(\mathcal{L})$ is bounded by a function of n only, combining Lemma 3.1 with Theorem 2.3, we obtain the following:

Theorem 3.2. RDSCP *is FPT parameterized by n.*

3.2.2 Other Results for RDSCP

First observe that RDSCP with $s = 0$ and $d = 1$ corresponds exactly to the classical Set Cover problem, which is NP-hard and even W[2]-hard parameterized by the size of the set cover (*i.e.* t in our case). Thus, we have the following:

Proposition 3.3. RDSCP *is W[2]-hard with parameters* (s, d, t) *and para-NP-hard with parameters* (s, d).

We start by proving that RDSCP is XP when parameterized by (s, d, t).

Proposition 3.4. RDSCP *can be solved in time* $O^*(m^s t^d (m-s)^{td})$.

Proof. There are at most $\binom{m}{s} \leq m^s$ choices for $\mathcal{S}$ of size s and for each such a choice there are at most

$$\left(t\binom{m-s}{t}\right)^d \leq t^d (m-s)^{td}$$

choices for $\mathcal{T}_1, \ldots, \mathcal{T}_d$, with $|\mathcal{T}_i| \leq t$ for all $i \in [d]$. (Indeed, there are at most

$$\sum_{i=1}^{t} \binom{m-s}{i} \leq t\binom{m-s}{t}$$

choices for each $\mathcal{T}_j$.) For each such a choice we can decide whether $\mathcal{T}_1, \ldots, \mathcal{T}_d$ are disjoint and whether each of them is a set cover in polynomial time. Thus, the running time of the brute-force algorithm for RDSCP can be bounded by $O^*(m^s t^d (m-s)^{td})$. The result follows. □

Despite its simplicity, this result is actually somewhat tight: we will show any smaller parameterization leads to a para-NP-hard or para-coNP-hard problem.

We now show that the problem is coNP-hard when $d = 1$ and $t \geq 3$ is fixed, implying para-coNP-hardness of RDSCP parameterized by (d, t).

Theorem 3.5. *If* $d = 1$ *and* $t \geq 3$ *is fixed, then* RDSCP *is coNP-hard.*

Proof. We reduce from the δ-Hitting Set problem, in which we are given a ground set $V = \{v_1, \ldots, v_n\}$, a set $S = \{S_1, \ldots, S_m\}$ with $S_j \subseteq V$ and $|S_j| = \delta$ for all $j \in [m]$, and an integer k; the goal is to decide whether there is a set $C \subseteq V$ of size at most k such that $C \cap S_j \neq \emptyset$ for all $j \in [m]$. This problem is known to be NP-hard for every $\delta \geq 2$ [16].

In order to show the claimed coNP-hardness of our problem, we will construct, from an instance $\mathcal{I} = (V, S, k)$ of δ-Hitting Set, an instance $\mathcal{I}' = (U, \mathcal{F}, d, t, s)$ of RDSCP with $d = 1$ and $t = \delta + 1$ such that there exists a hitting set of size k in $\mathcal{I}$ iff there exists a set $X \subseteq \mathcal{F}$ of size at most s such that for every set cover $\mathcal{T} \subseteq \mathcal{F}$ of size at most t, we have $X \cap \mathcal{T} \neq \emptyset$.

Hence, let $\mathcal{I} = (V, S, k)$ be an instance of δ-Hitting Set defined as above. For every $j \in [m]$, fix an arbitrary ordering of S_j, which can thus be seen as a tuple $(v_{i_1}, \ldots, v_{i_\delta})$, allowing us to define $S_j[x] = v_{i_x}$ for all $x \in [\delta]$.

We now define an instance $\mathcal{I}'$ of RDSCP with universe U and family $\mathcal{F}$ of sets. The universe consists of three parts: $U = U^V \cup U^S \cup \{u^*\}$, where $U^S = \bigcup_{j=1}^{m} P^j$ with $P^j = \{p_1^j, \ldots, p_\delta^j\}$ for every $j \in [m]$, and U^V contains one element u_Q^V for every subset Q of $\delta - 1$ elements among $[n]$. The family $\mathcal{F}$ of sets consists of two parts: $\mathcal{F} = \mathcal{F}^V \cup \mathcal{F}^S$, where $\mathcal{F}^V = \{s_1^V, \ldots, s_n^V\}$ and

$\mathcal{F}^S = \{s_1^S, \ldots, s_m^S\}$. We now define the elements in each set. For every $i \in [n]$, the set s_i^V consists of $\{p_x^j : j \in [m], x \in [\delta]$ such that $S_j[x] = v_i\}$ together with all elements u_Q^V such that $i \notin Q$. For all $j \in [m]$, s_j^S consists of u^* together with $U^S \setminus P^j$. To conclude the construction of $\mathcal{I}'$, we let $t = \delta + 1$, $d = 1$, and $s = k$. Clearly this reduction can be done in polynomial time (recall that $\delta \geq 2$ is constant). An example of the reduction is shown in Figures 2 and 3.

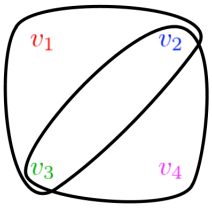

Figure 2: A 3-Hitting Set instance with a universe U of size 4, and 2 sets.

