Resolving Infeasibility of Linear Systems: A Parameterized Approach

Alexander Göke
Universität Bonn, Bonn, Germany
Technische Universität Hamburg, Hamburg, Germany
alexander.goeke@uni-bonn.de

Lydia Mirabel Mendoza Cadena
Eötvös Loránd University, Budapest, Hungary
lmmendoza@ciencias.unam.mx

Matthias Mnich
Universität Bonn, Bonn, Germany
Technische Universität Hamburg, Hamburg, Germany
mmnich@uni-bonn.de

Abstract

Deciding feasibility of large systems of linear equations and inequalities is one of the most fundamental algorithmic tasks. However, due to inaccuracies of the data or modeling errors, in practical applications one often faces linear systems that are *infeasible*.

Extensive theoretical and practical methods have been proposed for post-infeasibility analysis of linear systems. This generally amounts to detecting a feasibility blocker of small size k, which is a set of equations and inequalities whose removal or perturbation from the large system of size m yields a feasible system. This motivates a parameterized approach towards post-infeasibility analysis, where we aim to find a feasibility blocker of size at most k in fixed-parameter time $f(k) \cdot m^{\mathcal{O}(1)}$.

On the one hand, we establish parameterized intractability (W[1]-hardness) results even in very restricted settings. On the other hand, we develop fixed-parameter algorithms parameterized by the number of perturbed inequalities and the number of positive/negative right-hand sides. Our algorithms capture the case of Directed Feedback Arc Set, a fundamental parameterized problem whose fixed-parameter tractability was shown by Chen et al. (STOC 2008).

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Graph algorithms analysis

Keywords and phrases Infeasible subsystems, linear programming, fixed-parameter algorithms

Digital Object Identifier 10.4230/LIPIcs.IPEC.2019.17

Funding *Alexander Göke*: Supported by DAAD with funds of the Bundesministerium für Bildung und Forschung (BMBF) and by DFG project MN 59/1-1.
Matthias Mnich: Supported by DAAD with funds of the Bundesministerium für Bildung und Forschung (BMBF) and by DFG project MN 59/4-1.

SPONSORED BY THE Federal Ministry of Education and Research

Acknowledgements We thank an anonymous reviewer of an earlier version for a suggestion on run time improvements.

1 Introduction

Solving systems of linear equations and inequalities constitutes an algorithmic task of fundamental importance. The data that is used in these systems though may be subject to inaccuracies and uncertainties, and therefore may lead to systems which are infeasible. Another source of infeasibility may be modeling errors, or simply incompatibility of constraints. Infeasibility itself allows for little conclusions; for a large system of millions of inequalities, infeasibility may stem from a very small subset of data. A natural question is therefore to detect the smallest number of changes which must be made to a given system in order to make it feasible. The analysis of infeasible linear systems has been extensively investigated [1, 2, 4, 11, 12, 13, 28, 31]; we refer to the book by Chinneck [13] for an overview.

Formally, the Minimum Feasibility Blocker (MinFB) problem takes as input a system $\mathcal{S}$ of linear inequalities $Ax \leq b$ and asks for a smallest subset $\mathcal{I}$ such that $\mathcal{S} \setminus \mathcal{I}$ is *feasible.* As $\mathcal{I}$ "blocks" the feasibility of $\mathcal{S}$, we refer to $\mathcal{I}$ as a *feasibility blocker*; we avoid calling $\mathcal{I}$ a "solution", to avoid confusion with the solution of the linear system $\mathcal{S} \setminus \mathcal{I}$. Further note that instead of removing the set $\mathcal{I}$ of inequalities, we can equivalently perturb the right-hand sides $b_{\mathcal{I}}$ of the inequalities in $\mathcal{I}$; that is, we can increase the b-values of inequalities in $\mathcal{I}$ to a value such that the perturbed system becomes feasible. Of course, when talking about feasibility, we have to specify over which field, and our choice here is the field $\mathbb{Q}$. Over this field, MinFB is NP-hard [33]; as the feasibility of a linear system can be tested in polynomial time (e.g., by the ellipsoid method [24]) the MinFB problem is NP-complete. Thus, there is a simple XP-algorithm testing every possible feasibility blocker of size at most k. Due to its importance, the MinFB problem has been thoroughly investigated from several different viewpoints, including approximation algorithms [1, 28], polyhedral combinatorics [31], heuristics [11], mixed-integer programming [15], and hardness of approximation [2].

Here we take a new perspective on the MinFB problem, based on parameterized complexity. In parameterized complexity, the problem input of size n is additionally equipped with one or more integer parameters k and one measures the problem complexity in terms of both n and k. The goal is to solve such instances by *fixed-parameter algorithms*, which run in time $f(k) \cdot n^{\mathcal{O}(1)}$ for some computable function f. The motivation is that fixed-parameter algorithms can be practical for small parameter values k even for inputs of large size n, provided that the function f exhibits moderate growth. This contrasts them with algorithms that require time $n^{f(k)}$, which cannot presumed to be practical for large input sizes n. To show that such impractical run times are best possible, a common approach is to show the problem to be W[1]-hard; a standard hypothesis in parameterized complexity is that no W[1]-hard problem admits a fixed-parameter algorithm. For background on parameterized complexity, we refer to the book by Cygan et al. [18].

For the MinFB problem, arguably the most natural parameter is the minimum size k of a feasibility blocker $\mathcal{I}$. The motivation for this choice of parameter is that in applications, we are interested in *small* feasibility blockers $\mathcal{I}$; e.g., Chakravarti [9] argues that a feasibility blocker "with too large a cardinality may be hard to comprehend and may not be very useful for post-infeasibility analysis." Guillemot [25] explicitly posed the question of resolving the parameterized complexity of MinFB; he conjectured that the problem is fixed-parameter tractable parameterized by the size of a minimum feasibility blocker for matrices with at most 2 non-zero entries per row.

Another motivation for our approach comes from the fact that MinFB captures one of the most important problems in parameterized complexity, namely Directed Feedback Arc Set (DFAS): given a digraph G, decide if G admits a directed feedback arc set of size at most k, which is a set F such that $G - F$ is an acyclic digraph (DAG). It was a long-standing open question whether DFAS admits a fixed-parameter algorithm parameterized by the size k of the smallest directed feedback arc set, until Chen et al. [10] gave an algorithm with run time $4^k k! n^{\mathcal{O}(1)}$. The currently fastest algorithm for DFAS runs in time $4^k k! k \cdot \mathcal{O}(n+m)$, and is due to Lokshtanov et al. [29]. It is not difficult to give a parameter-preserving reduction from DFAS to MinFB: for every arc (u, v) of the digraph G that serves as input to DFAS we add the inequality $x_u - x_v \leq -1$ to the linear system $Ax \leq b$. Directed feedback arc sets F

of G are then mapped to feasibility blockers of the same size by removing the constraints corresponding to arcs in F, and vice-versa. Note that the constraint matrix A arising this way is totally unimodular, and each row has exactly two non-zero entries, one $+1$ and one -1. Totally unimodular matrices A whose every row has at most two non-zero entries, one $+1$ and one -1, are known as *difference constraints*; testing feasibility of systems of difference constraints has been investigated extensively [32, 35] due to their practical relevance most notably in temporal reasoning. It is therefore interesting to know whether the more general MinFB problem also admits a fixed-parameter algorithm for parameter k, even for special cases like totally unimodular matrices A (where testing feasibility is easy).

