Complexity of Minimum Irreducible Infeasible Subsystem Covers for Flow Networks

Imke Joormann and Marc E. Pfetsch

Research Group Optimization, Dept. of Mathematics, TU Darmstadt, Germany,
{joormann,pfetsch}@opt.tu-darmstadt.de

March 2015

Abstract

For an infeasible network flow system with supplies and demands, we consider the problem of finding a minimum irreducible infeasible subsystem cover, i.e., a smallest set of constraints that must be dropped to obtain a feasible system. The special cases of covers which only contain flow balance constraints (node cover) or only flow bounds (arc cover) are investigated as well. We show strong $\mathcal{NP}$-hardness of all three variants. We show that finding minimum arc covers for assignment problems is still hard and as hard to approximate as the set covering problem. However, the minimum arc cover problem is polynomially solvable for networks on cactus graphs. This leads to the development of two different fixed parameter algorithms with respect to the number of elementary cycles connected at arcs and the treewidth, respectively. The latter can be adapted for node covers and the general case.

1 Introduction

Analyzing infeasibility of linear programs (LPs) is an important topic, since it can help to find disrupted data or locate modeling errors. One tool for this purpose are small sets of constraints whose removal renders the LP feasible. This requires to remove at least one constraint from every *irreducible infeasible subsystem* (IIS), i.e., an infeasible subsystem such that each proper subsystem is feasible. Thus, we are interested in *minimum IIS covers* (minIISCs)

In this paper, we are concerned with the special case of network flow systems

$$x(\delta^+(v)) - x(\delta^-(v)) = b(v) \qquad \forall\, v \in V, \tag{1a}$$

$$\ell \leq x \leq u, \tag{1b}$$

for a simple, directed graph $G = (V, A)$ with upper bounds $u \in \mathbb{R}^A$, lower bounds $\ell \in \mathbb{R}^A$, and a supply vector $b \in \mathbb{R}^V$. For $S \subseteq V$ and $\bar{S} := V \setminus S$, we use the following standard notation: $\delta^+(S) := \{(v, w) \in A \,|\, v \in S,\ w \in \bar{S}\}$, $\delta^-(S) := \{(v, w) \in A \,|\, v \in \bar{S},\ w \in S\}$, and $\delta(S) := \delta^+(S) \cup \delta^-(S)$. Moreover, for a finite index set I, a vector $y \in \mathbb{R}^I$, and $I' \subseteq I$, we write $y(I') := \sum_{i \in I'} y_i$ and use $y(i) := y(\{i\}) = y_i$. The number of nodes and arcs of G are $n := |V|$ and $m := |A|$, respectively.

In order to avoid trivial infeasibilities, we assume that $\ell \leq u$ and $u \geq 0$ throughout the article.

The analysis of IISs and IIS covers for LPs has been treated extensively in the literature (see Sect. 1.1 below for a survey). However, to the best of our knowledge, the special case of IIS covers for flow networks has not been treated so far. In this article we investigate three kinds of such IIS covers: An *IIS node cover* (INC) covers all IISs of the network problem by node constraints alone, i.e., flow conservation equations (1a). Similarly, an *IIS arc cover* (IAC) contains only arc constraints, i.e., lower or upper bounds (1b). The combination of both then yields general IIS covers. The corresponding minimization problems are then called MINC, MIAC, and MIC, respectively.

The goal of this article is to extend the knowledge on finding minimum IIS covers by considering the base case of flow networks. We investigate the structural properties of the three types of IIS covers and, in particular, investigate their computational complexity and (non-)approximability properties.

To this end, we first establish $\mathcal{NP}$-hardness of the three problems MIC, MIAC, and MINC in Sect. 2. In Sect. 3, we study characteristics of IIS covers and MIAC, in particular. For instance, we show that MIAC is approximable within $c\,(n-1)$ for every constant $c > 0$. We demonstrate that MIC can be formulated as a MIAC instance; hence, MIC can also be approximated within $c\,n$. Furthermore, we introduce the concept of redundancy with respect to covering of IISs. This leads to two preprocessing rules that allow to simplify the network. We also examine the relation between MIAC and MINC. We then show in Sect. 4 that MIAC is $\mathcal{NP}$-hard to approximate within $c \ln n$, even on assignment problems. Moreover, we develop polynomial time algorithms on trees, cycles, and more generally on cactus graphs. Furthermore, we give two fixed parameter algorithms with respect to the number of elementary cycles connected at arcs (MIAC) and with respect to the treewidth (all three variants).

Thus, on the one hand it turns out that minimum IIS covers are already hard to compute for flow networks. In fact, MIAC is as hard to approximate as the minimum IIS cover problem for general LPs. On the other hand, one can use the underlying graph structure to derive tractable special cases. Consequently, this article complements the existing knowledge on computing IIS covers for infeasible LPs to a certain extent, which we review in the next subsection.

1.1 Literature Overview

The analysis of general infeasible linear systems has been extensively investigated. For a broad overview, we refer to the book by Chinneck [12]. Moreover, we also mention the article of Greenberg [21], which gives a unified presentation of infeasibility and redundancy. In [19] and [20], he also studied infeasible networks and gave heuristics to "localize" the cause of infeasibility. In the following, we concentrate on minIISC results.

For general LPs, minIISC is equivalent to its complementary problem, the *maximum feasible subsystem* (maxFS) problem. Chakravarti [9] showed that maxFS is strongly $\mathcal{NP}$-hard (thus, minIISC is also strongly $\mathcal{NP}$-hard). While minIISC and maxFS are equivalent with respect to optimality, this does not hold for approximability, see Amaldi and Kann [4, 5].

MinIISC can be formulated as a hitting set (or as a set covering) problem, where the sets are all IISs I of a linear system with p constraints. We can hence solve minIISC optimally via the following integer program (IP):

$$\begin{aligned} \min\ & \mathbb{1}^\top y \\ \text{s.t.}\ & \sum_{i\in\mathcal{I}} y_i \geq 1 \quad \forall \mathcal{I} \in I \\ & y \in \{0,1\}^p, \end{aligned} \tag{2}$$

where we identify the constraints by their indices $1, \dots, p$, and $\mathbb{1}$ denotes the all-ones vector of appropriate dimension.

Parker and Ryan [29] present an iterative process to solve (2), in which IISs are generated dynamically. Here, (2) is solved for a partial set of IISs. An integral solution can be used to efficiently find uncovered IISs by using a result of Gleeson and Ryan [16]: The index sets of IISs of an infeasible linear system are exactly the supports of the vertices of the associated alternative polyhedron. If an uncovered IIS is found, it is added to the set, and the process is iterated. A branch-and-cut approach to solve (2) is given in [30], which generates IISs on the fly. Chinneck presented heuristics to solve minIISC [10] and maxFS [11].

Sankaran [31] proved $\mathcal{NP}$-hardness of minIISC for $Dy \leq d$ with a transposed node-arc-incidence matrix D. Hence, his result does not carry over to our case. He also presented an easy special case: If the concatenated matrix $[D\,d]$ is totally unimodular, minIISC can be solved in polynomial time for the general linear system $Dy \leq d$.

Furthermore, Amaldi and Kann [5] use the name "unsatisfied linear relations" (MIN ULR) for MinIISC. They discussed the (non-)approximability properties of $Dy\mathscr{R}d$ for $\mathscr{R} \in \{\neq, =, \geq, >\}$ and arbitrary D and d. These results can be extended to the constrained C MIN ULR, where some constraints are mandatory and have to be satisfied. For $\mathscr{R} \in \{=, \geq, >\}$, they showed that MIN ULR can be approximated within $m+1$, where m is the number of variables, by the following observation: In the approach by Parker and Ryan mentioned above, including every constraint of a newly found IIS in the IIS cover would increase the cover by at most $m+1$ instead of 1, since each IIS can have at most $m+1$ constraints, see Motzkin [28]. Amaldi and Kann [5] also claimed (non-)approximability results for different versions of MIN ULR on node-arc-incidence matrices using Sankaran's results. However, these results are incorrect, since Sankaran [31] used *arc-node*-matrices.

In the following, we write (1) as $Mx = b$, $\ell \leq x \leq u$, where the matrix M is the totally unimodular node-arc-incidence matrix of G. In this context, McCormick [26] considered the following slack-formulation for an infeasible network flow:

$$\begin{aligned} \min\ & \|s\| \\ & Mx = b + s_1 \\ & \ell - s_2 \leq x \leq u + s_3 \\ & s_2, s_3 \geq 0, \end{aligned} \tag{3}$$

where s is the concatenation of s_1, s_2, and s_3. He developed efficient network flow methods for the cases in which $\|s\|$ is the ℓ_1-, ℓ_2-, or ℓ_∞-norm, respectively. He also distinguished between the cases where only node or arc constraints might be violated, by setting the respective part of s to zero. The problems MIC, MIAC, and MINC can be formulated using (3) with an ℓ_0-"norm", i.e., $\|s\|_0 := \{i \mid s_i \neq 0\}$.

Finally note that MIAC shares a certain structure with the min edge cost flow (MECF) problem, see [15, ND32]. MECF is a network flow problem with objective function

$$\sum_{a \in A\colon x_a > 0} c_a \tag{4}$$

for costs $c \in \mathbb{R}^A$, flow $x \in \mathbb{R}^A$, and lower bounds $\ell = 0$. For unitary costs, this can also be formulated using the ℓ_0-"norm" of x. We will come back to this relation in Remark 30.

1.2 Preparational Observations

A complete characterization of infeasibility in network flows is of course well known:

Theorem 1 (Gale [14] and Hoffman [22]). *The network flow system (1) is infeasible if and only if there exists $S \subseteq V$ such that*

$$b(S) > u(\delta^+(S)) - \ell(\delta^-(S)). \tag{5}$$

The characterization could as well be given with the inequality

$$-b(S) > u(\delta^-(S)) - \ell(\delta^+(S)), \tag{6}$$

since whenever S fulfills one inequality, $\bar{S}$ satisfies the other. We will call (6) the *demand form*, since it belongs to a subset with an unsatisfied demand, and (5) the *supply form*. Furthermore, we will refer to both inequalities as *GH-inequalities* and to the corresponding sets $S \subseteq V$ as *GH-sets*.

In [23] (see also Theorem 2 below), we used this characterization to show that the IISs of network flow problems correspond to exactly those violated GH-inequalities for which the induced subgraph $G[S]$ is weakly connected, i.e., it is connected in the underlying undirected graph. For a formal description, and for the following sections, we will need some more notation: Let σ_v refer to the flow conservation constraint for a node $v \in V$, μ_a to the constraint $x_a \leq u_a$, and λ_a to the constraint $x_a \geq \ell_a$ for $a \in A$. For a GH-set S, we denote the *GH-subsystem* by

$$\mathcal{I}(S) \coloneqq \{\sigma_v \,|\, v \in S\} \cup \{\mu_a \,|\, a \in \delta^+(S)\} \cup \{\lambda_a \,|\, a \in \delta^-(S)\}$$

for the supply form, and

$$\mathcal{I}(S) \coloneqq \{\sigma_v \,|\, v \in S\} \cup \{\mu_a \,|\, a \in \delta^-(S)\} \cup \{\lambda_a \,|\, a \in \delta^+(S)\}$$

for the demand form. Note that at most one case for a given S can apply. Whenever necessary, we will specify which one is meant.