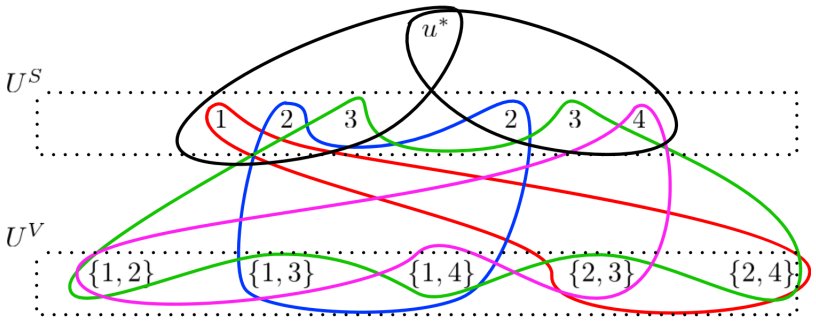

Figure 3: Schema of the reduction *w.r.t.* the instance of Figure 2. For the sake of readability, the elements of U^S are replaced by their corresponding indices in U. The color of each set of $\mathcal{F}^V$ corresponds to its vertex in the 3-Hitting Set instance.

We now show that there is a one-to-one correspondence between S (the sets of the δ-Hitting Set instance) and the set covers of $\mathcal{I}'$. More precisely, we prove the following claim:

Claim: Every set cover of I' of size at most $t = \delta + 1$ is of the form $\mathcal{T}_j = \{s_{i_1}^V, \ldots, s_{i_\delta}^V, s_j^S\}$, with $S_j = \{v_{i_1}, \ldots, v_{i_\delta}\}$.

Proof of claim: Let $\mathcal{T} \subseteq \mathcal{S}$ be of size at most t. By construction, we need at least δ sets from $\mathcal{F}^V$ to be able to include all elements from U^V (indeed, every set B of $\delta - 1$ sets of $\mathcal{F}^V$ corresponds to a subset Q_B of $[n]$, and thus B is only able to cover $U^V \setminus \{u_{Q_B}^V\}$), and we also need at least one set from $\mathcal{F}^S$ to contain u^*. Hence, $|\mathcal{T} \cap \mathcal{F}^V| = \delta$ and $\mathcal{T} \cap \mathcal{F}^S = \{s_j^S\}$ for some $j \in [m]$. Now, notice

that s_j^S contains all elements in U^S but P^j, which implies that $\mathcal{T} \cap \mathcal{S}^V$ must contain P^j. However, this can only happen if $\mathcal{T} \cap \mathcal{S}^V = \{s_{i_1}^V, \ldots, s_{i_\delta}^V\}$, where $S_j = \{v_{i_1}, \ldots, v_{i_\delta}\}$, ◁

Now, if $\mathcal{I}$ contains a hitting set $C = \{v_{i_1}, \ldots, v_{i_{|C|}}\}$ of size at most k, then because of the previous claim, $\{s_{i_1}^V, \ldots, s_{i_{|C|}}^V\}$ intersects every set cover $\mathcal{T}$ of $\mathcal{I}'$, and is of size $k = s$.

Conversely, if $X \subseteq \mathcal{F}$ is a set of size at most s intersecting every set cover $\mathcal{T}$ of $\mathcal{I}'$, then observe that since, for every $j \in [m]$, the set s_j^S only belongs to the set cover $\mathcal{T}_j$, we can assume that $s_j^S \notin X$ (if it does, we can remove this set and replace it by any other element of $\mathcal{T}_j$). Hence we have $X = \{s_{i_1}^V, \ldots, s_{i_{|X|}}^V\}$, and, as previously using the above claim, $\{v_{i_1}, \ldots, v_{i_{|X|}}\}$ is a hitting set of $\mathcal{I}$ of size at most $s = k$, which concludes the proof. □

We next settle the case of RDSCP parameterized by (s, t).

Theorem 3.6. *If $s = 0$ and $t = 4$,* RDSCP *is NP-hard.*

Proof. We reduce from the 3-Dimensional Matching problem, in which we are given three sets X, Y and Z of n elements each, a set $M \subseteq X \times Y \times Z$ of hyperedges, and an integer k. The goal is to find $M' \subseteq M$ with $|M'| \geq k$ such that for all $e, e' \in M'$ with $e \neq e'$, $e = (x, y, z)$, $e' = (x', y', z')$, we have $x \neq x'$, $y \neq y'$ and $z \neq z'$ (in that case, we will say that these two hyperedges are *disjoint*). Let $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1, \ldots, y_n\}$ $Z = \{z_1, \ldots, z_n\}$, and $M = \{e_1, \ldots, e_m\}$. We then define the following universe:

$$U = \{u_1^X, \ldots, u_m^X\} \cup \{u_1^Y, \ldots, u_m^Y\} \cup \{u_1^Z, \ldots, u_m^Z\} \cup \{u_X, u_Y, u_Z, u_*\}$$

The family $\mathcal{F}$ of sets is comprised of $\mathcal{F}_X$, $\mathcal{F}_Y$, $\mathcal{F}_Z$ and $\mathcal{F}^*$, where, for all $\omega \in \{X, Y, Z, *\}$, let $\mathcal{F}_\omega = \{s_1^\omega, \ldots, s_n^\omega\}$. For each hyperedge $e_j = \{x_{i_1}, y_{i_2}, z_{i_3}\}$, the set $s_{i_1}^X$ (resp. $s_{i_2}^Y$, $s_{i_3}^Z$) contains u_j^X (resp. u_j^Y, u_j^Z) and u_X (resp. u_Y, u_Z), and the set s_j^* consists of all elements but u_j^X, u_j^Y, u_j^Z, u_X, u_Y and u_Z. To conclude the construction, which can be done in polynomial time, we set $d = k$ (and recall that $t = 4$).