Another case of interest for MinFB is when the constraint matrix A has bounded treewidth, where the treewidth of A is defined as the treewidth of the bipartite graph that originates from assigning one vertex to every row and every column of A and connecting any two vertices by an edge whose corresponding entry in A is non-zero. Fomin et al. [21] gave a fast algorithm for MinFB with constraint matrices of bounded treewidth for the setting $k = 0$, i.e. checking for feasibility without deleting any constraints. At the same time, Bonamy et al. [8] showed that DFAS – a special case of MinFB – is fixed-parameter tractable parameterized by the treewidth of the underlying undirected graph of the input digraph[1]. So the questions arise whether Fomin et al.'s algorithm can be extended to arbitrary values of k, or whether Bonamy et al.'s algorithm can be extended from DFAS to MinFB.

One of the main currently unresolved questions around DFAS is whether it admits a *polynomial compression*. That is, one seeks an algorithm that, given any directed graph G and integer k, in polynomial time computes an instance I of a decision problem Π whose size is bounded by some polynomial $p(k)$, such that G admits a feedback arc set of size at most k if and only if I is a "yes"-instance of Π. The question for a polynomial compression has been stated numerous times as an open problem [5, 19, 17, 30]; from the algorithms by Chen et al. [10] and Lokshtanov et al. [29] only an *exponential* bound on the size of I follows. On the other hand, parameterized complexity provides tools such as cross-composition to rule out the existence of such polynomials $p(k)$ modulo the non-collapse of the polynomial hierarchy; we refer to Bodlaender et al. [6] for background. Given the elusiveness of this problem, we approach the (non-)existence of polynomial compression for DFAS from the angle of the more general MinFB problem.

1.1 Our results

We first show that the MinFB problem is strictly more general than DFAS, even for totally unimodular matrices, assuming that $\mathsf{FPT} \neq \mathsf{W}[1]$.

▶ **Theorem 1.** *The* MinFB *problem is* $\mathsf{W}[1]$*-hard parameterized by the minimum size k of a feasibility blocker, even for difference constraints and right-hand sides $b \in \{\pm 1\}^m$.*

Theorem 1 therefore disproves (assuming $\mathsf{FPT} \neq \mathsf{W}[1]$) the conjecture of Guillemot [25] that finding the minimum number of unsatisfied equations or inequalities is fixed-parameter tractable for linear systems with at most two variables per equation or inequality.

Given this strong parameterized intractability result, we resort to identifying tractable fragments (or classes of instances) of MinFB. In particular, we look for algorithms which

[1] The algorithm of Bonamy et al. [8] is stated for the vertex deletion problem, but it can be modified to work for DFAS as well. Note that the standard reduction from DFAS to the vertex deletion problem which preserves the solution size does not necessarily result in a digraph whose underlying undirected graph has bounded treewidth even if the DFAS instance has this property.

solve fragments of MinFB which capture the fundamental DFAS problem. As relevant parameters, we identify the number b_+ of positive entries in the right-hand side vector b, as well as the number b_- of negative entries in b. This choice comes from the fact that the case of $b_+ = 0$ generalizes the DFAS problem, whereas the case of $b_- = 0$ is always feasible as the all-0 vector is a trivial solution for a system of the form $Ax \leq b$.

Our positive algorithmic results for these parameters are as follows:

▶ **Theorem 2.** *There is an algorithm that solves* MinFB *for systems $\mathcal{S}$ of m difference constraints over n variables and right-hand sides $b \in \{\pm 1\}^m$ in time $2^{\mathcal{O}(k^3+b_++k\log b_+)} \cdot n^{\mathcal{O}(1)}$.*

▶ **Theorem 3.** *There is an algorithm that solves* MinFB *for systems $\mathcal{S}$ of m difference constraints over n variables and right-hand sides $b \in \{\pm 1\}^m$ in time $(k+1)(b_-)^{k+1}\mathcal{O}(nm)$.*

Armed with these fixed-parameter algorithms for parameters $k + b_+$ and $k + b_-$, it is time to consider the question of polynomial compressions for those tractable fragments of MinFB. Such polynomial compression would be particularly interesting, as it could be a step towards obtaining a polynomial compression for DFAS (where $b_+ = 0$); so a polynomial compression for MinFB for parameter k or $k + b_+$, even for node-arc incidence matrices, would imply a polynomial compression for DFAS (as the reduction from DFAS to MinFB does not increase the parameter).

Interestingly, we can actually rule out a polynomial compression for MinFB parameterized by $k + b_-$:

▶ **Theorem 4.** *Assuming* $\mathsf{NP} \not\subseteq \mathsf{coNP}/poly$, MinFB *does not admit a polynomial compression when parameterized by $k + b_-$ even for systems A of difference constraints and right-hand sides $b \in \{\pm 1\}^m$.*

The most intriguing open question arising from this result is whether our hardness result can be strengthened to rule out a polynomial compression for MinFB parameterized by $k + b_+$.

As mentioned, Fomin et al. [21] give an algorithm that solves MinFB for constraint matrices of bounded treewidth for $k = 0$. And Bonamy et al. [8] give an algorithm that solves the special case of MinFB known as DFAS for constraint matrices of bounded treewidth. Here we show that, somewhat surprisingly, MinFB is NP-hard even for constraint matrices of *constant* pathwidth (which are a subclass of matrices with constant treewidth).

▶ **Theorem 5.** *The* MinFB *problem is* NP*-hard even for constraint matrices of pathwidth 6.*

Due to space constraints, proofs of statements marked by ($\star$) are deferred to the full version of this paper.

1.2 Related work

In fundamental work, Arora, Babai, Stern and Sweedyk [2] considered the problem of removing a smallest set of *equations* to make a given system of linear equations feasible over $\mathbb{Q}$. They gave strong inapproximability results, showing that finding *any* constant-factor approximation is NP-hard. Berman and Karpinski [4] gave the first (randomized) polynomial-time algorithm with sublinear approximation ratio for this problem.

Giannopolous, Knauer and Rote [22] considered the "dual" of MinFB from a parameterized point of view: namely, in MaxFS we ask for a largest subsystem of an n-dimensional linear system $\mathcal{S}$ which is feasible over $\mathbb{Q}$. They showed that deciding whether a feasible subsystem of at least ℓ inequalities in $\mathcal{S}$ exists is W[1]-hard parameterized by $n + \ell$, even when $\mathcal{S}$ consists of equations only.

For systems of equations over *finite* fields, finding minimum feasibility blockers has been considered from a parameterized perspective. In particular, over the binary field $\mathbb{F}_2$, Crowston et al. [16] prove W[1]-hardness even if each equation has exactly three variables and every variable appears in exactly three equations; they further give a fixed-parameter algorithm for the case where each equation has at most two variables.