Theorem 2 ([23]). *A subsystem I of the network flow system (1) is an IIS if and only if there exists $S \subseteq V$ for which $I = \mathcal{I}(S)$ (in either demand or supply form) and the induced subgraph $G[S]$ is weakly connected.*

This characterization of IISs will be fundamental at different points throughout the present paper.

Network flow problems have exactly two types of linear constraints: the flow conservation constraints (1a) (*node constraints*) and the flow bounds (1b) (*arc constraints*). Especially in an application, where we want to offer guidance as to how infeasibility might be repaired, it can be useful to differentiate between node and arc constraints. While the flow bounds represent capacity constraints, e.g., corresponding to a pipeline diameter, flow conservation determines the flow amount that must be transported. It might be possible that the infrastructure of the network is fixed, while the requested flow amount can be changed, and vice versa. Thus, it can be beneficial to only suggest either node or arc constraints for removal or repair. Hence, we also investigate the settings where only one kind of constraint can be violated, while the other must be fulfilled; note that this fits into the framework of C MIN ULR (see [5]).

An instance for MIC, MIAC, and MINC will be denoted by $(G = (V, A); b, u, \ell)$ in the following, and we always regard directed graphs. For an arc constraint, the covering can

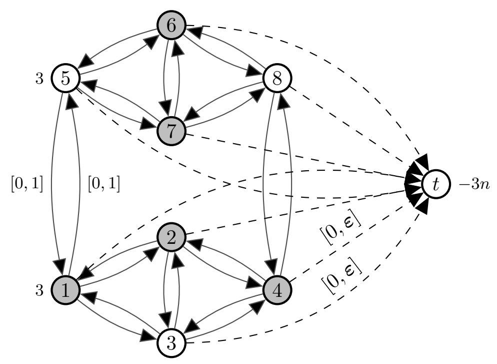


Figure 1: Sketch of the construction for the reduction; $\ell \equiv 0$, $u = 1$ for solid, $u = \varepsilon$ for dashed arcs; minimum vertex cover marked gray.

be expressed by setting $u \coloneqq \infty$ and $\ell \coloneqq -\infty$, respectively, to obtain the feasible instance corresponding to the deletion of the IAC. If there are negative lower bounds, an arc can also be used in the reverse direction. Hence, by including the lower bound in a cover, we can "make" an arc undirected (except for MINC). Because of this, the underlying undirected graph of G will play a major role, and statements about the structure of G will always refer to the underlying undirected graph, i.e., connected refers to weakly connected, and tree and cycle describe the respective undirected version.

2 Complexity of Minimum IIS Cover in Flow Networks

We begin by settling the hardness of MIC.

Theorem 3. *Given a network flow problem (1) and a positive integer k, it is $\mathcal{NP}$-complete in the strong sense to decide whether an IIS cover of size at most k exists.*

Proof. Obviously, the problem is in $\mathcal{NP}$: Given a potential cover, we simply remove the indicated constraints and check the remaining system for feasibility.

We reduce the strongly $\mathcal{NP}$-complete minimum vertex cover problem on cubic graphs (see [GT1] in [15]): Given an undirected graph $G' = (V', E')$, $|V'| = n$, which is cubic, i.e., $|\delta(v)| = 3$ for all $v \in V'$, does there exist a vertex cover $V^* \subseteq V'$, i.e., at least one of the endpoints of each edge is contained in V^*, such that $|V^*| \leq k$?

We construct a network problem on a directed graph $G = (V, A)$, which has an IIS cover of size k iff $|V^*| = k$. Starting with G', each node gets a supply of 3. The edges E' are replaced by pairs of oppositely directed arcs, each with an upper flow bound of 1. Additionally, we introduce a sink node t with a demand of $3n$ and arcs from every node to t with an upper bound of a small ε (e.g., $\varepsilon = 1/n$), see also Fig. 1. In summary, $V \coloneqq V' \cup \{t\}$, $A \coloneqq \{(v, w), (w, v) \mid \{v, w\} \in E'\} \cup \{(v, t) \mid v \in V'\}$,

$$b_v \coloneqq \begin{cases} 3, & v \in V', \\ -3n, & v = t, \end{cases} \qquad u_a \coloneqq \begin{cases} \varepsilon, & a = (v, t), \\ 1, & \text{otherwise}, \end{cases} \qquad \text{and} \qquad \ell_a = 0 \quad \forall a \in A.$$

Let us first examine the IISs $\mathcal{I}(S)$ of this network problem. Note that in the original undirected graph, it holds that $|\delta(S)| \leq 3|S| - 2(|S| - 1) = |S| + 2$ for every connected $S \subseteq V'$ since the graph is cubic. Now recall that by Theorem 2, IISs correspond to connected GH-sets. We start with the supply-form, so $S \subseteq V'$. Obviously, for $S = \{v\}$, $v \in V'$, $3 = b(v) \leq 3 + \varepsilon = u(\delta^+(v))$, so $\mathcal{I}(S)$ is not infeasible in this case. For $|S| \geq 2$

and S connected, $b(S) = 3|S| > (1+\varepsilon)|S| + 2 \geq u(\delta^+(S))$; therefore, every combination of at least two connected nodes yields an IIS. Every disconnected subset is of no concern since it cannot be an IIS. For the demand-form, $t \in S$ must hold. Furthermore, $S = \{t\}$ obviously yields an IIS. Note that there are more IISs, but these are all that we need.

Let V^* be a vertex cover in G' with $|V^*| = k$. Then $\mathcal{C} := \{\mu_a \,|\, a = (v,t), v \in V^*\}$ is an IIS cover since for every connected $S \subset V$ with $t \notin S$, there exists $a \in \delta^+(S)$ such that $\mu_a \in \mathcal{C}$ and for every $S \ni t$, there exists $a \in \delta^-(S)$ such that $\mu_a \in \mathcal{C}$. Furthermore, $|\mathcal{C}| = k$.

Now let $\mathcal{C}$ be an IIS cover with $|\mathcal{C}| = k$. First, suppose that $\mathcal{C}$ is an IIS node cover. We know that every pair of two connected nodes yields an IIS and that $\mathcal{C}$ must contain at least one of them – which is exactly the requirement of a vertex cover. Next, let $\mathcal{C}$ be an IIS arc cover, and let $A_{\mathcal{C}} := \{a \,|\, \mu_a \in \mathcal{C}\} \cup \{(w,v) \,|\, \lambda_a \in C,\ a = (v,w)\}$. Let $V(A_{\mathcal{C}}) := \{v \,|\, (v,w) \in A_{\mathcal{C}}\}$, then $|V(A_{\mathcal{C}})| \leq |A_{\mathcal{C}}|$. Suppose $V(A_{\mathcal{C}})$ is not a vertex cover: Then there exists $(v_1,v_2) \in A$ such that $(v_1,w) \notin A_{\mathcal{C}}$ and $(v_2,w) \notin A_{\mathcal{C}}$ for all $w \in V$. Hence, the IIS $\mathcal{I}(\{v_1,v_2\})$ is not covered and $\mathcal{C}$ is no IIS cover. Finally, let $\mathcal{C}$ be a mixture of node and arc constraints with $V_{\mathcal{C}} := \{v \,|\, \sigma_v \in \mathcal{C}\}$ and $A_{\mathcal{C}}$ as above. With the same reasoning as above, $V_{\mathcal{C}} \cup V(A_{\mathcal{C}})$ must be a vertex cover.

Note that the size of the constructed instance and all occurring numbers are polynomially bounded in $|A'| + |V'|$. □

Corollary 4. *Solving maxFS for network flows is strongly $\mathcal{NP}$-hard.*

Although we will achieve stronger results in the course of the paper, it follows immediately from the above proof that MIAC is also hard, since the constructed IIS cover only contains arc constraints.

Corollary 5. *Given a network flow problem* (1) *and a positive integer k, it is $\mathcal{NP}$-complete in the strong sense to decide whether an IIS arc cover of size at most k exists.*

Furthermore, with a slight modification, the proof also shows hardness of MINC.

Proposition 6. *Given a network flow problem* (1) *and a positive integer k, it is $\mathcal{NP}$-complete in the strong sense to decide whether an IIS node cover of size at most k exists.*

Proof. We use the same construction as for Theorem 3. There exists a vertex cover V^* of size k for G' iff the network problem on G has an IIS node cover $\mathcal{C}$ of size $k+1$: $\mathcal{C} = \{\sigma_v \,|\, v \in V^* \cup \{t\}\}$ is obviously an INC; because of the IISs of the network problem (see proof of Theorem 3) there cannot be a smaller one. □

Solving minimum vertex cover on cubic graphs is APX-complete (see [3]), i.e., the problem can be approximated in polynomial time within some constant factor, but not within *every* constant. Since the above proofs constitute L-reductions for MIC, MIAC, and MINC, all three problems are APX-hard. Unfortunately, there is no constant approximation algorithm for the IIS cover problems known, and we are about to show that for MIAC, there cannot exist one unless $\mathcal{P} = \mathcal{NP}$.

Proposition 7. *MIAC is not approximable within $c \ln n$ for any constant $0 < c < 1$ (unless $\mathcal{P} = \mathcal{NP}$).*

Proof. We provide an L-reduction from the strongly $\mathcal{NP}$-complete set cover problem: Given a set R and a collection D of subsets $d \subseteq R$, we are looking for a cover $D' \subseteq D$ of R. Moshkovitz [27] recently showed that the set cover problem is not approximable within $c \ln |D|$ for any constant $0 < c < 1$ if $\mathcal{P} \neq \mathcal{NP}$. We construct an infeasible flow

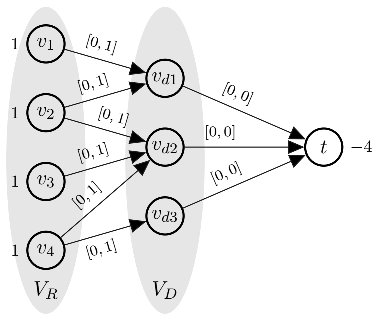


Figure 2: Sketch of the construction for the reduction from set cover to IAC, with ground set $R = \{1, 2, 3, 4\}$ and subset collection $D = \{\{1, 2\}, \{2, 3, 4\}, \{4\}\}$.

problem $(G; b, u, \ell)$ on a graph $G = (V, A)$ that has an IIS arc cover of size k iff there is a set cover D' with $|D'| \leq k$.