First, suppose that there exists a solution M' for the 3-Dimensional Matching problem. Without loss of generality, assume that $|M'| = k$, $M' = \{e_1, \ldots, e_k\}$, and that $e_i = (x_i, y_i, z_i)$ for all $i \in [k]$ (recall that all members of M' are pairwise disjoint). Then, observe that for all $i \in [k]$, the set s_i^* consists of all elements but u_i^X, u_i^Y, u_i^Z, u_X, u_Y and u_Z. However, s_i^X consists of u_i^X and u_X, set s_i^Y consists of u_i^Y and u_Y, and set s_i^Z is composed of u_i^Z and u_Z. Hence, $s_i^X \cup s_i^Y \cup s_i^Z \cup s_i^* = U$, and, since all members of M' are pairwise disjoint, we thus constructed d disjoint set covers of size at most 4 each.

Conversely, suppose that there exist $\mathcal{T}_1, \ldots, \mathcal{T}_d$, pairwise disjoint subfamilies of $\mathcal{F}$ such that for all $i \in [d]$, we have $|\mathcal{T}_i| = 4$ and $\bigcup_{X \in \mathcal{T}_i} X = U$. We first claim that for all $i \in [d]$, $\mathcal{T}_i$ intersects $\mathcal{F}_X$ (resp. $\mathcal{F}_Y$, $\mathcal{F}_Z$ and $\mathcal{F}_*$) on exactly one set. Indeed, otherwise, since $|\mathcal{T}_i| = 4$ and since all sets in $\mathcal{F}_X$ (resp. $\mathcal{F}_Y$, $\mathcal{F}_Z$, $\mathcal{F}_*$) only contains u_X (resp. u_Y, u_Z, u_*) among $\{u_X, u_Y, u_Z, u_*\}$, $\mathcal{T}_i$ would not be able to

cover all U. Thus, we know that for all $i \in [d]$, we have $\mathcal{T}_i = \{s^X_{i_1}, s^Y_{i_2}, s^Z_{i_3}, s^*_{i_4}\}$, for some $(i_1, i_2, i_3, i_4) \in [n] \times [n] \times [n] \times [m]$. We claim that $(x_{i_1}, y_{i_2}, z_{i_3}) = e_{i_4}$. Indeed, observe that the set $s^*_{i_4}$ contains all elements but $u^X_{i_4}$, $u^Y_{i_4}$, $u^Z_{i_4}$, u_X, u_Y and u_Z. By construction, the only way for $\mathcal{T}_i$ to cover U is that set $s^X_{i_1}$ (resp. $s^Y_{i_2}$, $s^Z_{i_3}$) contains $u^X_{i_4}$ (resp. $u^Y_{i_4}$, $u^Z_{i_4}$) or, in other words, that x_{i_1} (resp. y_{i_2}, z_{i_3}) belongs to hyperedge e_{i_4}. Thus, there exist k pairwise disjoint hyperedges in M. □

4 The Closest String Problem

In the Closest String problem, we are given a collection of k strings $s_1, \ldots, s_k$ of length L over a fixed alphabet Σ, and a non-negative integer d. The goal is to decide whether there exists a string s (of length L) such that $d_H(s, s_i) \leq d$ for all $i \in [k]$, where $d_H(s, s_i)$ denotes the Hamming distance between s and s_i (the number of positions in which s and s_i differ). If such a string exists, then it will be called a *d-closest string*. The problem was first introduced by Lanctot *et al.* [27] and proved to be NP-complete in [14, 27]. Gramm *et al.* [17] proved that the problem is FPT when parameterized by either d or k. It is common to represent an instance of the problem as a matrix C with k rows and L columns (*i.e.* where each row is a string of the input); hence, in the following, the term *column* will refer to a column of this matrix.

The motivation for studying resiliency with respect to this problem is the introduction of experimental errors, which may change the input strings [36]. While a solution of the Closest String problem tests whether the input strings are consistent, a resiliency variant asks whether these strings will remain consistent after some small changes. In the Resiliency Closest String problem we allow at most m changes to appear anywhere in the matrix C. Equivalently, we ask for the existence of a closest string for all matrices $\bar{C}$ which are in Hamming distance at most m from C.

Resiliency Closest String (RCS) Input: C, a $k \times L$ matrix of elements of Σ, $d \in \mathbb{N}$, $m \leq kL$. Question: For every $\bar{C}$, $k \times L$ matrix of elements of Σ such that the Hamming distance of C and C' is at most m, does $\bar{C}$ admit a d-closest string?

As is standard, we first preprocess the instance in order to shrink the size of the alphabet to at most k. This is easy because Hamming distance is measured column-wise and it is thus sufficient to replace each column $c \in \Sigma^k$ of C with a column $c' \in [k]^k$ satisfying $c_i = c_j \Leftrightarrow c'_i = c'_j$ for all $i, j \in [k]$, obtaining a reduced instance matrix $C' \in [k]^{k \times L}$. Denote by π^c the mapping satisfying $\pi^c(i) = c'_i$ for all $i \in [k]$, and assume that π^c is defined for all $c \in [k]^k$ even if C does not contain any column c.