2 Preliminaries

Throughout, we work with finite and loop-less directed graphs G, whose vertex set we denote by $V(G)$ and arc set by $A(G)$. For a vertex $v \in V(G)$, denote by $\delta^-(v)$ the incoming arcs of v, i.e., arcs of the form (w, v) for some $w \in V(G)$. Likewise, denote by $\delta^+(v)$ the outgoing arcs of v. A *walk* W in G is a sequence of vertices $W = (v_0, v_1, \ldots, v_\ell)$ such that $(v_i, v_{i+1}) \in A(G)$ for $i = 0, \ldots, \ell - 1$. We call ℓ the *length* of the walk. A walk is *closed* if $v_0 = v_\ell$. If all vertices are distinct we call the walk a *path*, if all except v_0 and v_ℓ are distinct we call it a cycle. For two walks W, R where the last vertex of W equals the first vertex of R, let $W \circ R$ be the *concatenation* of W and R, which is the sequence of all vertices in W followed by all vertices in R except the first. Our directed graphs G often come with arc weights $w : A(G) \to \mathbb{Q}$; the *weight* of a cycle C in G is then equal to the sum of its arc weights. In that spirit, we call a cycle *negative* (*non-negative*, *positive*) if its weight is *negative* (*non-negative*, *positive*). A *shortest* path or cycle is a path or cycle of minimum length. Note that "shortest" does not refer to the weight of a cycle.

For our hardness results we will use two different kinds of hardness. The first one is W[1]-hardness which under the standard assumption $\mathsf{W}[1] \neq \mathsf{FPT}$ implies that there is no fixed-parameter tractable algorithm for problems of this type. The other hardness considers compression: A *polynomial compression* of a language L into a language Q is a polynomial-time computable mapping $\Phi : \Sigma^* \times \mathbb{N} \to \Sigma^*$, $\Phi((x,k)) \mapsto y$ such that $((x,k) \in L \Leftrightarrow y \in Q)$ and $|y| \leq k^{\mathcal{O}(1)}$ for all $(x,k) \in \Sigma^* \times \mathbb{N}$. Many natural parameterized problems do not admit polynomial compressions, under the hypothesis that $\mathsf{NP} \subsetneq \mathsf{coNP}/poly$.

Both types of hardness can be transferred to other problems by "polynomial parameter transformations", which were first proposed by Bodlaender et al. [7].

▶ **Definition 6.** *Let Σ be an alphabet. A* polynomial parameter transformation (PPT) *from a parameterized problem $\Pi \subseteq \Sigma^* \times \mathbb{N}$ to a parameterized problem $\Pi' \subseteq \Sigma^* \times \mathbb{N}$ is a polynomial-time computable mapping $\Phi : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}, (x,k) \mapsto (x',k')$, such that $k' = k^{\mathcal{O}(1)}$, and $(x,k) \in \Pi \Leftrightarrow (x',k') \in \Pi'$ for all $(x,k) \in \Sigma^* \times \mathbb{N}$. Two parameterized problems are* parameter-equivalent *if there are PPTs in both directions and the transformations additionally fulfil that $k' = k$.*

Note that polynomial parameter transformations are transitive. Further, a PPT from Π to Π' together with a polynomial compression for Π' yields a polynomial compression for Π. This can be used to rule out polynomial compressions:

▶ **Proposition 7** ([27]). *Let Π, Π' be parameterized problems. If there is a polynomial parameter transformation from Π to Π' and Π admits no polynomial compression, then neither does Π'.*

Next, we formally define the MinFB problem. Here and throughout the rest of the paper, we denote by $a_{i,\bullet}$ the i^{th} row of the matrix A.

Minimum Feasibility Blocker (MinFB)	Parameter: k
Input: A coefficient matrix $A \in \mathbb{Q}^{m\times n}$, a right-hand side vector $b \in \mathbb{Q}^m$ and an integer k.	
Task: Find a set $\mathcal{I} \subseteq \{1, \dots, m\}$ of size at most k such that $(a_{i,\bullet} \cdot x \leq b_i)_{i\in\{1,\dots,m\}\setminus\mathcal{I}}$ is feasible for some $x \in \mathbb{Q}^n$.	

By multiplying rows of A and the corresponding entries of b with (-1) the MinFB problem also covers in a parameter equivalent way the case where some inequalities are of the type $a_{i,\bullet}x \geq b_i$ if there is no sign restriction on the entries of A and b.

Furthermore, equations of the form $a_{i,\bullet} \cdot x = b_i$ can be written as the two inequalities $a_{i,\bullet} \cdot x \leq b_i$ and $-a_{i,\bullet} \cdot x \leq -b_i$ (equivalent to $a_{i,\bullet} \cdot x \geq b_i$). In a feasible solution x^* ignoring such an equation, at most one of the above inequalities is violated. Thus, the MinFB problem with equations can be reduced to the formulation of the MinFB as formulated above without changing the parameter. Conversely though, MinFB as presented above in general cannot be expressed by the MinFB problem having only equations. However, if we allow additionally inequalities with only one variable or require all variables to be non-negative, those are representable by adding slack variables and (if necessary) splitting variables into two non-negative parts x_i^+ and x_i^-.

For a graph H, a *tree decomposition* (*path decompostion*) is a pair $(T, \mathcal{B})$ where T is a tree (path) and $\mathcal{B}$ a collection of bags $B_v \subseteq V(H)$, each bag corresponding to some node $v \in V(T)$. The bags have the property that for any edge of H both its endpoints appear in some common bag in $\mathcal{B}$, and for each vertex $v \in V(H)$ the bags containing v form a subtree of T. The *width* of $(T, \mathcal{B})$ is defined as the largest bag size of $\mathcal{B}$ minus one. The *treewidth* (*pathwidth*) of H is the minimum width over all tree (path) decompositions of H.

3 Parameterized Intractability of Minimum Feasibility Blocker

In this section we show W[1]-hardness of the MinFB problem by giving a reduction from the Bounded Edge Directed (s,t)-Cut problem. This problem takes as input a digraph G, vertices s, t, and integers $k, \ell \in \mathbb{N}$, and asks for a set $X \subseteq E(G)$ of size at most k such that $G - X$ contains no s-t-paths of length at most ℓ. Golovach and Thilikos [23] proved its parameterized intractability for parameter k:

▶ **Proposition 8** ([23]). Bounded Edge Directed (s,t)-Cut *is* W[1]*-hard when parameterized in k even for the special case where G is a DAG.*

The Bounded Edge Directed (s,t)-Cut was also considered by Fluschnik et al. [20]. They showed that Bounded Edge Directed (s,t)-Cut does not admit a kernel of size polynomial in k and ℓ, assuming NP $\not\subseteq$ coNP/*poly*, even for acyclic input digraphs. In fact, their construction allows for a stronger result, ruling out a polynomial compression:

▶ **Proposition 9** ([20]). *Assuming* NP $\not\subseteq$ coNP/*poly,* Bounded Edge Directed (s,t)-Cut *does not admit a polynomial compression in $k + \ell$ even when G is a DAG.*