First, we use the known formulation for set cover problems as a bipartite, directed graph, where we have a node v_r in a set V_R for every element $r \in R$ and a node v_d in a set V_D for every subset $d \in D$. If $r \in d$, there is an arc from $v_r \in V_R$ to $v_d \in V_D$. We add a sink node t and connect every $v_d \in V_D$ to t. Hence,

$$\begin{aligned}
&V_R \coloneqq \{v_r \,|\, r \in R\}, \quad V_D \coloneqq \{v_d \,|\, d \in D\}, \quad V \coloneqq V_R \cup V_D \cup \{t\},\\
&A_1 \coloneqq \{(v_r, v_d) \,|\, v_r \in V_R,\, v_d \in V_D,\, r \in d\},\\
&A_2 \coloneqq \{(v_d, t) \,|\, v_d \in V_D\}, \quad A \coloneqq A_1 \cup A_2,\\
&b_v \coloneqq \begin{cases} 1, & v \in V_R,\\ 0, & v \in V_D,\\ -|R|, & v = t, \end{cases} \qquad u_a \coloneqq \begin{cases} 1, & a \in A_1,\\ 0, & a \in A_2, \end{cases} \qquad \ell \coloneqq 0;
\end{aligned}$$

see also Fig. 2.

Let D^* be a set cover with $|D^*| = k$, and let $V_{D^*} \coloneqq \{v_d \in V_D \,|\, d \in D^*\}$. It holds that $|D^*| = |V_{D^*}| = |\delta^+(V_{D^*})| = k$. Furthermore, the flow problem with $u_a \coloneqq \infty$ for $a \in \delta^+(V_{D^*})$ is feasible: Since D^* is a set cover, all supply from V_R can be transported to V_{D^*}, and via the infinite-capacitated arcs to t. Hence, $\{\mu_a \,|\, a \in \delta^+(V_{D^*})\}$ is an IAC with the claimed size.

Now suppose there is an IAC $\mathcal{C}$ with cardinality $\tilde{k} < k$, but no set cover with cardinality smaller than k. Let x be the corresponding feasible flow for $\mathcal{F} \setminus \mathcal{C}$ and let $\hat{V}_D \coloneqq \{v_d \in V_D \,|\, \mu_a \in \mathcal{C},\ a \in \delta^+(v_d)\}$. Obviously $\hat{k} \coloneqq |\hat{V}_D| \leq \tilde{k}$ and there exists an undirected path P with respect to $\{a \in A_1 \,|\, x_a \neq 0\}$ from every node $v \in V_R$ to $\hat{V}_D$. Since $|\hat{V}_D| < k$, $\hat{V}_D$ is not a set cover, and there is at least one $v \in V_R$ with $\delta(v) \cap \delta(\hat{V}_D) = \emptyset$. Moreover, $(V_R \cup V_D, A_1)$ is bipartite, so $|P| \geq 3$ and the path must use an arc $a \in A_1$ in backward direction. To obtain a feasible flow, we must therefore have $\lambda_a \in \mathcal{C}$. Now observe that $V' \coloneqq \{v \in V_D \,|\, \lambda_{(w,v)} \in \mathcal{C}\} \cup \hat{V}_D$ yields a set cover, i.e., $k \leq |V'|$. Since $|\mathcal{C}| \geq |V'| \geq k$, this yields a contradiction. □

In Sect. 4, we will further strengthen this proposition, but it is beneficial for the presentation to first capture the proof in its basic version.

Remark 8. The given construction is inspired by a non-approximability result of Di Gaspero et al. [13], who reduced set cover to the infinite capacity min edge cost flow problem (ICF) on directed acyclic graphs with unitary edge costs, although for our problem, a simpler construction suffices.

Note that the stronger non-approximability result for set cover, proven in the meantime by Moshkovitz [27], also carries over to the ICF and the min shift design problem in the paper of Di Gaspero et al. [13].

3 Characteristics of the IIS covers

In the previous section, we have seen that the computation of minimum IIS covers in each version is hard. To obtain positive (approximation) results, we will now explore characteristics of the IIS covers, which will help in understanding the underlying mechanisms. The results are structured by type of the IIS covers, although the first basic observation holds not only for flow systems, but even for mixed-integer linear programs (MILPs). The statement is nearly trivial, however, we could not find a proof in the literature.

Lemma 9. *A minimal IIS cover for an arbitrary linear system cannot contain lower and upper bounds of the same variable simultaneously if the bounds are consistent.*

Proof. Let $\mathcal{C}$ be an IIS cover for an infeasible linear constraint system $\mathcal{F}$ such that $\{y_i \geq \ell_i\}, \{y_i \leq u_i\} \in \mathcal{C}$ for some variable y_i. By definition, there exists a solution $\hat{y}$ for $\mathcal{F} \setminus \mathcal{C}$. The value of $\hat{y}_i$ cannot violate the consistent upper and lower bound at the same time, so let w.l.o.g. $\hat{y}_i \leq u_i$. Then $\hat{y}$ is still feasible for $\mathcal{F} \setminus (\mathcal{C} \setminus \{y_i \leq u_i\})$, and $\mathcal{C}$ is not minimal. □

3.1 IIS Arc Covers

We start with an existence statement for IACs, which will dictate our preconditions for the following sections.

Lemma 10. *There exists an IIS arc cover iff every connected component of G is balanced.*

Proof. Let G decompose into connected components $(V_1, A_1), \dots, (V_k, A_k)$, each of them balanced, i.e., $b(V_i) = 0$ for all $i = 1, \dots, k$. First, suppose $k = 1$, i.e., G is connected. Hence, there is a spanning tree $T \subseteq A$ for G. The set $\{\mu_a, \lambda_a \,|\, a \in T\}$ covers every possible IIS: For any $S \subset V$, there exists $a \in \delta(S) \cap T$. Suppose the GH-inequality for S in supply-form is violated. Then μ_a is covered for $a \in \delta^+(S)$ and λ_a is covered for $a \in \delta^-(S)$ (and vice versa for the demand-form). If $S = V$, the GH-inequality for S is not violated, since $b(V) = 0$.

Now let $k > 1$. Since there are no IISs $\mathcal{I}(S)$ with $v \in S \cap V_i$, $w \in S \cap V_j$, $i \neq j$, by Theorem 2, the above argumentation holds componentwise if $b(V_i) = 0$ for all $i \in [k]$, where $[k] \coloneqq \{1, \dots k\}$.

For the opposite direction, suppose there exists $i \in [k]$ with $b(V_i) \neq 0$. Since $\delta(V_i) = \emptyset$, $\mathcal{I}(V_i)$ is an IIS and cannot be covered by arc constraints. □

For the remainder of the paper, we will therefore assume that every connected component of the given graph is balanced, such that an IAC always exists. Furthermore, we can actually assume w.l.o.g. that the graph is connected, since under these preconditions every connected component can be handled independently.

The following lemma shows that enumerating minimal IACs is not a promising strategy in general.

Lemma 11. *There can be exponentially many minimal IACs.*

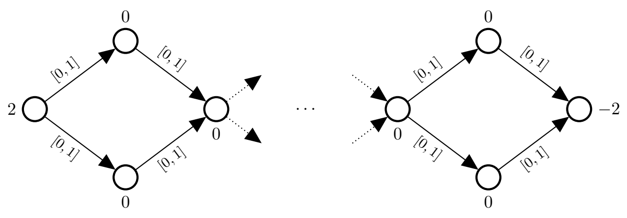


Figure 3: Network problem (node labels b, arc labels $[\ell, u]$), with an exponential number of minimal IACs.

Proof. Consider the example in Fig. 3. For every IAC $\mathcal{C}$, both upper bounds in the top half of each rhombus or both bounds in the bottom part have to be in $\mathcal{C}$. This proceeds throughout the graph, and at each intersection, we can freely choose if we take the lower or upper path. Hence, there are $2^{(n-1)/3}$ minimal IACs.

The described IACs are actually minimum in this case, therefore we can even have exponentially many MIACs. $\square$

Since we can assume the graph to be connected, we can pursue the idea of regarding spanning trees and obtain bounds on the size of a MIAC.

Lemma 12. *Let $\mathcal{C}$ be a minimum IAC. Then $1 \leq |\mathcal{C}| \leq n-1$ and these bounds are tight.*

Proof. Let T be a spanning tree. We have shown in the proof of Lemma 10 that $\mathcal{C} := \{\mu_a, \lambda_a \mid a \in T\}$ is an IAC, regardless of the data. Hence, there exists a solution x for $\mathcal{F} \setminus \mathcal{C}$. Let $\mathcal{C}' := \mathcal{C} \setminus \{\mu_a \mid x_a \leq u_a,\ a \in A\} \setminus \{\lambda_a \mid x_a \geq \ell_a,\ a \in A\}$. Due to Lemma 9, it holds that $|C'| \leq n-1$.

The instance $V = [n]$, $A = \{(i, i+1) \mid i \in [n-1]\}$, $b_1 = 2$, $b_n = -2$, $b_i = 0$ for $i = 2, \ldots, n-1$ and $u \equiv \mathbb{1}$, $\ell \equiv 0$ attains the upper bound. Examples for a tight lower bound are easily constructed. $\square$

In the following, we will use the notation $A(\mathcal{C}) := \{a \in A \mid \mu_a \in \mathcal{C}\} \cup \{a \in A \mid \lambda_a \in \mathcal{C}\}$ to denote the affected network elements of an arc cover $\mathcal{C}$, and correspondingly $V(\mathcal{C}) := \{v \in V \mid \sigma_v \in \mathcal{C}\}$ for a node cover $\mathcal{C}$.

The tree structure also has a special meaning for minimal IACs:

Lemma 13. *For every minimal IAC $\mathcal{C}$, $A(\mathcal{C})$ is a forest. Hence, Lemma 12 also holds for minimal IACs.*

Proof. Suppose $\mathcal{C}$ is a minimal IAC of the constraint system $\mathcal{F}$ such that $A(\mathcal{C})$ contains an (undirected) cycle $Z \subseteq A$, and let x be a solution for $\mathcal{F} \setminus \mathcal{C}$. Now we augment flow along the cycle with the amount of the minimum bound violation: For all $a \in Z$, if $\lambda_a \in \mathcal{C}$, then $c_a := \ell_a - x_a$, and if $\mu_a \in \mathcal{C}$, then $c_a := x_a - u_a$ (by Lemma 9, $\mathcal{C}$ would not be minimal if both of them were in $\mathcal{C}$). Note that $c_a > 0$, otherwise $\mathcal{C}$ is not minimal. Let $\hat{a} \in \arg\min\{|c_a| \mid a \in Z\}$, and define the orientation of the cycle in the direction of $\hat{a}$. Let $x'_a = x_a$ for all $a \notin Z$, set $x'_a = x_a + c_{\hat{a}}$ for all arcs in forward direction and $x'_a = x_a - c_{\hat{a}}$ for backward arcs. Then x' is feasible for $\mathcal{F} \setminus (\mathcal{C} \setminus \{\lambda_{\hat{a}}, \mu_{\hat{a}}\})$. Therefore, $\mathcal{C}$ is not minimal. $\square$

The idea in Lemma 12 actually gives us an approximation algorithm for MIAC in n. This factor is non-trivial since we have $2m$ coverable constraints and m variables, which can be much larger than n.