However, some caution is due because it is not immediately clear how adversarial changes to C (*i.e.* the matrix $\bar{C}$) translate to C'. In other words,

is C resilient if and only if C' is? Let us argue that it is. Observe that, for any $\bar{C} \in \Sigma^{k \times L}$, $d_H(C, \bar{C}) = \sum_{\ell=1}^{L} d_H(c^\ell, \bar{c}^\ell)$. Now consider C' and a matrix $\bar{C}' \in [k]^{k \times L}$ obtained by same process, *i.e.* $\bar{c}'_i = \pi^{\bar{c}}(i)$ for every column $\bar{c}$ of $\bar{C}$. Because for every $\ell \in [L]$, $d_H(c^\ell, \bar{c}^\ell) = d_H((c')^\ell, (\bar{c}')^\ell)$, we have that $d_H(C, \bar{C}) = \sum_{\ell=1}^{L} d_H(c^\ell, \bar{c}^\ell) = \sum_{\ell=1}^{L} d_H((c')^\ell, (\bar{c}')^\ell) = d_H(C', \bar{C}')$.

We continue by grouping columns into types by denoting the set of all possible columns $T = [k]^k$. Using the observation that $|T| \leq k^k$, Gramm *et al.* [17] proved that Closest String is FPT parameterized by k, using an ILP with a number of variables depending on k only, and then applying Lenstra's theorem. Our approach builds on theirs.

Let $\#_t$ be the number of columns of type t in C'. For two types $t, \bar{t} \in T$ let $\delta(t, \bar{t})$ be their Hamming distance. Let $z_{t,\bar{t}}$, for all $t, \bar{t} \in T$, be a variable meaning "how many columns of type t in C' are changed to type $\bar{t}$ in $\bar{C}'$" (we allow $t = \bar{t}$). Thus we have the following constraints:

$$\sum_{\bar{t} \in T} z_{t,\bar{t}} = \#_t \qquad \forall t \in T \tag{12}$$

$$\sum_{t, \bar{t} \in T} \delta(t, \bar{t}) z_{t,\bar{t}} \leq m \tag{13}$$

These constraints clearly capture all possible scenarios of how the input strings can be modified in at most m places (while staying in the reduced alphabet $\Sigma' = [k]$, but this is sufficient as argued previously). Then let $\bar{\#}_t$ be a variable meaning "how many columns of $\bar{C}'$ are of type t", and let $x_{t,\varphi}$ represent the number of columns of type t in $\bar{C}'$ whose corresponding character in the solution is set to φ. Finally, for every $t \in T$, let $[t_i \neq \varphi]$ be 1 if $t_i \neq \varphi$ and 0 otherwise. As the remaining constraints correspond to our formulation of ILP Resiliency, we have:

$$\sum_{t \in T} z_{t,\bar{t}} = \bar{\#}_{\bar{t}} \qquad \forall \bar{t} \in T \tag{14}$$

$$\sum_{\varphi \in \Sigma} x_{t,\varphi} = \bar{\#}_t \qquad \forall t \in T \tag{15}$$

$$\sum_{t \in T} \sum_{\varphi \in \Sigma} [t_i \neq \varphi] x_{t,\varphi} \leq d \qquad \forall i \in [k] \tag{16}$$

This is the standard ILP for Closest String [17], except that $\bar{\#}_t$ are now variables, and there exists a solution x exactly when there is a string at distance at most d from the modified strings given by the variables $\bar{\#}$. Let $\mathcal{L}$ denote the ILP composed of constraints (12), (13), (14), (15) and (16). Finally, let $\mathcal{Z}$ denote variables $z_{t,\bar{t}}$ and $\bar{\#}_t$ for every $t, \bar{t} \in T$.

Lemma 4.1. *The* Resiliency Closest String *instance is satisfiable if and only if $\mathcal{L}$ is $\mathcal{Z}$-resilient.*

Proof. Constraints involving variables $\mathcal{Z}$ are (12), (13) and (14). Suppose first that the instance is satisfiable, and let $\sigma_{\mathcal{Z}}$ be an integral assignment for $\mathcal{Z}$.

We construct $\bar{C}'$ from C' by turning, in an arbitrary way, $\sigma_{\mathcal{Z}}(z_{t,\bar{t}})$ columns of type t to columns of type $\bar{t}$. By constraints (12), $\bar{C}'$ is well-defined, and by constraints (13), the Hamming distance between C' and $\bar{C}'$ is at most m. Then, by constraint (14), matrix $\bar{C}'$ contains $\sigma_{\mathcal{Z}}(\bar{\#}_t)$ columns of type t, for every $t \in T$. Since the instance is satisfiable, there exists a d-closest string s of $\bar{C}'$. For $t \in T$ and $\varphi \in \Sigma$, define $\sigma_x(x_{t,\varphi})$ to be the number of columns of type t in $\bar{C}'$ whose corresponding character in s is φ. Since, as we said previously, $\bar{C}'$ has exactly $\sigma_{\mathcal{Z}}(\bar{\#}_t)$ columns of type t, constraint (15) is satisfied for every $t \in T$. Then, since s is a d-closest string for $\bar{C}'$, constraint (16) is also satisfied.