To get to the MinFB problem, we first consider as an intermediate step the Directed Small Cycle Transversal problem: given a directed graph G and integers k, ℓ, the task is to find a set $X \subseteq E(G)$ of size at most k such that $G - X$ contains no cycles of length at most ℓ. Fluschnik et al. [20] showed that Directed Small Cycle Transversal does not admit a kernel of size polynomial in k and ℓ, unless NP $\subseteq$ coNP/*poly*. Again their result can be strengthened to not admitting a polynomial compression. They further observed that

Directed Small Cycle Tranversal admits a simple branching algorithm that runs in time $\mathcal{O}(\ell^k \cdot n \cdot (n+m))$. Here we argue that a dependence on both parameters k and ℓ is necessary for fixed-parameter tractability:

▶ **Lemma 10.** *There is a polynomial parameter transformation from* Bounded Edge Directed (s,t)-Cut *in DAGs with parameter k (parameter ℓ, parameter $k+\ell$) to* Directed Small Cycle Transversal *with parameter k (resp. ℓ, resp. $k+\ell$).*

This even holds if Directed Small Cycle Transversal *is restricted to instances where there are vertices $s,t \in V(G)$ such that all cycles of G consist of an s-t-path and an arc of the form (t,s).*

Proof. Let (G,s,t,k,ℓ) be a Bounded Edge Directed (s,t)-Cut instance where G is a DAG. As G is a DAG, it admits a topological ordering $v_1,\ldots,v_{|V(G)|}$ of its vertices, so that there are no arcs (v_i,v_j) for $j<i$. Without loss of generality, let $v_1=s$ and $v_{|V(G)|}=t$, as vertices before s or after t in a topological ordering are never part of any s-t-path.

We now create a digraph G' from G by adding $k+1$ parallel arcs $a_1,\ldots,a_{k+1}$ from t to s. Then every cycle in G' consists of an s-t-path and an arc a_i, as G was acyclic. Set $\ell'=\ell+1$. Then (G',k,ℓ') is an instance of Directed Small Cycle Transversal. The above transformation can be done in polynomial time and the parameter increases by at most one (depending on whether ℓ is part of the parameter).

It remains to show that (G',k,ℓ') is a "yes"-instance of Directed Small Cycle Transversal if and only if (G,s,t,k,ℓ) is a "yes"-instance of Bounded Edge Directed (s,t)-Cut.

For the forward direction, let X' be a solution to (G',k,ℓ'). Consider $X=X'\setminus\{a_1,\ldots,a_{k+1}\}$. For sake of contradiction, suppose that $G-X$ contains an s-t-path P of length at most ℓ. As $|X'|\le k$, it can not contain all arcs a_i. Without loss of generality, $a_1\notin X'$. Then P followed by a_1 is a cycle in $G'-X'$ of length at most $\ell+1=\ell'$ – a contradiction to X' being a solution to (G',k,ℓ').

For the reverse direction, let X be a solution to (G,s,t,k,ℓ). Then X is also a solution to (G',k,ℓ') by the following argument. Suppose, for sake of contradiction, that $G'-X$ contains a cycle C of length at most ℓ'. By the structure of G', C consists of an s-t-path P in $G-X$ and an arc a_i. Then $|P|=|C|-1\le\ell$, contradicting that X is a solution to (G,k,ℓ). ◀

Now we introduce another cycle deletion problem, this time on arc-weighted digraphs.

Negative Directed Feedback Arc Set (Negative DFAS) Parameter: k
Input: A digraph G, a weight function $w: A(G)\to\mathbb{Q}$ and an integer k.
Task: Find a set $X\subseteq A(G)$ of size at most k such that $G-X$ has no negative cycles.

▶ **Lemma 11.** *There is a PPT from* Directed Small Cycle Transversal *on instances where every cycle uses an arc of type (t,s) when parameterized by k (by $k+\ell$) to* Negative DFAS *parameterized by k (resp. $k+w_-$, where w_- is the number of arcs with negative weight); this even holds in the case where $w: A(G)\to\{\pm1\}$.*

Proof. We start with a Directed Small Cycle Transversal instance (G,k,ℓ) as described in the lemma. Let $a_1,\ldots,a_p$ be the arcs of the form (t,s). For any $p\ge k+1$ there is always an arc which survives the deletion of some arc set of at most k elements. So we can assume $p\le k+1$ as deleting superfluous arcs does not change the solution. Set $A_{+1}=A(G)\setminus\{a_1,\ldots,a_p\}$. Now replace the a_i by mutually disjoint (except for s and t)

paths P_i of length ℓ. Call the resulting directed graph G' and let $A_{-1} = \cup_{i=1}^{p} A(P_i)$. Finally, define $w(a) = 1$ for $a \in A_{+1}$ and $w(a) = -1$ for $a \in A_{-1}$. As $A(G') = A_{-1} \uplus A_{+1}$, the function $w : A(G') \to \{-1, +1\}$ is well defined.

The instance (G', w, k) has the required form. Also the transformation can be made in polynomial time. As k remains unchanged and $w_- = |A_{-1}| = \ell \cdot p \leq \ell \cdot (k+1)$ is bounded by a polynomial in $k + \ell$, the parameter restrictions of PPTs are fulfilled. It remains to prove that (G', w, k) is a "yes"-instance of Negative DFAS if and only if (G, k, ℓ) is a "yes"-instance of Directed Small Cycle Transversal.

For the forward direction, let X' be a solution of (G', w, k). Let X be the set where every arc of X' which is part of some P_i is replaced by a_i (and duplicates are removed). Clearly, $|X| \leq |X'| \leq k$. Suppose there is a cycle C of length at most ℓ in $G - X$. Then C contains a unique arc a_j. Let C' be the cycle in G' resulting from the replacement of a_j by P_j in C. Then C' is also in $G' - X'$ by choice of X and contains at most $\ell - 1$ arcs in A_{+1} and ℓ arcs in A_{-1}. This yields the contradiction $w(C') = |A(C') \cap A_{+1}| - |A(C') \cap A_{-1}| \leq \ell - 1 - \ell = -1$.

For the reverse direction, let X be a solution of (G, k, ℓ). Let X' the set X where every arc a_i is replaced by the first arc of P_i. By definition of G', $X' \subseteq A(G')$ and $|X'| = |X| \leq k$ holds. Now suppose there is a cycle C' in $G' - X'$ with $w(C') < 0$. As the paths P_i are mutually disjoint (except the end vertices) every inner vertex of each P_i has in-degree and out-degree one. Thus, if there is an arc of some P_i inside C' the whole path P_i is. Replace each such P_i by a_i to obtain a cycle C. By construction of G' and X', this cycle C is in $G - X$. Each cycle in G and therefore also C contains exactly one arc of type a_i. Therefore, C' contains exactly one path P_i and from $0 > w(C') = |A(C') \cap A_{+1}| - |A(C') \cap A_{-1}| = |A(C') \cap A_{+1}| - \ell$ we get that $|A(C') \cap A_{+1}| < \ell$. As $|A(C') \cap A_{+1}|$ is integral we can sharpen the bound to $|A(C') \cap A_{+1}| \leq \ell - 1$. By $A(C) = (A(C') \cap A_{+1}) \uplus \{a_i\}$, we get that $|A(C)| = |A(C') \cap A_{+1}| + 1 \leq \ell$ – a contradiction. ◀