Algorithm 1 MIAC: $(n-1)$-approximation algorithm.

```
Input: Infeasible flow network problem $(G; b, u, \ell)$ for a connected graph $G = (V, A)$ and
    balanced supply vector.
Output: An IIS arc cover $\mathcal{C}$.
1: Compute an (undirected) spanning tree $T$ for $G$
2: $\mathcal{C} \leftarrow \{\mu_a, \lambda_a \,|\, a \in T\}$
3: Set $\tilde{u}_a = \infty$, $\tilde{\ell}_a = -\infty$ for all $a \in T$
4: Compute a feasible solution $\tilde{x}$ for $(G; b, \tilde{u}, \tilde{\ell})$
5: $\mathcal{C} \leftarrow \mathcal{C} \setminus \{\mu_a \,|\, \tilde{x}_a \leq u_a\} \setminus \{\lambda_a \,|\, \tilde{x}_a \geq \ell_a\}$
6: return $\mathcal{C}$
```

Theorem 14. *MIAC is approximable within a factor of $c\,(n-1)$ for every constant $c > 0$ in polynomial time.*

Proof. Consider Algorithm 1. The proof of Lemma 12 shows that Algorithm 1 always returns an IAC $\mathcal{C}$ with $|\mathcal{C}| \leq n-1$. The running time of the algorithm is dominated by the computation of a feasible solution in Step 4, which can be done with one max-flow computation, and is therefore polynomial.

Regarding the additional factor c, note that we can check in polynomial time whether an IAC of size k exists, for every *constant* $k \in [m]$. Before invoking Algorithm 1, we can therefore rule out a minimum IAC of size k for every $1 \leq k \leq \min\{\lceil 1/c \rceil, n-2\}$ (or stop if we have found one). Hence, either a MIAC has size at most $\lceil 1/c \rceil$ and the approximation algorithm provides an optimal solution, or the size of a MIAC $\mathcal{C}^*$ is larger than $\lceil 1/c \rceil$ and the approximated solution has size at most $n-1$. Thus, $c\,(n-1)|\mathcal{C}^*| \geq n-1 \geq |\mathcal{C}|$. ☐

The following example has an optimal solution $\mathcal{C}_{\text{opt}}$ with $|\mathcal{C}_{\text{opt}}| = 1$, while for the approximated solution $\mathcal{C}$, $|\mathcal{C}| = n-1$: $V = [n]$, $A = \{(i, i+1) \,|\, i \in [n-1]\} \cup \{(1, n)\}$, $b_1 = 2$, $b_n = -2$, $b_i = 0$, $i = 2, \ldots, n-1$, and $u \equiv \mathbb{1}$, $\ell \equiv 0$. If the tree chosen in Step 1 is $\{(i, i+1) \,|\, i \in [n-1]\}$, the approximation factor is $n-1$, which shows tightness of the bound for Algorithm 1.

Remark 15. To obtain at least a *minimal* IIS arc cover, we can extend Algorithm 1 after Step 5 and check greedily if a bound can be removed from the cover.

Another idea to obtain approximation results might be the primal-dual approximation approach (see [1]), which was successfully used by Goemans and Williamson [18] on a hitting set formulation for different network design problems. For a graph $G = (V, E)$ and a suitable function $f : 2^V \to \mathbb{R}$ the generalized set covering formulation

$$\begin{aligned} &\min\ c^\top y \\ &\text{s.t.} \sum_{e \in \delta(S)} y_e \geq f(S) \quad \forall\, S \subseteq V \\ &\qquad y \in \{0,1\}^E, \end{aligned} \tag{7}$$

can describe, e.g., Steiner trees or minimum weight perfect matchings. Depending on the properties of f, rather good approximation results can be achieved. For instance, every proper function leads to a 2-approximation, as do downwards monotone functions. Goemans et al. [17] extended the algorithm for weakly supermodular functions. Comparing (7) with (2) shows that MIAC (and MINC and MIC, after slight modifications) can be modeled in this way, with $f(S) = 1$ if $\mathcal{I}(S)$ is an IIS and $f(S) = 0$ otherwise.

Unfortunately, this function is neither proper, nor downwards monotone or weakly supermodular, since $\mathcal{I}(S_1)$ and $\mathcal{I}(S_2)$ might be IISs, while $\mathcal{I}(S_1 \cup S_2)$ is not, and vice versa.

As Theorem 1 states, the feasibility of a flow network can be fully described by the system $b(S) \leq u(\delta^+(S)) - \ell(\delta^-(S))$ for all $S \subset V$. However, not all of these inequalities are actually necessary.

Theorem 16 (Wallace and Wets [33])**.** *Let $b(V) = 0$ and G be connected. For all $\emptyset \neq S \subset V$, the inequality $b(S) \leq u(\delta^+(S)) - \ell(\delta^-(S))$ is redundant if and only if at least one of S and $\bar{S}$ is not connected.*

Note that Wallace and Wets gave the theorem for $\ell \equiv 0$, but the proof for a network with lower bounds $\ell \neq 0$ runs completely analogously. From Theorem 16, we can conclude that certain IISs are irrelevant for the covering problem.

Definition 17. *An IIS $\mathcal{I}(S)$ is called* redundant *(with respect to covering) if an arbitrary cover for all other IISs also covers $\mathcal{I}(S)$.*

For IISs $\mathcal{I}(S)$, we know that S must always be connected (see Theorem 2), but $\bar{S}$ does not need to be. The additional connectivity requirement for $\bar{S}$ carries over to redundancy of IISs in arc covers.

Lemma 18. *Let $b(V) = 0$ and G be connected. For IACs, an IIS $\mathcal{I}(S)$ is redundant if $\bar{S}$ is* not *connected.*

Proof. Suppose we have an IAC $\mathcal{C}$ covering all IISs $\mathcal{I}(S)$ for which $\bar{S}$ is connected. An IIS arc cover is equivalent to setting the corresponding bounds to $\pm\infty$, so the GH-inequalities become satisfied. Since every GH-inequality with a disconnected side $\bar{S}$ is redundant by Theorem 16, removing the covered IISs already shows feasibility of the complete network problem, and $\mathcal{C}$ is an IIS arc cover for all IISs. □

Note that the condition is not necessary: A redundant IIS can correspond to a non-redundant GH-inequality. This is obvious for $\mathcal{I}(S)$ and $\mathcal{I}(\bar{S})$, since one of them is always redundant for IACs, even if both S and $\bar{S}$ are connected.

Remark 19. For IIS node covers, Lemma 18 does not hold. A counterexample is given by $V = \{1, 2, 3\}$, $E = \{(1, 2), (1, 3)\}$, $b_1 = 5$, $b_2 = -2$, $b_3 = -3$, $u_{(1,2)} = 2$, $u_{(1,3)} = 2$. The IISs are $\mathcal{I}(\{1\})$, $\mathcal{I}(\{3\})$, $\mathcal{I}(\{1, 2\})$. Since nodes 2 and 3 are disconnected, $\mathcal{I}(\{1\})$ would be a candidate for being redundant, but $\{\sigma_2, \sigma_3\}$ is not a cover. Consequently, the lemma does not hold for IIS covers either.

In the cases of MIC and MINC, Theorem 16 cannot be used as in the proof of Lemma 18 for MIAC, since setting the supply value to $\pm\infty$ would lead to $b(V) \neq 0$, a crucial condition for the theorem to hold.

Redundancy shows us which IISs can be ignored in an arbitrary solution process, and we can use this statement for algorithmic purposes. Connectivity of subsets is fast to recognize in the case of leaves of the graph. Algorithm 2 removes all leaves and decides if the connecting arcs are in a cover.

The following lemma shows correctness of Algorithm 2. To this end, let $\mathrm{Opt}(G; b, u, \ell)$ be the set of optimal IIS arc covers to the network problem $(G; b, u, \ell)$. For a set C of arc covers, we denote the elementwise concatenation with a cover $\mathcal{C}$ by

$$\mathcal{C} \oplus C \coloneqq \big\{\{\mu_a \mid \mu_a \in \mathcal{C}\} \cup \{\lambda_a \mid \lambda_a \in \mathcal{C}\} \cup \mathcal{C}' \mid \mathcal{C}' \in C\big\}.$$

Algorithm 2 MIAC: Leaf Shrinking

Input: An infeasible network flow problem $(G; b, u, \ell)$
Output: A subset $\mathcal{C}$ of every minimal IAC and a possibly smaller network problem

```
 1: $\mathcal{C} \leftarrow \emptyset$
 2: $\hat{V} \leftarrow \{v \in V \mid |\delta(v)| = 1\}$                                  ▷ Leaves
 3: while $\hat{V} \neq \emptyset$ do
 4:     Choose $v \in \hat{V}$ arbitrarily
 5:     Let $a = (v, w) \in \delta^+(v)$ or $a = (w, v) \in \delta^-(v)$ be the unique arc incident to $v$
 6:     if $b(v) > u(\delta^+(v)) - \ell(\delta^-(v))$ then                       ▷ IIS in supply form
 7:         if $a \in \delta^+(v)$ then $\mathcal{C} \leftarrow \mathcal{C} \cup \mu_a$
 8:         else $\mathcal{C} \leftarrow \mathcal{C} \cup \lambda_a$
 9:     else if $-b(v) > u(\delta^-(v)) - \ell(\delta^+(v))$ then                 ▷ IIS in demand form
10:         if $a \in \delta^+(v)$ then $\mathcal{C} \leftarrow \mathcal{C} \cup \lambda_a$
11:         else $\mathcal{C} \leftarrow \mathcal{C} \cup \mu_a$
12:     $b(w) \leftarrow b(w) + b(v)$, $V \leftarrow V \setminus \{v\}$, $A \leftarrow A \setminus \{a\}$                 ▷ Shrink $a$
13:     if $|\delta(w)| = 1$ then $\hat{V} \leftarrow \hat{V} \cup \{w\}$                          ▷ New leaf
14:     $\hat{V} \leftarrow \hat{V} \setminus \{v\}$
15: return $\mathcal{C}, (G; b, u, \ell)$
```

Lemma 20. *Let $(G; b, u, \ell)$ be the input to Algorithm 2 and $\mathcal{C}$, $(\tilde{G}; \tilde{b}, \tilde{u}, \tilde{\ell})$ be the output. Then*

$$\operatorname{Opt}(G; b, u, \ell) = \mathcal{C} \oplus \operatorname{Opt}(\tilde{G}; \tilde{b}, \tilde{u}, \tilde{\ell}).$$

Proof. The algorithm only includes bounds λ_a or μ_a in Steps 7 or 8 and 10 or 11 if $a \in \delta(v)$ and $\mathcal{I}(\{v\})$ is an IIS, checked in Steps 6 and 9, respectively. Since $|\delta(v)| = 1$, either λ_a or μ_a must be contained in every cover in $\operatorname{Opt}(G; b, u, \ell)$.