Conversely, suppose that $\mathcal{L}$ is $\mathcal{Z}$-resilient and consider $\bar{C}'$, a $k \times L$ matrix with elements from $\Sigma' = [k]$ such that the Hamming distance of C' and $\bar{C}'$ is at most m. In polynomial time, we construct $\sigma_{\mathcal{Z}}(z_{t,\bar{t}})$ for every $t, \bar{t} \in T$ such that (12) and (14) are satisfied. By definition of $\bar{C}'$, constraint (13) is satisfied. Thus, there exists an integral assignment σ_x satisfying (15) and (16). We now construct s as a string having, for every column type $t \in T$ in $\bar{C}'$, $\sigma_x(x_{t,\varphi})$ occurrences of character φ, for every $\varphi \in \Sigma$ (columns chosen arbitrarily among those of type t in $\bar{C}'$). Because of constraint (15), and since $\bar{\#}_t$ is the number of columns of type t in $\bar{C}'$, s is well-defined. Finally, observe that constraint (16) ensures that s is a d-closest string of $\bar{C}'$, which concludes the proof. □

It remains to observe that for the above system of constraints $\mathcal{L}$, $\kappa(\mathcal{L})$ is bounded by a function of k (since $|T| = O(2^{k \log_2 k})$). We thus obtain the following result:

Theorem 4.2. RCS *is FPT parameterized by* k.

5 Resilient Scheduling

Makespan minimization on unrelated machines is a fundamental scheduling problem [37]. We have m machines and n jobs, and each job has a vector of processing times with respect to the machines $p_j = (p_j^1, \ldots, p_j^m)$, $j \in [n]$. If the vectors p_j and $p_{j'}$ are identical for two jobs j, j', we say these jobs are of the same *type*. Here we consider the case when m and the number of types θ are parameters and the input is given as θ numbers $n_1, \ldots, n_\theta$ of job multiplicities and (with a slight abuse of notation) θ vectors $p_t = (p_t^1, \ldots, p_t^m)$, $t \in [\theta]$, of processing times of a job of type t. A *schedule* is an assignment of jobs to machines. For a particular schedule, let n_t^i be the number of jobs of type t assigned to machine i. Then, the *completion time* of machine i is $C^i = \sum_{t \in [\theta]} p_t^i n_t^i$ and the largest C^i is the *makespan* of the schedule, denoted $C_{\max}$.

The problem is long known to be NP-hard already when all machines are identical (cf. Garey and Johnson [16, SS9]). The parameterization by θ and m might seem very restrictive, but note that when m alone is a parameter, the problem is W[1]-hard even when the machines are identical (*i.e.* the number of types $\theta = 1$) and the job lengths are given in unary [21]. Also, Asahiro et al. [1] show that it is strongly NP-hard already for *restricted assignment* when there is a number p_j for each job such that for each machine i, $p_j^i \in \{p_j, \infty\}$ and all

$p_j \in \{1, 2\}$ and for every job there are exactly two machines where it can run. Mnich and Wiese [32] proved that the problem is FPT with parameters θ and m.

A natural way to introduce resiliency is when we consider unexpected delays due to repairs, fixing software bugs, etc., but we have an upper bound K on the total expected downtime. We assume that the execution of jobs can be resumed after the machine becomes available again, but cannot be moved to another machine, that is, we assume preemption but not migration. Under these assumptions it is the total downtime for each machine that matters, not when the downtime actually occurs. Given m machines, n jobs, $C_{\max} \in \mathbb{N}$ and $K \in \mathbb{N}$, we say that a scheduling instance has *K-tolerant makespan $C_{\max}$* if, for every $d_1, \ldots, d_m \in \mathbb{N}$ such that $\sum_{i=1}^{m} d_i \leq K$, there exists a schedule where each machine $i \in [m]$ finishes by the time $C_{\max} - d_i$. We obtain the following problem:

RESILIENCY MAKESPAN MINIMIZATION ON UNRELATED MACHINES
Input: m machines, θ job types $p_1, \ldots, p_\theta \in \mathbb{N}^m$, job multiplicities $n_1, \ldots, n_\theta$, $K \in \mathbb{N}$ and $C_{\max} \in \mathbb{N}$.
Question: Does this instance have a K-tolerant makespan $C_{\max}$?

Let x_t^i be a variable expressing how many jobs of type t are scheduled to machine i. We have the following constraints, with the first constraint describing the feasible set of delays, and the subsequent constraints guaranteeing that every job is scheduled on some machine and that every machine finishes by time $C_{\max} - d_i$:

$$\sum_{i=1}^{m} d_i \leq K \tag{17}$$

$$\sum_{i=1}^{m} x_t^i = n_t \qquad \forall t \in [\theta] \tag{18}$$

$$\sum_{t=1}^{\theta} x_t^i p_t^i \leq C_{\max} - d_i \qquad \forall i \in [m] \tag{19}$$

Theorem 2.3 and the system of constraints above implies the following result related to the above-mentioned result of Mnich and Wiese [32].