▶ **Theorem 12.** *The* Negative DFAS *problem and the* MinFB *problem for difference constraints are parameter-equivalent. Additionally, the equivalence can be constructed such that there is a one-to-one correspondence between constraints and arc weights with $b_a = w(a)$.*

Proof. Let (G, w, k) be a Negative DFAS problem instance, and let $n = |V(G)|$ and $m = |A(G)|$. Fix an arbitrary order $v_1, \ldots, v_n$ of the vertices of G and $\alpha_1, \ldots, \alpha_m$ of the arcs of G. As matrix A we take the incidence matrix of G which is defined as matrix $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$ with entries $a_{i,j} = +1$ for $\alpha_i \in \delta^-(v_j)$, $a_{i,j} = -1$ for $\alpha_i \in \delta^+(v_j)$, and $a_{i,j} = 0$ otherwise. Furthermore, let $b_i = w(\alpha_i)$. The resulting tuple (A, b, k) is an instance of the MinFB problem with A being a matrix of difference constraints.

The construction is bijective by the following reverse construction: Define a directed graph on n vertices $v_1, \ldots, v_n$, then for every constraint $a_{i,\bullet} \cdot x \leq b_i$ add an arc as follows: Let j^- be the unique index with $a_{i,j^-} = -1$, and let j^+ be the unique index with $a_{i,j^+} = +1$. Add an arc $\alpha = (v_{j^+}, v_{j^-})$ with weight $w(\alpha) = b_i$ to the current digraph. Let G be the resulting digraph after all arcs are added. Then (G, w, k) is the constructed Negative DFAS instance. It is easy to verify that this indeed reverses the first construction.

Now we want to compare solutions of both problems. Intuitively, deleted constraints and arcs have an one to one correspondence, but we will formally prove the equivalence here.

For this we need the notion of "feasible potentials". A *feasible potential* (with respect to G and w) is a function $\pi : V(G) \to \mathbb{Q}$ such that, for every arc $a = (x, y) \in A(G)$, the following inequality holds: $w(a) - \pi(x) + \pi(y) \geq 0$. It is well-known that a weighted digraph has a feasible potential if and only if it has no cycle of negative weight (see, for example, the book of Schrijver [34, Theorem 8.2]).

In the following, for each $X \subseteq A(G)$ denote by $X_{\mathcal{I}}$ the corresponding indices of the constraints and vice-versa. Then the following equivalences hold:

$(G - X, w)$ contains no negative cycles with respect to w.

$\Leftrightarrow$ $(G - X, w)$ has a feasible potential $\pi : V(G) \to \mathbb{Q}$.

$\Leftrightarrow$ There is some $\pi : V(G) \to \mathbb{Q}$ such that $\pi(u) \leq \pi(v) + w(\alpha)$ for all $\alpha = (u, v) \in A(G) \setminus X$.

$\Leftrightarrow$ There is some $x \in \mathbb{Q}^{V(G)}$ such that $x_u - x_v \leq w(\alpha)$ for all $\alpha = (u, v) \in A(G) \setminus X$.

$\Leftrightarrow$ There is some $x \in \mathbb{Q}^n$ such that $a_{i,\bullet} \cdot x \leq b_i$ for all $i \in \{1, \dots, m\} \setminus X_{\mathcal{I}}$.

Furthermore, as X and $X_{\mathcal{I}}$ have the same cardinality, the last statement is equivalent to the statement that X is a solution to (G, w, k) if and only if $X_{\mathcal{I}}$ is a solution to (A, b, k). ◀

Concatenating all reductions above, we obtain the following corollary:

▶ **Corollary 13.** *There is a PPT from* Bounded Edge Directed (s,t)-Cut *parameterized by k (in $k + \ell$) to* MinFB *parameterized by k (in $k + b_-$). This even holds for instances of* MinFB *where A is a system of difference constraints and $b \in \{\pm 1\}^m$.*

This corollary yields our two hardness results:

Proof of Theorem 1. With Proposition 7 we can use the W[1]-hardness of Bounded Edge Directed (s,t)-Cut from Proposition 8 and the PPT from Corollary 13 to get the W[1]-hardness of MinFB. Also the structure of the instance follows from this. ◀

Proof of Theorem 4. This follows by combining Proposition 9 and Corollary 13 with the help of Proposition 7. The structure of the MinFB instance follows as in Theorem 1. ◀

4 Fixed-parameter Algorithms for Systems of Difference Constraints

In this section we develop fixed-parameter algorithm for MinFB for constraint matrices A of difference constraints and right-hand sides $b \in \{\pm 1\}^m$. Our first algorithm takes as parameters the number k of constraints that must be deleted from A to make the system feasible and the number w_- of negative entries in the b-vector; our second algorithm takes as parameters k and the number w_+ of positive entries in b. The naming convention for w_+, w_- stems from Theorem 12 and the 1-to-1 correspondence between b and w.

Recall that the DFAS problem corresponds to the case when $w_+ = 0$, as in this case all arcs have negative weight. In fact, our algorithm makes oracle calls to an algorithm for the more general problem Subset DFAS, in which we are given a digraph G, an arc set $U \subseteq A(G)$ and an integer k, and seek a set $X \subseteq A(G)$ of at most k arcs that intersects each cycle containing some arc of U. The Subset DFAS problem was shown to be fixed-parameter tractable for parameter k by Chitnis et al. [14].

▶ **Proposition 14** ([14]). Subset DFAS *is solvable in time $2^{\mathcal{O}(k^3)} \cdot n^{\mathcal{O}(1)}$.*

For both algorithms we need a subroutine finding a shortest negative cycle C in a given digraph. Recall that *shortest* is defined in terms of number of arcs. Negative cycles can be found with the Moore-Bellman-Ford algorithm (cf. Bang-Jensen and Gutin [3, Sect. 2.3.4]), which runs in time $\mathcal{O}(nm)$. That algorithm can be modified to find a shortest negative cycle of length at most ℓ in time $\mathcal{O}(\ell nm)$; such modification is well-known or at least an easy exercise.

▶ **Lemma 15** ($\star$). *There is an algorithm that, given a digraph G, arc weights $w : A(G) \to \mathbb{Q}$ and an integer $\ell \in \mathbb{N}$, in time $\mathcal{O}(\ell nm)$ either finds a shortest negative cycle C of length at most ℓ in G, or decides that none exists.*

Algorithm 1 SimpleNegativeCycleDeletion.

Input : A digraph G, arc weights $w : A(G) \to \mathbb{Q}$ and $k \in \mathbb{N}$.
Output : A set $S \subseteq A(G)$ of at most k arcs such that $G - S$ has no negative cycles or `false` if no such set exists.