On the other hand, the only bounds λ_a, μ_a which could be in $\operatorname{Opt}(G; b, u, \ell)$, but not in $\operatorname{Opt}(\tilde{G}; \tilde{b}, \tilde{u}, \tilde{\ell})$, are on arcs a removed in Step 12. Let v be chosen in Step 4 and a, w as in Step 5. If $\lambda_a, \mu_a \notin \mathcal{C}$, then $\mathcal{I}(v)$ is not an IIS, otherwise the bound would have been added to $\mathcal{C}$. But besides $S = \{v\}$, there can be no non-redundant IIS $\mathcal{I}(S)$ with $a \in \delta(S)$: If $v \in S$, then either $w \in S$ (and $a \notin \delta(S)$), or S is not connected. Hence, $\mathcal{I}(S)$ is not an IIS, but $\mathcal{I}(\bar{S})$ might be. However, because of Lemma 18, $\mathcal{I}(\bar{S})$ would be redundant. Therefore, $\lambda_a, \mu_a \notin \operatorname{Opt}(G; b, u, \ell)$. ☐

In a way, bridges generalize the concept of leaves, and we can use the concept of redundant IISs to decompose the graph at them. Algorithm 3 checks for every bridge a if the connected components on either side of a yield an IIS, in which case bounds on a are the only possibility to cover them. After that, we can safely remove a:

Lemma 21. *Let $(G; b, u, \ell)$ be the input to Algorithm 3 and $\mathcal{C}$, $\mathcal{P}$ be the output. Then*

$$\operatorname{Opt}(G; b, u, \ell) = \mathcal{C} \oplus \bigcup_{(\tilde{G}; \tilde{b}, u, \ell) \in \mathcal{P}} \operatorname{Opt}(\tilde{G}; \tilde{b}, u, \ell).$$

Proof. The proof is an extension of the one for Lemma 20. If a bound is added to $\mathcal{C}$ in Steps 6 or 7, this bound must also be in $\operatorname{Opt}(G; b, u, \ell)$, since the IIS $\mathcal{I}(V_v)$ can only be covered by precisely this bound. Note that $\mathcal{I}(V_v)$ is indeed an IIS, otherwise (V_v, A_v) would be disconnected, which would imply that G is also disconnected since a is a bridge.

Algorithm 3 MIAC: Bridge Deletion

```
Input: An infeasible network flow problem $P = (G; b, u, \ell)$, $G = (V, A)$
Output: Infeasible network flow problems $\mathcal{P} = \{(\tilde{G}; \tilde{b}, u, \ell)\}$, where $\tilde{G} \subseteq G$, and a subset
    $\mathcal{C}$ of every minimal IAC
 1: $\mathcal{C} \leftarrow \emptyset$, $\mathcal{P} \leftarrow \{P\}$
 2: for all $(v, w) = a \in A$ do
 3:     Let $\tilde{P} = ((\tilde{V}, \tilde{A}); \tilde{b}, u, \ell) \in \mathcal{P}$ such that $a \in \tilde{A}$
 4:     if $(\tilde{V}, \tilde{A} \setminus \{a\})$ is disconnected then                    ▷ $a$ is a bridge
 5:         Let $G_v = (V_v, A_v)$, $G_w = (V_w, A_w)$ be the components with $v \in V_v$, $w \in V_w$
 6:         if $b(V_v) > u_a - \ell_a$ then $\mathcal{C} \leftarrow \mathcal{C} \cup \{\mu_a\}$                ▷ IIS in supply form
 7:         else if $b(V_v) < \ell_a - u_a$ then $\mathcal{C} \leftarrow \mathcal{C} \cup \{\lambda_a\}$         ▷ IIS in demand form
 8:         $b_v(\tilde{v}) \leftarrow \tilde{b}(\tilde{v})\ \forall \tilde{v} \in V_v \setminus \{v\}$, $b_v(v) \leftarrow \tilde{b}(v) + b(V_w)$      ▷ Decompose $V_v$
 9:         $b_w(\tilde{w}) \leftarrow \tilde{b}(\tilde{w})\ \forall \tilde{w} \in V_w \setminus \{w\}$, $b_w(w) \leftarrow \tilde{b}(w) + b(V_v)$      ▷ Decompose $V_w$
10:         $\mathcal{P} \setminus \{\tilde{P}\} \cup \{(G_v; b_v, u, \ell),\ (G_w; b_w, u, \ell)\}$
11: return $\mathcal{C}$, $\mathcal{P}$
```

On the other hand, assume that $\mathcal{I}(V_v)$ is feasible and hence, λ_a, $\mu_a \notin \mathcal{C}$. Then $\mathcal{I}(V_w)$ is feasible as well. It follows that for every IIS $\mathcal{I}(S)$ either v, $w \in S$ or v, $w \notin S$, because otherwise S or $\bar{S}$ would be disconnected. Therefore, the decomposition in Steps 8–9 does not lose (or add) any potential covers, since by construction any later feasibility test takes this requirement into account. □

Note that Algorithm 3 generalizes Algorithm 2; nevertheless, we will need Algorithm 2 in Sect. 4 in the stated form.

3.2 IIS Node Covers

We will now turn to some fundamental statements about IIS node covers. Different to IAC, it turns out that we do not need the requirement of balanced connected components in this case.

Lemma 22. *There always exists an IIS node cover.*

Proof. Obviously, $\{\sigma_v \,|\, v \in V\}$ is a cover for every network problem with consistent bounds. □

Similarly to IAC, the enumeration of minimal INCs might not be effective.

Lemma 23. *There can be exponentially many minimal INCs.*

Proof. See the example in Fig. 4. There are IISs $\mathcal{I}(\{1, 2\})$, $\mathcal{I}(\{3, 4\}), \dots, \mathcal{I}(\{n-1, n\})$. It holds that every minimal INC contains exactly one of σ_{i-1}, σ_i for $i \in V$ with $i \equiv 0 \mod 2$. Hence, there are $2^{n/2}$ minimal INCs. The minimal INCs are also minimum in this case. □

The following lemma about the number of IISs will translate to the size of IIS covers.

Lemma 24. *For an infeasible flow problem with $b(V) = 0$, there are at least two IISs without a common node.*

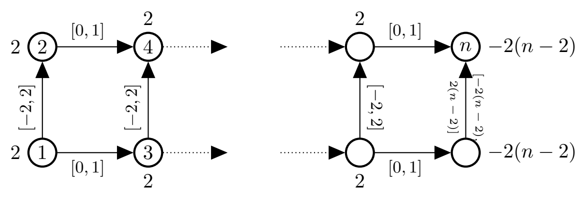


Figure 4: Network problem (node labels b, arc labels $[\ell, u]$) with an exponential number of minimal INCs.

Proof. For an infeasible flow problem, there are at least two violated GH-inequalities for the sets S and $\bar{S}$. Suppose S is disconnected. Since an infeasible system must have at least one IIS, $\bar{S}$ must be connected. Let S decompose into the connected components S_i, $i = 1, \dots k$. We have shown in [23] that then at least one of $\mathcal{I}(S_i)$ is an IIS. □

We can now give bounds for the size of IIS node covers. Unlike in the IAC case, these hold for all INCs, not just minimal ones.

Lemma 25. *Let $\mathcal{C}$ be an INC. Then $2 \leq |\mathcal{C}| \leq n$ and these bounds are tight, even for minimum INCs.*

Proof. Since there always exists an INC by Lemma 22, $|\mathcal{C}| \leq n$. Lemma 24 already implies that $|\mathcal{C}| \geq 2$. An example with min $|\mathcal{C}| = n$ is the problem $V = [n]$, $A = \{(i, i+1) \mid i \in [n-1]\}$, $b_i = 2$ for $i \in [n-1]$, $b_n = -2(n-1)$, $u \equiv \mathbb{1}$, and $\ell \equiv 0$. □

As mentioned in Remark 19, we cannot express the covering of node constraints by setting b_v to $\pm\infty$. However, to every INC, there is a corresponding feasible flow, which leads to a feasible supply vector $(b + d)$. This is also known as the excess of a node in the context of max-flow algorithms, see, e.g., [2].

Lemma 26. *Let $b \in \mathbb{R}^V$ with $b(V) = 0$. For every INC $\mathcal{C}$, there exists $d \in \mathbb{R}^V$ with $d_v = 0$ for $v \notin V(\mathcal{C})$ and $\mathbb{1}^\top d = 0$ such that $Mx = b + d$, $\ell \leq x \leq u$ is feasible.*

Proof. Since $\mathcal{C}$ is an INC, there exists a $\hat{x}$ feasible for $\mathcal{F} \setminus \mathcal{C}$. Let d_v be the excess of $v \in V$ with respect to $\hat{x}$, i.e., $d_v := \hat{x}(\delta^+(v)) - \hat{x}(\delta^-(v)) - b_v$. Then $d_v = 0$ for all $v \in V \setminus V(\mathcal{C})$ and $\mathbb{1}^\top d = \mathbb{1}^\top(M\hat{x} - b) = (\mathbb{1}^\top M)\hat{x} - \mathbb{1}^\top b = 0^\top \hat{x} - 0 = 0$. □

Unfortunately, other then the infinite bounds for IAC, we cannot use this at intermediate steps of an INC calculation, since the values for d can only be determined efficiently *after* the complete node cover is known.

3.3 Relation between IIS Node and Arc Covers

We are now concerned with the relation between IIS arc and node covers. The following lemma links the size of IIS arc and node covers to one another.

Lemma 27. *Let $\mathcal{C}^A$ be a minimum IIS arc cover and $\mathcal{C}^V$ a minimum IIS node cover for an arbitrary flow problem. It holds that*

a. $2\,|\mathcal{C}^A| \geq |\mathcal{C}^V|$ and

b. there is no constant $c \in \mathbb{R}$ such that $c\,|\mathcal{C}^V| \geq |\mathcal{C}^A|$ for all problems.

Proof.

a. Let $\mathcal{C}^A$ be an IIS arc cover. Then $\mathcal{C}^V := \{\sigma_v, \sigma_w \mid \mu_{(v,w)} \in \mathcal{C}^A\} \cup \{\sigma_v, \sigma_w \mid \lambda_{(v,w)} \in \mathcal{C}^A\}$ is an IIS node cover and $2\,|\mathcal{C}^A| \geq |\mathcal{C}^V|$.

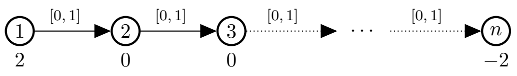
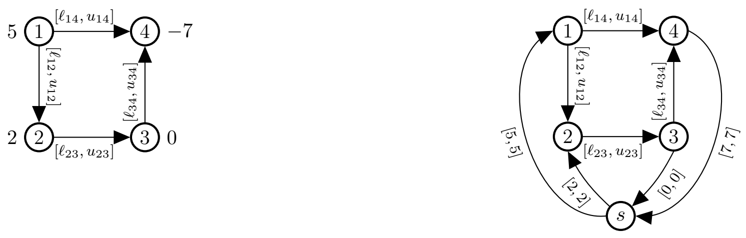


Figure 5: Network problem (node labels b, arc labels $[\ell, u]$), where the size of MINC is 2 and the size of MIAC is $n - 1$.