Theorem 5.1. RESILIENCY MAKESPAN MINIMIZATION ON UNRELATED MACHINES *is FPT when parameterized by θ and m and with $p_{\max} \leq N$, where $p_{\max} = \max_{t \in [\theta], i \in [m]} p_t^i$, for some number N given in unary.*

Proof. We recall the constraints (17)–(19) and denote the system by $\mathcal{L}$. We prove that the instance is satisfiable (*i.e.* has a K-tolerant makespan $C_{\max}$) if and only if $\mathcal{L}$ is d-resilient. Suppose first that the instance is satisfiable, and let σ_d be an integral assignment of variables d_i satisfying constraint (17), that is, we have delays $\sigma_d(d_1), \ldots, \sigma_d(d_m)$ with total delay at most K. Thus, there exists a

schedule where each machine $i \in [m]$ finishes by time $C_{\max}$ with expected delay $\sigma_d(d_i)$. By defining, for every machine $i \in [m]$ and every type $t \in [\theta]$, $\sigma_x(x_t^i)$ to be the number of jobs of type t assigned to machine i, we obtain an integral assignment for variables x_t^i satisfying constraints (18) and (19). That is, $\mathcal{L}$ is d-resilient.

Conversely, suppose that $\mathcal{L}$ is d-resilient, and let us consider delays $d_1, \ldots, d_m$ with total delay at most K, or, equivalently, an integral assignment σ_d of variables d satisfying constraint (17). Since $\mathcal{L}$ is d-resilient, there exists an assignment σ_x of variables x_t^i satisfying (18) and (19). Using the same arguments as above, there exists a schedule where each machine $i \in [m]$ finishes by time $C_{\max} - d_i$.

Finally, observe that $\kappa(\mathcal{L})$ is bounded by a function of θ and m, and all coefficients appearing in $\mathcal{L}$ are bounded by N given in unary (recall the definition of $\kappa(\bullet)$ in (4)). □

6 Resilient Swap Bribery

The field of computational social choice is concerned with computational problems associated with voting in elections. Swap Bribery, where the goal is to find the cheapest way to bribe voters such that a preferred candidate wins, was introduced by Elkind, Faliszewski and Slinko [11] who showed its NP-hardness and has subsequently received considerable attention [13]. The problem may serve as a model not only of actual bribery, but also many other kinds of campaigning, as a better measure of the notion of "margin of victory", or as a tool to actually *detect* bribery. It is natural to consider the case where an adversarial counterparty first performs their bribery which is unknown *a priori*, but an estimate on its budget is known. The question becomes whether, for each such bribery, it is possible, within a given budget, to bribe the election such that our preferred candidate still wins. The number of candidates is a well studied parameter [2, 7]. In this section we will show that the resilient variant of Swap Bribery with unit costs (unit costs are a common setting, cf. Dorn and Schlotter [7]) is FPT using our framework. Let us now give formal definitions.

Elections. An election $E = (C, V)$ consists of a set C of m candidates $c_1, \ldots, c_m$ and a set V of voters (or votes). Each voter i is a linear order $\succ_i$ over the set C. For distinct candidates a and b, we write $a \succ_i b$ if voter i prefers a over b. We denote by $\mathrm{rank}(c, i)$ the position of candidate $c \in C$ in the order $\succ_i$. Without loss of generality, assume that the preferred candidate is c_1.

Swaps. Let (C, V) be an election and let $i \in V$ be a voter. A *swap* $\gamma = (a, b)_i$ in preference order $\succ_i$ means to exchange the positions of a and b in $\succ_i$; denote the resulting order by $\succ_i^\gamma$; the *cost* of $(a, b)_i$ is $\pi_i(a, b)$ (in the problem studied in this paper, we have $\pi_i(a, b) = 1$ for every voter i and candidates a, b). A swap $\gamma = (a, b)_i$ is *admissible in* $\succ_i$ if $\mathrm{rank}(a, i) = \mathrm{rank}(b, i) - 1$. A set Γ of swaps is *admissible in* $\succ_i$ if they can be applied sequentially in $\succ_i$, one after the other, in some order, such that each one of them is admissible. Note that the obtained vote, denoted by $\succ_i^\Gamma$, is independent from the order in which the swaps

of Γ are applied. We also extend this notation and the notion of admissibility for applying swaps in several votes and denote it V^Γ. A *bribery* Γ is an admissible set of swaps and its cost is the sum of the costs of all swaps (in the unit case studied here, the cost is simply the cardinality of Γ).

Voting rules. A voting rule $\mathcal{R}$ is a function that maps an election to a subset of candidates, the set of winners. We will show our example for rules which are scoring protocols, but following the framework of the so-called "election systems described by linear inequalities" [7] it is easily seen that the result below holds for many other voting rules. With a scoring protocol $s = (s_1, \ldots, s_m) \in \mathbb{N}^m$, a voter i gives s_1 points to their most preferred candidate, s_2 points to their second most preferred candidate and so on. The candidate with most points wins.

RESILIENCY UNIT SWAP BRIBERY
Input: An election $E = (C, V)$ with each swap of unit cost and with a scoring protocol $s \in \mathbb{N}^m$, the adversary's budget B_a, our budget B.
Question: For every adversarial bribery Γ_a of cost at most B_a, is there a bribery Γ of cost at most B such that $E = (C, (V^{\Gamma_a})^\Gamma)$ is won by c_1?