1 **for** *every* $k_-, k_+ \in \mathbb{Z}_{\geq 0}$ *with* $k_- + k_+ = k$ **do**
2 **for** *every subset* S_- *of* $w^{-1}(-1) \subseteq A(G)$ *with* $|S_-| = k_-$ **do**
3   $S_0 = S_-$.
4   **while** $G - S_0$ *contains a negative cycle* C **and** $|S_0| \leq k$ **do**
5   Branch on adding an arc $a \in A(C)$ with $w(a) = 1$ to S_0.
6   **if** $|S_0| \leq k$ **then**
7   **return** S_0.
8 **return** `false`.

4.1 Few negative right-hand sides

We are now ready to give our fixed-parameter algorithm for MinFB for constraint matrices A of difference constraints and right-hand sides $b \in \{\pm 1\}^m$, parameterized by the size k of the deletion set and the number b_- of negative entries in b. For the rest of this subsection we will use Theorem 12 and only work on the Negative DFAS problem with $w : A(G) \to \{\pm 1\}$. We give a simple algorithm for the Negative DFAS problem with $w : A(G) \to \{\pm 1\}$, parameterized by k and b_-. Pseudocode of the algorithm can be found in Algorithm 1. The algorithm first guesses the negative arcs in the solution and then recursively branches on the positive arcs of a negative cycle (as long as such a cycle exists). The algorithm keeps track of the already deleted arcs in the set S_0.

▶ **Lemma 16.** *Algorithm 1 is correct and runs in time $(k+1)w_-^{k+1}\mathcal{O}(nm)$.*

Proof. The algorithm works in three steps: The first **for** loop guesses how many of the deleted arcs have weight -1 with the variable k_-. The second **for** loop then iterates over every (k_-)-element subset of these arcs. The last procedure then tries to fix the negative cycle by only deleting arcs of weight $+1$.

As we enumerate all choices of k_- and the subsets of negative arcs we only need to argue correctness for the last procedure. The procedure only returns a value other than "false" if this value is a solution. So we only need to argue that if there is a solution we will find it. So let S be a solution and S_0 contain all k_- arcs of weight -1 in S. We get S_0 by correct guessing of the **for** loops. If $S_0 = S$, then $|S_0| \leq k$ and the graph $G - S_0$ will contain no negative cycle, so we correctly return S_0. Otherwise, there is a negative cycle C in $G - S_0$. By choice of S_0 there must be an arc $a \in A(C) \cap S$ with weight $+1$. Branching on all possible choices in line 5, one of the branches must have found the right arc and added it to S_0. Thus, in each recursive call we find an additional element of S until $S_0 = S$.

For the runtime, the first main observation to be made is that any negative cycle C has length at most $2w_- - 1$. Furthermore, at most $w_- - 1$ arcs of it can have weight $+1$. Therefore, we can check by Lemma 15 for the existence of a negative cycle in time $\mathcal{O}(w_- nm)$ and iterate over all arcs with weight $+1$ in a cycle in time $\mathcal{O}(w_-)$. As S_0's size increases by one with each branching and we stop (correctly) if $|S_0| > k$ at most $k - k_-$ recursive calls are made by the branching. Thus, the runtime of the inner branching procedure for fixed S_- is at most $(w_-)^{k-k_-+1}\mathcal{O}(nm)$. The inner **for** loop enumerates, for a fixed value of k_-, at most $w_-^{k_-}$ sets S_-. This **for** loop is executed $k+1$ many times by the outer **for** loop. So the algorithm runs in time $(k+1)w_-^{k_-} \cdot (w_-)^{k-k_-+1}\mathcal{O}(nm) = (k+1)(w_-)^{k+1}\mathcal{O}(nm)$. ◀

Proof of Theorem 3. By Theorem 12 we can construct, in polynomial time, an instance of Negative DFAS that has the same parameters $k + w_-$, and such that the original instances is a "yes"-instance if and only if the constructed instance is. Lemma 16 then allows us to solve this instance in the claimed time. ◀

4.2 Few positive right-hand sides

In the second part of this section we will study Negative DFAS with weight functions $w : A(G) \to \{\pm 1\}$ when parameterized by k and w_+. The main observation for our algorithm is made in the following lemma:

▶ **Lemma 17** ($\star$). *Let G be a digraph with arc weights $w : A(G) \to \{\pm 1\}$. Then either G has a negative cycle of length at most $2(w_+)^2 + 2w_+$, or every negative cycle C has some arc $a \in A(C)$ that lies only on negative cycles of G.*

This lemma forms the basis of our algorithm, Algorithm 2. First, the algorithm checks for negative cycles with up to $2(w_+)^2 + 2w_+$ arcs. It then guesses the arc contained in a solution like in our first algorithm, Algorithm 1. Afterwards, we are left with a digraph without small cycles. We now identify the set U of arcs which are not part of a non-negative cycle. Then, for any solution S to Negative DFAS, $G - S$ may not contain a cycle on which an arc of U lies, as such a cycle would be negative by definition of U. Likewise, any negative cycle in G has some arc in U by the previous lemma. Thus, it remains to solve an instance of Subset DFAS for input (G, U, k).

Algorithm 2 NegativeCycleDeletion.

```
Input  : A digraph G with arc weights w : A(G) → ℚ and k ∈ ℕ.
Output : A set S ⊆ A(G) of at most k arcs such that G − S has no negative cycle, or
         false if no such set exists.
if k < 0 then
  | return false.
if there is some negative cycle C of length at most 2(w+)^2 + 2w+ in G then
  | Branch on deleting an arc of C and try to solve with k − 1.
else
  | Identify the set U of all arcs which do not lie on a non-negative cycle.
  | return SubsetDirectedFeedbackArcSet (G, U, k).
```

Before we can prove correctness and runtime we have to show how we can detect the set U of all arcs which lie only on negative cycles. We first argue that this problem is NP-hard even for weights $w : A(G) \to \{\pm 1\}$. To this end, we provide a reduction from the Hamiltonian s-t-Path problem, which for a digraph H and vertices $s, t \in V(H)$ asks for an s-t-path in H visiting each vertex of H exactly once. Its NP-hardness was shown by Karp [26]. The reduction works as follows: Take the original digraph H and two vertices $s, t \in V(H)$ which we want to test for the existence of an Hamiltonian path starting in s and ending in t. Add a path P of length $n - 1$ from t to s to the graph. Assign weight $+1$ to each arc of H, and weight -1 to each arc of P. Then an arc of P lies on a cycle of non-negative length if and only if there is an Hamiltonian s-t-path in H.

However, for this construction of weights w we have $w_+ \in \Omega(n)$. We will now show that the task is indeed fixed-parameter tractable when parameterized by w_+. For that, the main observation is that every non-negative cycle has length at most $2w_+$. We now consider the

WEIGHTED ℓ-PATH problem: given a digraph G with arc weights $w : A(G) \to \mathbb{Q}$ and numbers $W \in \mathbb{Q}, \ell \in \mathbb{N}$, the task is to find a path of length exactly ℓ and weight at least W in G. Zehavi [36] gave a fast algorithm for WEIGHTED ℓ-PATH, based on color-coding techniques.