(a) MIC instance (b) reformulated as MIAC instance

Figure 6: Example for the transformation from a network instance where node and arc violations are allowed to a circulation instance where only arc violations are permitted; node labels (on the left side) b, arc labels $[\ell, u]$.

b. Consider the following network problem with n nodes, source node 1, and sink n: $V = [n]$, $A = \{(i, i+1) \mid i \in [n-1]\}$, $b_1 = 2$, $b_n = -2$, and $b_i = 0$ for $i = 2, \dots, n-1$ (see also Fig. 5). For $u = \mathbb{1}$, $\ell = 0$, $\mathcal{C}^V = \{\sigma_1, \sigma_n\}$ is the minimum IIS node cover, and the minimum arc cover is $\mathcal{C}^A = \{\mu_a \mid a \in A\}$. For $n \to \infty$, $|\mathcal{C}^A| \to \infty$, while $|C^V| = 2$. □

Lemma 27 a. could additionally be used for algorithmic purposes: By computing a (minimum) IAC, the construction in the proof gives us an INC, although not necessarily a good one.

Conversely, we can also construct an IAC from an INC, although without any constant estimation, as Lemma 27 b. emphasizes: For a given node cover, let d denote the corresponding excess, as defined in Lemma 26, so that $Mx = b + d$ is feasible. We can obtain an IAC by determining shortest undirected paths between the nodes with positive and negative excess and including all upper bounds for forward arcs and all lower bounds for backward arcs in the arc cover. A possibly better heuristic would be to solve a min-cost flow problem $\min\{\mathbb{1}^\top x \mid Mx = d\}$. For a node cover with few nodes and short paths, this might lead to a small arc cover. In general, though, the optimal assignment of flow is again an instance of MIAC.

3.4 IIS Cover

McCormick [26] gave a reformulation for his slack formulation (3), which transforms the case of both node and arc violations to the case of only arc violations. This essentially amounts to the classical reformulation as a circulation problem with additional lower bounds: Add a node s, arcs $a = (s, v)$ with $\ell_a = u_a = b_v$ for all $v \in V$ with $b_v > 0$, arcs $a = (v, s)$ with $\ell_a = u_a = -b_v$ for all $v \in V$ with $b_v \leq 0$, and set all $b_v = 0$ (see also Fig. 6). Hereby, the violation of the bounds for $\delta(s)$ in an "arc" solution is in one-to-one correspondence to the violation of a node constraint in a "node-arc" solution.

Since this reformulation works independently from the objective function, we can use it to formulate MIC as a MIAC instance.

For purely algorithmic purposes, we can therefore use methods developed for MIAC.

Nevertheless, the theoretical observations for MIAC do not necessarily carry over to MIC. Whenever we take advantage of a special graph structure (as in most of Sect. 4), it is probably lost after the reformulation, e.g., adding the auxiliary arcs to a tree will inevitably produce cycles. However, the approximability result in Theorem 14 does not use any information about the graph (besides connectivity), whence we have the following result for MIC.

Corollary 28. *MIC is approximable within* $c\,n$ *for every constant* $c > 0$ *in polynomial time.*

The best known approximability factor from general linear systems is $m + 1$ (see Amaldi and Kann [5]), so for the special case of flow networks, we have a better result, since $m + 1 \geq n$ for connected graphs.

The next section deals primarily with MIAC. We have picked MIAC as our base problem for the deeper analysis of the hardness because of the reducibility of MIC to it and the possibility to construct a node from an arc cover with at most a constant increase in size.

4 Boundaries of the Hardness of Minimum IIS Arc Cover

In this section, we will analyze the difficulty of solving MIAC in detail. To this end, we regard special cases of instances, where we restrict, on the one hand, the allowed data and on the other hand, graph structures.

First, we require the graph to be bipartite. By tweaking the set cover reduction from Proposition 7, we can show non-approximability and even combine the restrictions bipartite graph and ternary data for MIAC.

Theorem 29. *MIAC is not approximable in polynomial time within* $c \ln n$ *for any constant* $0 < c < 1$ *for assignment problems, i.e., on bipartite, directed graphs* $G = (V \dot\cup W, A)$ *with* $b_v = 1$ *for all* $v \in V$*,* $b_w = -1$ *for all* $w \in W$*,* $u \equiv 1$*,* $\ell \equiv 0$*, and all arcs directed from* V *to* W*, unless* $\mathcal{P} = \mathcal{NP}$*.*

Proof. Let again a set cover problem be given by a set R and a collection D of subsets $d \subset R$, where we are looking for a cover $D' \subseteq D$ of R. We start the construction of the MIAC instance as in Proposition 7 with the graph representation of the set cover instance and add some more nodes to obtain the claimed demands (see also Fig. 7): Since the nodes w_d representing the subsets D must have a demand, we need nodes v_d for $d \in D$ with a supply of 1, connected to w_d. Moreover, instead of t we use a set of nodes w_i, $i \in [|R|]$ and v_t, w_t, where v_t replaces t and each w_i is connected to v_t. Hence,

$$
\begin{aligned}
V_R &:= \{v_r \mid r \in R\},\ V_D := \{v_d \mid d \in D\},\\
V &:= V_R \cup V_D \cup \{v_t\},\\
W_D &:= \{w_d \mid d \in D\},\ W_R := \{w_r \mid r \in R\},\\
W &:= W_D \cup W_R \cup \{w_t\},\\
A_1 &:= \{(v_r, w_d) \mid r \in R,\ d \in D,\ r \in d\},\ A_2 := \{(v_d, w_d) \mid d \in D\},\\
A_3 &:= \{(v_t, w_d) \mid d \in D\},\ A_4 := \{(v_t, w_r) \mid r \in R\},\\
A &:= A_1 \cup A_2 \cup A_3 \cup A_4 \cup \{(v_t, w_t)\},\\
b_v &:= 1\ \forall\, v \in V,\ b_w := -1\ \forall\, w \in W,\\
u &\equiv \mathbb{1},\ \ell \equiv 0.
\end{aligned}
$$

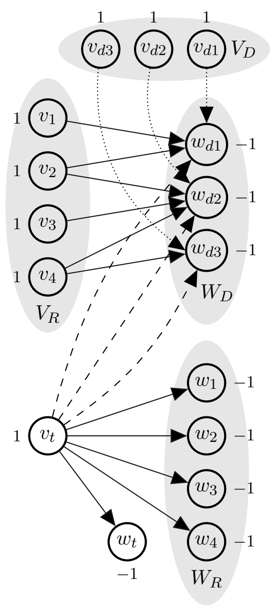


Figure 7: Sketch of the construction for the reduction from set cover to IAC for assignment problems, where R is the ground set and D is the collection of subsets of the set cover instance. All arc bounds are $[\ell, u] = [0, 1]$.

Regarding the cut $(V_R \cup V_D \cup W_D, \{v_t, w_t\} \cup W_R)$, we see that every flow satisfying flow conservation must send $|R|$ units backwards over A_3. Hence, at least one λ_a, $a \in A_3$, must be in the cover. The proof then proceeds as in the proof of Proposition 7, showing that there can be no smaller cover by using backward arcs in A_1 than the cover consisting of λ_a, $a \in \{(v_t, w_d) \mid d \in D'\}$. Hence, the size of the IIS arc cover equals the size of the set cover, and since the number of nodes in our construction is $2(|R| + |D| + 1)$, the non-approximability property carries over. □

Remark 30. For the relation between the min edge cost flow (MECF) problem and MIAC mentioned in the introduction, we regard a special case of MECF: If the capacities are infinite (the problem is then called ICF), we can transform (to some extent) ICF to an IIS arc cover problem by setting the upper bound in the arc cover instance to 0 for every cost $c_a = 1$ in the edge cost flow problem, see (4). Despite this transformation, some caution is necessary: In a MIAC instance, we can also use “backward arcs” by relaxing the lower bound. Hereby, the solution set can be larger than in the MECF instance; in particular, there might be a different minimum.

Regarding the special case of bipartite graphs, Benoist and Chauvet [8] showed that then ICF with costs $c_a = 1$ for every arc is hard, yet the caveat holds true in this case: We cannot simply transform MIAC from the ICF instance, since we might get a better solution by using backward arcs. On the other hand, their reduction from 3-Partition can be used almost without change to show strong $\mathcal{NP}$-completeness of MIAC for bipartite graphs. However, we obtained a stronger result by the set cover reduction.

This result also separates MIAC and MECF (with unitary costs). The latter is easy on the above special case of assignment problems: Either a perfect matching exists – which is the optimal solution for MECF – or there is no perfect matching, and the instance is infeasible for MECF.

On the other hand, there are also some polynomially solvable special cases, regarding the graph structure. Recall that tree and cycle are meant in the undirected sense in the following lemmas.

Lemma 31. *If G is a tree, MIAC is solvable in time $\mathcal{O}(m) = \mathcal{O}(n)$.*

Proof. Algorithm 2 applied to a tree starts with the leaves and finishes with $V = \emptyset$ and, by Lemma 20, an optimal cover. In order to turn this into a linear time algorithm, we proceed as follows.

We first perform a breadth first search (BFS) starting with an arbitrary node as root. Using bucket sort, we order the nodes according to their depth in the tree. We now perform Algorithm 2 from bottom to top, picking the nodes layer by layer. In this way, we make sure that the shrinking operations in Step 12 do not interfere. In Step 5 and Step 12, we use the unique parent arc, which can be obtained during BFS. This procedure runs in linear time, since BFS, bucket sort, and this implementation of Algorithm 2 all run in linear time. □

With a different approach, we can also solve cycle graphs. For this, we will use the following definitions: A *cycle*, sometimes also called a closed walk, is a sequence of nodes starting and ending at the same node, where each node is adjacent to the node following in the sequence. A cycle is *simple* if no edge appears more than once in this cycle; a cycle is *elementary* if no node, besides the start and end node, appears more than once.

Lemma 32. *If G is a cycle graph, MIAC is solvable in time $\mathcal{O}(m \log m) = \mathcal{O}(n \log n)$.*

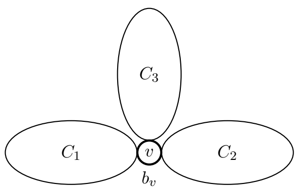
Proof. Let $G = (V, A)$ be a cycle graph and $\mathcal{C}^*$ an optimal IAC. We first show that there is a feasible flow x^* of $\mathcal{F} \setminus \mathcal{C}^*$ that satisfies at least one flow bound with equality. Suppose otherwise, i.e., $\ell_a < x^*_a < u_a$ for all $a \in A \setminus A(\mathcal{C}^*)$. Let $r = \min\{|\ell_a - x^*_a|, |u_a - x^*_a| \mid a \in A\}$ and augment x^* by r units of flow (in the respective direction). The resulting flow is feasible for $\mathcal{F} \setminus \mathcal{C}^*$ and satisfies at least one bound with equality.