Theorem 6.1. RESILIENCY UNIT SWAP BRIBERY *with a scoring protocol is FPT when parameterized by the number of candidates m.*

Proof. A standard way of looking at an election when the number of candidates m is a parameter is as given by *multiplicities* of *voter types*: there are at most $m!$ total orders on C, so we count the voters and output numbers $n_1, \ldots, n_{m!}$. Observe that for two orders $\succ, \succ'$, the admissible set of swaps Γ such that $\succ' = \succ^\Gamma$ is uniquely given as the set of pairs (c_i, c_j) for which either $c_i \succ c_j \wedge c_j \succ' c_i$ or $c_j \succ c_i \wedge c_i \succ' c_j$ (cf. [11, Proposition 3.2]). Thus it is possible to define the number of swaps $\pi(i, j)$, for $i, j \in [m!]$, required to bribe a voter of type i to become of type j (since every swap is of unit cost, $\pi(i, j)$ does not depend on individual voters). Moreover, we can extend our notation $\mathrm{rank}(c, i)$ to denote the position of c in the order of type i.

As in our approach to RESILIENCY CLOSEST STRING, let z_{ij}, for all $i, j \in [m!]$, be a variable representing the number of voters of type i bribed by the adversary to become of type j, and let y_i, $i \in [m!]$, represent the election $E = (C, V^{\Gamma_a})$ after the adversarial bribery. The following constraints describe all possible adversarial briberies and resulting elections:

$$\sum_{j=1}^{m!} z_{ij} = n_i \qquad \forall i \in [m!]$$

$$\sum_{i=1}^{m!} z_{ij} = y_j \qquad \forall j \in [m!]$$

$$\sum_{i=1}^{m!} \sum_{j=1}^{n} \pi(i, j) z_{ij} \leq B_a$$

The rest of the ILP is standard; variables x_{ij} will describe the second bribery in the same way as z_{ij} and variables w will describe the election after this bribery, on which we will impose a constraint which is satisfied when c_1 is a winner:

$$\sum_{j=1}^{m!} x_{ij} = y_i \qquad \forall i \in [m!]$$

$$\sum_{i=1}^{m!} x_{ij} = w_j \qquad \forall j \in [m!]$$

$$\sum_{i=1}^{m!}\sum_{j=1}^{m!} \pi(i,j)x_{ij} \leq B$$

$$\sum_{k=1}^{m} \sum_{i:\mathrm{rank}(c_1,i)=k} w_i s_k \geq \sum_{k=1}^{m} \sum_{i:\mathrm{rank}(c_j,i)=k} w_i s_k \qquad \forall j = 2,\ldots,m$$

We remark that the last constraint may be modified to contain a strict inequality ">" which would ensure that c_1 is the unique winner. Both formulations are valid because the custom when studying bribery is to leave matters related to tie breaking to be dealt with separately. For this reason we do not discuss tie-breaking further.

The rest of the proof is similar to the proof of Lemma 4.1. □

7 Discussion and Open Problems

For some time, Lenstra's theorem was the only approach in parameterized algorithms and complexity based on integer programming. Recently other tools based on integer programming have been introduced: the use of Graver bases for the n-fold integer programming problem [19], the use of ILP approaches in kernelization [12], or, conversely, kernelization results for testing ILP feasibility [20], and an integer quadratic programming analogue of Lenstra's theorem [30]. Our approach is a new addition to this powerful arsenal.

However, there still remain powerful tools from the theory of integer programming which, surprisingly, have not found applications in the design of parameterized algorithms. For one example take a result of Hemmecke, Köppe and Weismantel [18] about *2-stage stochastic integer programming with n scenarios*. They describe an FPT algorithm (which builds on a deep structural insight) for solving integer programs with a certain block structure and bounded coefficients.

Another important research direction is the optimality program in parameterized algorithms pioneered by Marx [31]. Many of the prototypical uses of Lenstra's algorithm lead to FPT algorithms which have a double-exponential (*i.e.* $2^{2^{k^{O(1)}}}$) dependency on the parameter, such as the algorithms for Closest String [17] or Swap Bribery [7]. Very recently, Knop et al. [24] showed that many of these ILP formulations have a particular format which is solvable

exponentially faster than by Lenstra's algorithm, thus bringing down the dependency on the parameter down to single-exponential, cf. e.g. [24, Theorems 4 and 5, and Section 4.5] This leads us to wonder what is the true complexity of, e.g., Resiliency Closest String? All we can say is that the complexity of our algorithm is at best double-exponential but probably worse (depending on the complexity of parametric ILP in fixed dimension). Is this the best possible, or does a single-exponential algorithm exist?

References

[1] Yuichi Asahiro, Jesper Jansson, Eiji Miyano, Hirotaka Ono, and Kouhei Zenmyo. Approximation algorithms for the graph orientation minimizing the maximum weighted outdegree. In *AAIM*, LNCS 4508, pages 167–177, 2007.

[2] Robert Bredereck, Piotr Faliszewski, Rolf Niedermeier, Piotr Skowron, and Nimrod Talmon. Elections with few candidates: Prices, weights, and covering problems. In *ADT'15*, LNCS 9346, pages 414–431, 2015.

[3] Jason Crampton, Gregory Gutin, Martin Koutecký, and Rémi Watrigant. Parameterized resiliency problems via integer linear programming. In *CIAC*, LNCS 10236, pages 164–176, 2017.