▶ **Proposition 18** ([36]). WEIGHTED ℓ-PATH *can be solved in time* $2^{\mathcal{O}(\ell)} \cdot \mathcal{O}(m \log n)$ *on digraphs with* n *vertices and* m *arcs.*

So given an arc $a = (s, t)$ one can enumerate all path sizes ℓ from 1 to $2w_+ - 1$ and ask whether there is a t-s-path of length ℓ of weight at least $-w(a)$. This way one can detect a non-negative cycle containing a.

▶ **Corollary 19.** *Let* G *be a digraph, let* $w : A(G) \to \{\pm 1\}$ *and* $(s, t) \in A(G)$*. Then one can detect in time* $2^{\mathcal{O}(w_+)} \cdot m \log n$ *if* $a = (s, t)$ *is part of some non-negative cycle* C*.*

Finally, to argue the correctness and runtime of Algorithm 2, we prove the following:

▶ **Lemma 20** ($\star$). *Algorithm 2 is correct and runs in time* $2^{\mathcal{O}(k^3 + w_+ + k \log w_+)} \cdot n^{\mathcal{O}(1)}$.

Proof of Theorem 2. By Theorem 12 we can construct, in polynomial time, an instance of NEGATIVE DFAS that has the same parameters $k + w_-$, and such that the original instances is a "yes"-instance if and only if the constructed instance is. Lemma 20 then allows us to solve this instance in the claimed time. Regarding the run time, note that $m \leq (k+1)n^2 \in 2^{\mathcal{O}(\log k)} \cdot n^{\mathcal{O}(1)}$. ◀

5 NP-Hardness for Incidence Matrices of Constant Pathwidth

In this section we show that MINFB is NP-hard even for constraint matrices A whose pathwidth is bounded by 6. By this we mean that the non-parameterized variant of MINFB (where k is part of the input) is NP-hard. To this end, we reduce PARTITION to NEGATIVE DFAS in digraphs whose underlying undirected graph has pathwidth at most 6. Recall that PARTITION is the problem of finding, in a set $\mathcal{A} = \{a_1, \dots, a_n\}$ of positive integers, a subset $\mathcal{A}'$ so that $\sum_{a_i \in \mathcal{A}'} a_i = \sum_{a_i \in \mathcal{A} \setminus \mathcal{A}'} a_i$ or decide that no such set exists. Equivalently, let $A = \sum_{i=1}^{n} a_i$ and reformulate the PARTITION problem as that of finding a subset $\mathcal{A}'$ such that $\mathcal{A}'$ and $\mathcal{A} \setminus \mathcal{A}'$ each sum up to $\frac{A}{2}$. Karp [26] showed that PARTITION is NP-complete.

Starting from an instance $\mathcal{A} = \{a_1, \dots, a_n\} \in \mathbb{N}^n$ of PARTITION, we now construct a NEGATIVE DFAS instance consisting of a digraph G with arc weights $w : A(G) \to \mathbb{Z}$ and some $k \in \mathbb{N}$. Afterwards, we argue why (G, w, k) has a solution if and only if $\mathcal{A}$ has one.

For every number $a_i \in \mathcal{A}$ construct a gadget G_i as follows (see Fig. 1): Let $V^{(i)} = \{s_i^{(j)}, x_i^{(j)}, y_i^{(j)}, z_i^{(j)}, t_i^{(j)} \mid j = 0, 1\}$ be the vertex set of G_i. We have three different kinds of arcs forming the arc set: the first arc set $A_1^{(i)} = \{(x_i^{(j)}, y_i^{(j)}), (y_i^{(j)}, z_i^{(j)}) \mid j = 0, 1\}$ contains the arcs we will consider for deletion later. The arc weight of all arcs $a \in A_1^{(i)}$ is 0.

The second arc set $A_2^{(i)} = \{(z_i^{(j)}, x_i^{(j)}), (y_i^{(j)}, x_i^{(1-j)}), (z_i^{(j)}, y_i^{(1-j)})) \mid j = 0, 1\}$ enforces the deletion of arcs form the first arc set by inducing negative cycles. Also, these arcs are the only arcs between vertices with different superscripts. The weight of each arc $a \in A_2^{(i)}$ is -1.

Finally, the arcs $A_3^{(i)} = \{(s_i^{(j)}, x_i^{(j)}), (s_i^{(j)}, y_i^{(j)}), (y_i^{(j)}, t_i^{(j)}), (z_i^{(j)}, t_i^{(j)}) \mid j = 0, 1\}$ connect the vertices $s_i^{(j)}$ and $t_i^{(j)}$ to the rest of the graph. The arc weights are as follows:

$$w\left((s_i^{(j)}, x_i^{(j)})\right) = 0, \qquad w\left((s_i^{(j)}, y_i^{(j)})\right) = A + 1 + a_i,$$
$$w\left((z_i^{(j)}, t_i^{(j)})\right) = 0, \qquad w\left((y_i^{(j)}, t_i^{(j)})\right) = A + 1 \ .$$

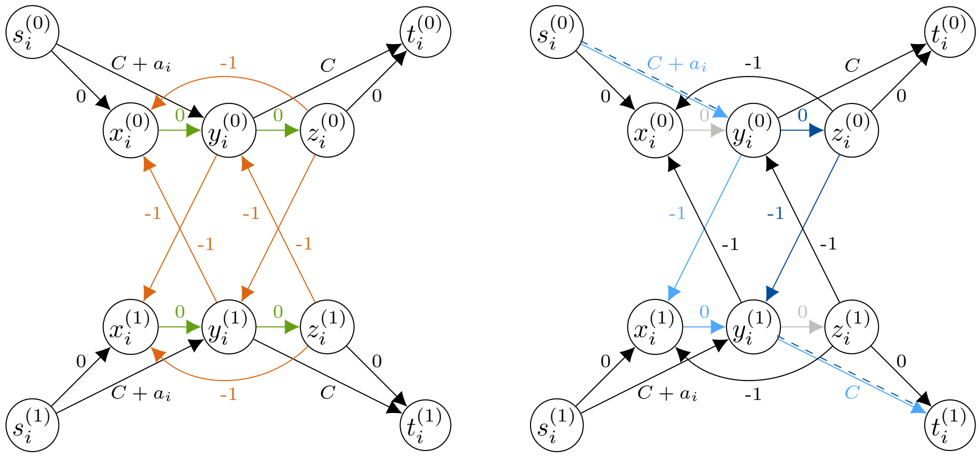

Figure 1 The gadget graph G_i (left) and paths through G_i after removal of a solution $S_i^{(p)}$ (right).

The whole gadget G_i is then defined as $(V^{(i)}, A_1^{(i)} \cup A_2^{(i)} \cup A_3^{(i)})$. The proof of soundness of the reduction and the path decomposition of G of width 6 is omitted from this version of the paper due to space constraints.

Overall, we get that NEGATIVE DFAS is NP-hard in graphs of pathwidth 6. Applying Theorem 12 to this result completes the proof of Theorem 5.

6 Discussion

We considered the fundamental MINFB problem from the perspective of parameterized complexity. Our results include a general parameterized intractability result (W[1]-hardness) even for totally unimodular matrices and parameter solution size, as well as fixed-parameter algorithms for totally unimodular matrices when the additional parameter of number of positive/negative right-hand sides is taken into account. It would be interesting to know whether the run times of our algorithms can be improved to $2^{\mathcal{O}(k)} \cdot n^{\mathcal{O}(1)}$.