Since G is a cycle, the solution space $\{x \mid Mx = b\}$ is 1-dimensional. Thus, fixing the flow value on one arc uniquely determines a solution, which can be computed in $\mathcal{O}(m)$ time. Consequently, to find x^* (and hence $\mathcal{C}^*$) it suffices to enumerate the solutions obtained by fixing the flow to the lower and upper bound for each arc. This can be efficiently done as follows.

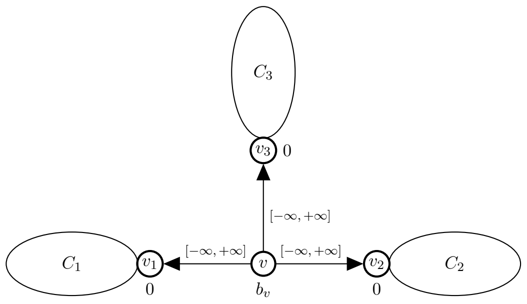
Consider an arbitrary solution $\tilde{x}$ of the equation system. Moreover, fix an arbitrary orientation of G and let $\chi \in \{-1, 1\}^A$ with $\chi_a = +1$ if $a \in A$ corresponds to the forward direction and $\chi_a = -1$ otherwise. We then have: $\{x \mid Mx = b\} = \{\tilde{x} + \lambda \chi \mid \lambda \in \mathbb{R}\}$. Now for every arc $a \in A$ compute two values of λ such that $\tilde{x}_a + \lambda \chi_a$ equals ℓ_a and u_a, respectively. Sorting these $2m$ points gives a partition of the real line into $2m+1$ intervals. We then traverse these points from smallest to largest, i.e., the intervals from left to right. Each time we reach a lower bound, the number of satisfied bounds of the interval starting at this point is increased; each time we reach an upper bound, the number of satisfied bounds for the next interval is decreased. We then take the minimal number of violated bounds. This yields a $\mathcal{O}(m \log m)$ time algorithm, dominated by the time to sort the intervals. □

In fact, we can combine Lemma 31 and Lemma 32. To this end, and for the next proposition, we will need to enumerate elementary, undirected cycles.

Remark 33. There are a several known algorithms that enumerate all elementary cycles of a directed graph. One of the best running times is obtained by an algorithm of Szwarcfiter and Lauer [32], which runs in $\mathcal{O}(n + m(\alpha + 1))$, where α denotes the number



(a) Three simple cycles C_1, C_2, C_3, each containing node v.



(b) Cycles connected via copies of v.

Figure 8: Transformation from simple cycles sharing a single node to elementary cycles; node labels b, arc labels $[\ell, u]$.

of found cycles. Mateti and Deo [25] showed how enumeration algorithms for directed graphs can be modified to work on undirected graphs by doubling the edges. The resulting algorithm then finds $2\alpha+m$, instead of α, cycles. Hence, we can enumerate all elementary, undirected cycles in time $\mathcal{O}(n + m^2 + m(\alpha + 1))$.

Theorem 34. *MIAC is solvable in time $\mathcal{O}(n + m^2)$ if G is a (simple) cactus graph, i.e., every cycle in G is simple or equivalently, G does not have the diamond graph (the complete graph on four nodes minus one edge) as a minor.*

Proof. First, we transform a cactus graph G to a graph $\tilde{G}$ in which every cycle is elementary (see also Fig. 8): Consider each node v that is contained in $k > 1$ elementary cycles $C_1, \dots, C_k$. Introduce k new nodes $v_1, \dots, v_k$ and replace v in cycle C_i by v_i, $i = 1, \dots, k$. Connect these nodes by arcs $a_i = (v, v_i)$ with $u_{a_i} = \infty$, $\ell_{a_i} = -\infty$. The supply resides at v and $b_{v_i} = 0$, $i = 1, \dots, k$.

By Remark 33, all elementary cycles in G can be enumerated in $\mathcal{O}(n+m^2+m(\alpha+1))$ time. Since every cycle is simple and must have at least three arcs, there can be at most $m/3$ elementary cycles. We can then check whether a node is contained in more than one cycle and perform the above transformation in time $\mathcal{O}(m^2)$.

Afterwards, we use the Bridge-Preprocessing (Algorithm 3) to decide optimally about all bounds on bridges in $\tilde{G}$; one can show that this can be done in $\mathcal{O}(m^2)$ time. Since all cycles in $\tilde{G}$ are elementary, we end up with a collection of disconnected cycles, which can be handled independently. By Lemma 32, this step takes $\mathcal{O}(m \log m)$ time. In total, we obtain an $\mathcal{O}(n + m^2)$ time algorithm. □

With the previous results, we try to pinpoint what causes the hardness of the problem. As Theorem 29 shows, the problem is still hard for "easy" data. All polynomially solvable special cases considered here share a certain characteristic: The number of paths – or routing possibilities – between sources and sinks is polynomially bounded. This marks the boundary between easy and hard problems, as the next result emphasizes.

We need the maximal number α of elementary cycles, such that each cycle shares an arc with at least one other cycle (see Fig. 9 for an example). The following proposition then essentially states that we could solve MIAC by enumerating the possible cycle-flows.

Proposition 35. *MIAC is fixed-parameter tractable (FPT) with respect to the maximal number α of elementary cycles connected at arcs.*

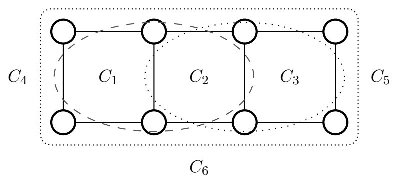


Figure 9: Graph with $\alpha = 6$ elementary cycles.

Proof. For the MIAC instance $(G; b, u, \ell)$, we first apply the transformation as in the proof of Theorem 34 (Fig. 8) and apply Algorithm 3 in $\mathcal{O}(m^2)$ time. The result is a set of k components that do not contain bridges or articulation nodes. We can then handle these components independently.

Consider the ith component with n_i nodes, m_i arcs, and c_i elementary cycles. The decomposition implies that each elementary cycle in the ith component has an arc in common with some other elementary cycle (if it exists). It follows that $c_i \leq \alpha$. Using Remark 33, we can enumerate these elementary cycles in time $\mathcal{O}(n_i + m_i^2 + m_i(c_i + 1))$.

Now there exists a flow x^* for the optimal cover which satisfies at least one remaining bound in each of the elementary cycles with equality: For each cycle for which no bound is satisfied with equality by x^*, we augment as in the proof of Lemma 32; this does not affect the flow that meets the bounds on any of the other elementary cycles.

Let m_{ij}, $j \in [c_i]$, be the number of arcs of the j-th elementary cycle. To find an optimal IAC, we now fix the flow to either the lower or upper bound on one arc for each elementary cycle, yielding $2m_{i,1}\,2m_{i,2} \cdots 2m_{i,c_i}$ possibilities. The remaining part of the graph forms a forest, and the corresponding flow (if it exists) can be computed in $\mathcal{O}(m_i)$ time, see Lemma 31. We then count the number of violated bounds for this flow. Because $m_{ij} \leq m_i$ for all $j \in [c_i]$, we can solve MIAC on this component in time $\mathcal{O}(m_i^{c_i+1})$ time.

Since the components are disjoint, $n_1 + \cdots + n_k \leq n$, $m_1 + \cdots + m_k \leq m$, and $c_i \leq \alpha$, we obtain a total running time of $\mathcal{O}(n + m^2 + m(\alpha + 1) + m^{\alpha+1})$. □

We can also show a more "classical" fixed-parameter result for MIAC using the treewidth. To this end, we will first need the following observation, which is used in the proof of Theorem 38 below.

In Algorithm 1, we have seen how a cover with two oppositely directed bounds can be transformed to a cover with only one bound per variable. Pursuing this principle, we define the following (theoretical) variant of an IIS arc cover.

Definition 36. *A subset of arcs K such that $\mathcal{C}(K) := \{\mu_a, \lambda_a \,|\, a \in K\}$ is an IAC is called* open arc cover.

One could say that the difference between open arc covers and IIS arc covers is the way of determining their size: For IACs we count the number of bounds, for open arc covers the number of affected arcs.

Lemma 37. *Minimum open arc cover and MIAC are equivalent w.r.t. their optimal solutions.*

Proof. Let $\mathcal{C}^*$ be a MIAC and K^* be a minimum open arc cover. Clearly, for $K := A(\mathcal{C}^*)$, $\mathcal{F} \backslash \mathcal{C}(K)$ is feasible. Hence, K is an open arc cover. By Lemma 9, $\mathcal{C}^*$ does not contain both upper and lower bounds for any arc and thus $|\mathcal{C}^*| = |K|$. Therefore, $|\mathcal{C}^*| = |K| \geq |K^*|$.

On the other hand, as in Step 4 and 5 of Algorithm 1, compute a feasible solution x for $\mathcal{F} \setminus \mathcal{C}(K^*)$. Then $\mathcal{C} = \mathcal{C}(K^*) \setminus \{\mu_a \,|\, x_a \leq u_a\} \setminus \{\lambda_a \,|\, x_a \geq \ell_a\}$ is an IAC, so $|\mathcal{C}| \geq |\mathcal{C}^*|$, and by construction, $|\mathcal{C}| \leq |K^*|$. Hence, $|\mathcal{C}^*| \leq |\mathcal{C}| \leq |K^*| \leq |\mathcal{C}^*|$ and equality must hold throughout. □

In particular, this means that in a solution algorithm for MIAC, we can iteratively decide to include "arcs" in a cover without committing to a specific bound yet and still get an optimal solution. Regarding the running time of potential algorithms, it might still be better to forgo this possibility, since it adds one max-flow computation at the end of the algorithm. Nevertheless, since max-flow is polynomially solvable, if we show that minimum open arc cover is solvable in polynomial time (for a special case), it follows that MIAC is polynomially solvable. Furthermore, the result carries over to MIC without change.

As a further step towards an FPT result w.r.t. treewidth, we will make an excursion into propositional logic and use common notation from this area in the next paragraph. We will give a very brief description of the employed concepts and refer to [7] for a detailed introduction.

Arnborg et al. [6] showed that all graph properties definable in extended monadic second-order logic (EMS) can be decided in linear time for graphs of fixed treewidth. Following the definitions in [6], monadic second-order logic provides individual variables (denoted in lower case letters), the connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, as well as quantifiers $\exists$ and $\forall$. In distinction to first-order logic, there are also set variables (denoted in capital letters), membership $\in$ and quantifiers over set variables. Extended monadic second-order logic additionally allows to formulate certain sums of evaluations and certain objective functions. For more detailed explanations, we refer the reader to Arnborg et al. [6].

Theorem 38. *MIAC is FPT with respect to the treewidth of the underlying graph.*

Proof. With the result of Arnborg et al. [6], it suffices to give an EMS formulation for MIAC. In fact, we will work with the equivalent min open arc cover (see Lemma 37).