[4] Jason Crampton, Gregory Gutin, and Remi Watrigant. A multivariate approach for checking resiliency in access control. In *AAIM 2016*, LNCS 9778, pages 173–184, 2016.

[5] Jason Crampton, Gregory Gutin, and R'emi Watrigant. Resiliency policies in access control revisited. In *Proc. SACMAT'16*, pages 101–111. ACM, 2016.

[6] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[7] Britta Dorn and Ildikó Schlotter. Multivariate complexity analysis of swap bribery. *Algorithmica*, 64(1):126–151, 2012.

[8] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

[9] Friedrich Eisenbrand. Personal communication, 2016.

[10] Friedrich Eisenbrand and Gennady Shmonin. Parametric integer programming in fixed dimension. *Math. Oper. Res*, 33(4):839–850, 2008.

[11] Edith Elkind, Piotr Faliszewski, and Arkadii Slinko. Swap bribery. In *Proc. SAGT 2009*, volume 5814 of *LNCS*, pages 299–310, 2009.

[12] Michael Etscheid, Stefan Kratsch, Matthias Mnich, and Heiko Röglin. Polynomial kernels for weighted problems. *J. Comput. Syst. Sci.*, 84:1–10, 2017.

[13] Piotr Faliszewski and Jörg Rothe. Control and bribery in voting. In Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia, editors, *Handbook of Computational Social Choice*, pages 146–168. Cambridge University Press, 2016.

[14] Moti Frances and Ami Litman. On covering problems of codes. *Theory Comput. Syst.*, 30(2):113–119, 1997.

[15] András Frank and Éva Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987.

[16] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[17] Jens Gramm, Rolf Niedermeier, and Peter Rossmanith. Fixed-parameter algorithms for CLOSEST STRING and related problems. *Algorithmica*, 37(1):25–42, 2003.

[18] Raymond Hemmecke, Matthias Köppe, and Robert Weismantel. Graver basis and proximity techniques for block-structured separable convex integer minimization problems. *Mathematical Programming*, 145(1-2):1–18, 2014.

[19] Raymond Hemmecke, Shmuel Onn, and Lyubov Romanchuk. n-fold integer programming in cubic time. *Mathematical Programming*, 137(1):325–341, 2013.

[20] Bart M. P. Jansen and Stefan Kratsch. A structural approach to kernels for ILPs: Treewidth and total unimodularity. In *ESA*, pages 779–791, 2015.

[21] Klaus Jansen, Stefan Kratsch, Dániel Marx, and Ildikó Schlotter. Bin packing with fixed number of bins revisited. *Journal of Computer and System Sciences*, 79(1):39–49, 2013.

[22] Ravi Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, August 1987.

[23] Ravi Kannan. Test sets for integer programs, $\forall\exists$ sentences. In William J. Cook and Paul D. Seymour, editors, *Polyhedral Combinatorics, Proceedings of a DIMACS Workshop, 1989*, volume 1 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 39–48. DIMACS/AMS, 1990.

[24] Dušan Knop, Martin Koutecký, and Matthias Mnich. Combinatorial n-fold integer programming and applications. *arXiv preprint arXiv:1705.08657*, 2017.

[25] Dusan Knop, Martin Koutecký, and Matthias Mnich. A unifying framework for manipulation problems. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pages 256–264, 2018.

[26] Dexter C. Kozen. *Theory of Computation*. Springer, 2006.

[27] J. Kevin Lanctot, Ming Li, Bin Ma, Shaojiu Wang, and Louxin Zhang. Distinguishing string selection problems. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '99, pages 633–642, 1999.

[28] Hendrik W. Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.

[29] Ninghui Li, Manesh Tripunitara, and Qihua Wang. Resiliency policies in access control. *ACM Trans. Inf. Syst. Secur.*, 12(4), 2009.

[30] Daniel Lokshtanov. Parameterized integer quadratic programming: Variables and coefficients. *CoRR*, abs/1511.00310, 2015.

[31] Dániel Marx. What's next? future directions in parameterized complexity. In *The Multivariate Algorithmic Revolution and Beyond*, pages 469–496. Springer, 2012.

[32] Matthias Mnich and Andreas Wiese. Scheduling and fixed-parameter tractability. *Mathematical Programming*, 154(1):533–562, 2014.

[33] Danny Nguyen and Igor Pak. Complexity of short Presburger arithmetic. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of STOC 2017*, pages 812–820, 2017.

[34] Ashwin Pananjady, Vivek Kumar Bagaria, and Rahul Vaze. The online disjoint set cover problem and its applications. In *2015 IEEE Conference on Computer Communications, INFOCOM 2015*, pages 1221–1229, 2015.

[35] Ashwin Pananjady, Vivek Kumar Bagaria, and Rahul Vaze. Optimally approximating the coverage lifetime of wireless sensor networks. *IEEE/ACM Trans. Netw.*, 25(1):98–111, 2017.

[36] Pavel Pevzner. *Computational Molecular Biology: An Algorithmic Approach*. MIT Press, 2000.

[37] Michael L. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer Publishing Company, Incorporated, 3rd edition, 2008.

[38] András Sebő. Integer plane multiflows with a fixed number of demands. *J. Comb. Theory, Ser. B*, 59(2):163–171, 1993.