We also ruled out the existence of a polynomial compression for combined parameter $k + w_+$, assuming that $\mathsf{coNP} \not\subseteq \mathsf{NP}/poly$. It remains a challenging open problem whether DFAS admits a polynomial compression for parameter k, and whether our perspective from the more general MINFB problem can help with that.

References

1 Edoardo Amaldi and Viggo Kann. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoret. Comput. Sci.*, 209(1-2):237–260, 1998.

2 Sanjeev Arora, László Babai, Jacques Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. System Sci.*, 54(2, part 2):317–331, 1997.

3 Jørgen Bang-Jensen and Gregory Z. Gutin. *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.

4 Piotr Berman and Marek Karpinski. Approximating Minimum Unsatisfiability of Linear Equations. In *Proc. SODA 2002*, pages 514–516, 2002.

5 Hans L. Bodlaender, Fedor V. Fomin, and Saket Saurabh. Open problems from 2010 Workshop on Kernels, 2010. URL: `http://fpt.wdfiles.com/local--files/open-problems/open-problems.pdf`.

6 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Kernelization Lower Bounds by Cross-Composition. *SIAM J. Discrete Math.*, 28(1):277–305, 2014.

7 Hans L. Bodlaender, Stéphan Thomassé, and Anders Yeo. Kernel bounds for disjoint cycles and disjoint paths. *Theoret. Comput. Sci.*, 412(35):4570–4578, 2011.

8 Marthe Bonamy, Łukasz Kowalik, Jesper Nederlof, Michał Pilipczuk, Arkadiusz Socała, and Marcin Wrochna. On Directed Feedback Vertex Set Parameterized by Treewidth. In *Proc. WG 2018*, pages 65–78, 2018.

9 Nilotpal Chakravarti. Some results concerning post-infeasibility analysis. *European J. Oper. Res.*, 73(1):139–143, 1994.

10 Jianer Chen, Yang Liu, Songjian Lu, Barry O'Sullivan, and Igor Razgon. A fixed-parameter algorithm for the directed feedback vertex set problem. *J. ACM*, 55(5):Art. 21, 19, 2008.

11 John W. Chinneck. An effective polynomial-time heuristic for the minimum-cardinality IIS set-covering problem. *Ann. Math. Artif. Intell.*, 17(1):127–144, 1996.

12 John W. Chinneck. Finding a Useful Subset of Constraints for Analysis in an Infeasible Linear Program. *INFORMS J. Comput.*, 9(2):164–174, 1997.

13 John W. Chinneck. *Feasibility and infeasibility in optimization: algorithms and computational methods*, volume 118 of *International Series in Operations Research & Management Science*. Springer, New York, 2008.

14 Rajesh Chitnis, Marek Cygan, Mohammataghi Hajiaghayi, and Dániel Marx. Directed subset feedback vertex set is fixed-parameter tractable. *ACM Trans. Algorithms*, 11(4):Art. 28, 28, 2015.

15 Gianni Codato and Matteo Fischetti. Combinatorial Benders' cuts for mixed-integer linear programming. *Oper. Res.*, 54(4):756–766, 2006.

16 Robert Crowston, Gregory Gutin, Mark Jones, and Anders Yeo. Parameterized complexity of satisfying almost all linear equations over $\mathbb{F}_2$. *Theory Comput. Syst.*, 52(4):719–728, 2013.

17 Marek Cygan, Fedor Fomin, Bart M.P. Jansen, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. Open Problems collected for the 2014 School on Parameterized Complexity in Będlewo, Poland, 2014. URL: `http://fptschool.mimuw.edu.pl/opl.pdf`.

18 Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*. Springer, 2015.

19 Marek Cygan, Łukasz Kowalik, and Marcin Pilipczuk. Open problems from 2013 Workshop on Kernels, 2013. URL: `http://worker2013.mimuw.edu.pl/slides/worker-opl.pdf`.

20 Till Fluschnik, Danny Hermelin, André Nichterlein, and Rolf Niedermeier. Fractals for Kernelization Lower Bounds, With an Application to Length-Bounded Cut Problems. In *Proc. ICALP 2016*, volume 55 of *Leibniz Int. Proc. Informatics*, pages 25:1–25:14, 2016.

21 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, MichaŁPilipczuk, and Marcin Wrochna. Fully Polynomial-Time Parameterized Computations for Graphs and Matrices of Low Treewidth. *ACM Trans. Algorithms*, 14(3):34:1–34:45, 2018.

22 Panos Giannopoulos, Christian Knauer, and Günter Rote. The parameterized complexity of some geometric problems in unbounded dimension. In *Proc. IWPEC 2009*, volume 5917 of *Lecture Notes Comput. Sci.*, pages 198–209. 2009.

23 Petr A. Golovach and Dimitrios M. Thilikos. Paths of bounded length and their cuts: Parameterized complexity and algorithms. *Discrete Optim.*, 8(1):72–86, 2011.

24 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, second edition, 1993.

25 Sylvain Guillemot. FPT algorithms for path-transversal and cycle-transversal problems. *Discrete Optim.*, 8(1):61–71, 2011.

26 Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations (Proc. Sympos., IBM, Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972)*, pages 85–103. Plenum, New York, 1972.

27 Stefan Kratsch and Magnus Wahlström. Two edge modification problems without polynomial kernels. *Discrete Optim.*, 10(3):193–199, 2013.

28 Neele Leithäuser, Sven O. Krumke, and Maximilian Merkert. Approximating infeasible 2VPI-systems. In *Proc. WG 2012*, volume 7551 of *Lecture Notes Comput. Sci.*, pages 225–236. 2012.

29 Daniel Lokshtanov, M. S. Ramanuajn, and Saket Saurabh. When Recursion is Better Than Iteration: A Linear-time Algorithm for Acyclicity with Few Error Vertices. In *Proc. SODA 2018*, pages 1916–1933, 2018.

30 Matthias Mnich and Erik Jan van Leeuwen. Polynomial kernels for deletion to classes of acyclic digraphs. *Discrete Optim.*, 25:48–76, 2017.

31 Marc E. Pfetsch. Branch-and-cut for the maximum feasible subsystem problem. *SIAM J. Optim.*, 19(1):21–38, 2008.

32 G. Ramalingam, J. Song, L. Joskowicz, and R. E. Miller. Solving Systems of Difference Constraints Incrementally. *Algorithmica*, 23(3):261–275, 1999.

33 Jayaram K. Sankaran. A note on resolving infeasibility in linear programs by constraint relaxation. *Oper. Res. Lett.*, 13(1):19–20, 1993.

34 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.

35 K. Subramani and Piotr Wojciechowski. A Combinatorial Certifying Algorithm for Linear Feasibility in UTVPI Constraints. *Algorithmica*, 78(1):166–208, 2017.

36 Meirav Zehavi. Mixing color coding-related techniques. In *Proc. ESA 2015*, volume 9294 of *Lecture Notes Comput. Sci.*, pages 1037–1049. 2015.