Let *Head* and *Tail* be binary predicates, such that $\mathit{Head}(v,a)$ holds if and only if v is the head of arc a and $\mathit{Tail}(v,a)$ holds if and only v is the tail of a. We regard the vectors b, u, and ℓ as functions of both nodes and arcs by extending their definitions with zero in the obvious way, e.g., $b(v) = b_v$ for all $v \in V$ and $b(a) = 0$ for all $a \in A$. Then we can define the formulas

$$\begin{aligned}
\mathit{Compl}(S,v) &\equiv v \in V \wedge \neg(v \in S),\\
\Delta^+(S,a) &\equiv \exists v \exists w (v \in S \wedge \mathit{Compl}(S,w) \wedge \mathit{Tail}(v,a) \wedge \mathit{Head}(w,a)),\\
\Delta^-(S,a) &\equiv \exists v \exists w (v \in S \wedge \mathit{Compl}(S,w) \wedge \mathit{Head}(v,a) \wedge \mathit{Tail}(w,a)),\\
\Delta(S,D_1,D_2) &\equiv \forall a(\Delta^+(S,a) \leftrightarrow a \in D_1) \wedge \forall a(\Delta^-(S,a) \leftrightarrow a \in D_2),\\
\mathit{Subset}_V(S) &\equiv \forall v(v \in S \rightarrow v \in V),\\
\mathit{Subset}_A(D) &\equiv \forall a(a \in D \rightarrow a \in A),\\
\psi(|X_1|_b, |X_2|_u, |X_3|_\ell) &\equiv \sum_{x \in X_1} b(x) - \sum_{x \in X_2} u(x) + \sum_{x \in X_3} \ell(x) \leq 0,\\
\mathit{GHSet}(S) &\equiv \forall D_1 \forall D_2 (\mathit{Subset}_A(D_1) \wedge \mathit{Subset}_A(D_2) \wedge \Delta(S,D_1,D_2)\\
&\qquad \rightarrow \neg\psi(|S|_b, |D_1|_u, |D_2|_\ell)),\\
\mathit{IAC}(C) &\equiv \mathit{Subset}_A(C) \wedge \forall S((\mathit{Subset}_V(S) \wedge \mathit{GHSet}(S))\\
&\qquad \rightarrow \exists a(a \in C \wedge (\Delta^+(S,a) \vee \Delta^-(S,a)))),\\
F(|X_1|_{-1}) &= \sum_{x \in X_1} -1.
\end{aligned}$$

Then MIAC can be formulated as

$$\max_{IAC(C)} F(C),$$

where the objective function conforms to EMS and expresses that C should be a minimum IAC. Note that a feasible / optimal cover of GH-sets will obviously also be a feasible / optimal IIS cover, whence we do not need to formulate the connectedness requirement for an *IIS* function, but can use the weaker *GHSet* function. □

With some additional formulas, this result also holds for MINC and MIC.

Proposition 39. *MINC and MIC are FPT with respect to the treewidth of the underlying graph.*

Proof. Supplementing the formulas in the proof of Theorem 38, we define

$$\begin{aligned} INC(C) &\equiv Subset_V(C) \wedge \forall S((Subset_V(S) \wedge GHSet(S)) \rightarrow \exists v(v \in C \wedge v \in S)), \\ Subset_{VA}(H) &\equiv \forall h(h \in H \rightarrow (h \in V \vee h \in A)), \\ IC(C) &\equiv Subset_{VA}(C) \wedge \forall S((Subset_V(S) \wedge GHSet(S)) \\ &\qquad \rightarrow (\exists v(v \in C \wedge v \in S) \vee \exists a(a \in C \wedge (\Delta^+(S,a) \vee \Delta^-(S,a))))) \end{aligned}$$

Then MINC and MIC are given by

$$\max_{INC(C)} F(C) \quad \text{and} \quad \max_{IC(C)} F(C),$$

respectively. □

5 Outlook

This paper has investigated the special characteristics of minimum IIS covers in flow networks. On the one hand, these problems are already hard to solve and hard to approximate. On the other hand, we have obtained exact and fixed parameter algorithms for graphs with a special structure.

Our next step will be to utilize these findings for algorithmic purposes, both optimally and heuristically. One of our motivations for this paper was the analysis of infeasible systems arising in stationary gas transportation, see, e.g., [24]; we hope to generalize the insight gained from pure flow networks to such broader network structures.

In Sect. 4, we have analyzed the complexity of MIAC in greater detail. Although we could show negative and positive approximation results, there is still a gap, which we leave for future research. Furthermore, finding easy special cases for MINC would be interesting, as would be predictions about when the covering of a node constraint in an IIS results in a smaller cover than the covering of an arc constraint for MIC instances.

To establish a closer relationship to network design problems, one can extend the given graph to a complete one by adding artificial arcs with 0 bounds. Then the characteristics of MIAC in this new graph can be investigated. Here, the occurrence of an artificial arc in the cover would dictate the construction of a new arc, e.g., a pipeline in an application.

Acknowledgements The authors thank the DFG for their support within Project A01 in CRC TRR 154.

References

[1] Agrawal, A., Klein, P., Ravi, R.: When trees collide: An approximation algorithm for the generalized steiner problem on networks. SIAM Journal on Computing 24(3), 440–456 (1995)

[2] Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Network flows: theory, algorithms, and applications. Prentice Hall (1993)

[3] Alimonti, P., Kann, V.: Hardness of approximating problems on cubic graphs. In: Bongiovanni, G., Bovet, D., Battista, G. (eds.) Algorithms and Complexity, Lecture Notes in Computer Science, vol. 1203, pp. 288–298. Springer Berlin Heidelberg (1997)

[4] Amaldi, E., Kann, V.: The complexity and approximability of finding maximum feasible subsystems of linear relations. Theoretical computer science 147(1), 181–210 (1995)

[5] Amaldi, E., Kann, V.: On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. Theoretical Computer Science 209(1), 237–260 (1998)

[6] Arnborg, S., Lagergren, J., Seese, D.: Easy problems for tree-decomposable graphs. Journal of Algorithms 12(2), 308–340 (1991)

[7] Barwise, J. (ed.): Handbook of mathematical logic. Elsevier (1982)

[8] Benoist, T., Chauvet, F.: Complexity of some FPP related problems. Tech. rep., e-lab (2001)

[9] Chakravarti, N.: Some results concerning post-infeasibility analysis. European Journal of Operational Research 73(1), 139–143 (1994)

[10] Chinneck, J.W.: An effective polynomial-time heuristic for the minimum-cardinality IIS set-covering problem. Ann. Math. Artif. Intell. 17(1–2), 127–144 (1996)

[11] Chinneck, J.W.: Fast heuristics for the maximum feasible subsystem problem. INFORMS J. Comput. 13(3), 210–223 (2001)

[12] Chinneck, J.W.: Feasibility and infeasibility in optimization: algorithms and computational methods, International Series in Operations Research and Management Sciences, vol. 118. Springer (2008)

[13] Di Gaspero, L., Gärtner, J., Kortsarz, G., Musliu, N., Schaerf, A., Slany, W.: The minimum shift design problem. Ann. Oper. Res. 155(1), 79–105 (2007)

[14] Gale, D.: A theorem on flows in networks. Pacific J. Math. 7(2), 1073–1082 (1957)

[15] Garey, M.R., Johnson, D.S.: Computers and intractability, vol. 174. Freeman New York (1979)

[16] Gleeson, J., Ryan, J.: Identifying minimally infeasible subsystems of inequalities. ORSA J. Comput. 2(1), 61–63 (1990)

[17] Goemans, M.X., Goldberg, A.V., Plotkin, S.A., Shmoys, D.B., Tardos, E., Williamson, D.P.: Improved approximation algorithms for network design problems. In: SODA. vol. 94, pp. 223–232 (1994)

[18] Goemans, M.X., Williamson, D.P.: The primal-dual method for approximation algorithms and its application to network design problems. In: Hochbaum, D.S. (ed.) Approximation algorithms for NP-hard problems, pp. 144–191. PWS Publishing Co. (1997)

[19] Greenberg, H.J.: Diagnosing infeasibility in min-cost network flow problems part I: Dual infeasibility. IMA J. Manag. Math. 1(2), 99–109 (1987)

[20] Greenberg, H.J.: Diagnosing infeasibility in min-cost network flow problems part II: Primal infeasibility. IMA J. Manag. Math. 2(1), 39–50 (1988)

[21] Greenberg, H.J.: Consistency, redundancy, and implied equalities in linear systems. Ann. Math. Artif. Intell. 17(1), 37–83 (1996)

[22] Hoffman, A.J.: Some recent applications of the theory of linear inequalities to extremal combinatorial analysis. In: Proc. Sympos. Appl. Math. vol. 10, pp. 113–127. World Scientific (1960)

[23] Joormann, I., Pfetsch, M.E.: On the relation of flow cuts and irreducible infeasible subsystems. Tech. rep., Optimization Online (March 2014), `http://www.optimization-online.org/DB_HTML/2014/03/4277.html`

[24] Joormann, I., Schmidt, M., Steinbach, M.C., Willert, B.M.: What does "feasible" mean? In: Koch, T., Hiller, B., Pfetsch, M.E., Schewe, L. (eds.) Evaluating gas network capacities, chap. 11, pp. 211–232. MOS-SIAM Series on Optimization, SIAM (2015)

[25] Mateti, P., Deo, N.: On algorithms for enumerating all circuits of a graph. SIAM Journal on Computing 5(1), 90–99 (1976)

[26] McCormick, S.T.: How to compute least infeasible flows. Math. Progam. 78(2), 179–194 (1997)

[27] Moshkovitz, D.: The projection games conjecture and the NP-hardness of ln n-approximating set-cover (July 2014), `http://people.csail.mit.edu/dmoshkov/papers/set-cover/set-cover-full.pdf`

[28] Motzkin, T.S.: Beiträge zur Theorie der Linearen Ungleichungen. Ph.D. thesis, Basel (1933), english version: "Contributions to the Theory of Linear Inequalities", translated by D. R. Fulkerson, in "Theodore S. Motzkin: Selected Papers", D. Cantor, B. Gordon and B. Rothschild, Eds., Birkhäuser, Boston (1983).

[29] Parker, M., Ryan, J.: Finding the minimum weight IIS cover of an infeasible system of linear inequalities. Ann. Math. Artif. Intell. 17(1), 107–126 (1996)

[30] Pfetsch, M.E.: Branch-and-cut for the maximum feasible subsystem problem. SIAM J. Optim. 19(1), 21–38 (2008)

[31] Sankaran, J.K.: A note on resolving infeasibility in linear programs by constraint relaxation. Operations Research Letters 13(1), 19–20 (1993)

[32] Szwarcfiter, J., Lauer, P.: A search strategy for the elementary cycles of a directed graph. BIT Numerical Mathematics 16(2), 192–204 (1976)

[33] Wallace, S.W., Wets, R.J.B.: The facets of the polyhedral set determined by the Gale-Hoffman inequalities. Math. Program. 62(1), 215–222 (1993)