Integer Programming and Incidence Treedepth

Eduard Eiben[1], Robert Ganian[2], Dušan Knop[3,4], Sebastian Ordyniak[5](✉), Michał Pilipczuk[6], and Marcin Wrochna[6,7]

[1] Department of Informatics, University of Bergen, Bergen, Norway
eduard.eiben@uib.no
[2] Algorithms and Complexity Group, Technische Universität Wien, Vienna, Austria
rganian@ac.tuwien.ac.at
[3] Algorithmics and Computational Complexity, Faculty IV, TU Berlin, Berlin, Germany
[4] Department of Theoretical Computer Science, Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic
dusan.knop@fit.cvut.cz
[5] Algorithms Group, Department of Computer Science, University of Sheffield, Sheffield, UK
s.ordyniak@sheffield.ac.uk
[6] Institute of Informatics, University of Warsaw, Warsaw, Poland
{michal.pilipczuk,marcin.wrochna}@mimuw.edu.pl
[7] University of Oxford, Oxford, UK

Abstract. Recently a strong connection has been shown between the tractability of integer programming (IP) with bounded coefficients on the one side and the structure of its constraint matrix on the other side. To that end, integer linear programming is fixed-parameter tractable with respect to the primal (or dual) treedepth of the Gaifman graph of its constraint matrix and the largest coefficient (in absolute value). Motivated by this, Koutecký, Levin, and Onn [ICALP 2018] asked whether it is possible to extend these result to a more broader class of integer linear programs. More formally, is integer linear programming fixed-parameter tractable with respect to the incidence treedepth of its constraint matrix and the largest coefficient (in absolute value)?

We answer this question in negative. We prove that deciding the feasibility of a system in the standard form, $A\mathbf{x} = \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$, is NP-hard

Eduard Eiben was supported by Pareto-Optimal Parameterized Algorithms (ERC Starting Grant 715744). This work is a part of projects CUTACOMBS, PowAlgDO (M. Wrochna) and TOTAL (M. Pilipczuk) that have received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreements No. 714704, No. 714532, and No. 677651). Dušan Knop is supported by DFG, project "MaMu", NI 369/19. Marcin Wrochna is supported by Foundation for Polish Science (FNP) via the START stipend. Robert Ganian acknowledges support from the FWF Austrian Science Fund (Project P31336: NFPC) and is also affiliated with FI MU, Brno, Czech Republic.

A. Lodi and V. Nagarajan (Eds.): IPCO 2019, LNCS 11480, pp. 194–204, 2019.
https://doi.org/10.1007/978-3-030-17953-3_15

even when the absolute value of any coefficient in A is 1 and the incidence treedepth of A is 5. Consequently, it is not possible to decide feasibility in polynomial time even if both the assumed parameters are constant, unless $\mathsf{P} = \mathsf{NP}$.

Keywords: Integer programming · Incidence treedepth · Gaifman graph · Computational complexity

1 Introduction

In this paper we consider the decision version of Integer Linear Program (ILP) in *standard form*. Here, given a matrix $A \in \mathbb{Z}^{m\times n}$ with m rows (constraints) and n columns and vectors $\mathbf{b} \in \mathbb{Z}^m$ and $\mathbf{l}, \mathbf{u} \in \mathbb{Z}^n$ the task is to decide whether the set

$$\{\mathbf{x} \in \mathbb{Z}^n \mid A\mathbf{x} = \mathbf{b},\, \mathbf{l} \le \mathbf{x} \le \mathbf{u}\} \tag{SSol}$$

is non-empty. We are going to study structural properties of the incidence graph of the matrix A. An integer program (IP) is a *standard IP* (SIP) if its set of solutions is described by (SSol), that is, if it is of the form

$$\min\{f(\mathbf{x}) \mid A\mathbf{x} = \mathbf{b},\, \mathbf{l} \le \mathbf{x} \le \mathbf{u}\,, \mathbf{x} \in \mathbb{Z}^n\}\,, \tag{SIP}$$

where $f\colon \mathbb{N}^n \to \mathbb{N}$ is the *objective function*; in case f is a linear function the above SIP is said to be a linear SIP. Before we go into more details we first review some recent development concerning algorithms for solving (linear) SIPs in variable dimension with the matrix A admitting a certain decomposition.

Let E be a 2×2 block matrix, that is, $E = \left(\begin{smallmatrix} A_1 & A_2 \\ A_3 & A_4 \end{smallmatrix}\right)$, where $A_1, \dots, A_4$ are integral matrices. We define an *n-fold 4-block product of* E for a positive integer n as the following block matrix

$$E^{(n)} = \begin{pmatrix} A_1 & A_2 & A_2 & \cdots & A_2 \\ A_3 & A_4 & 0 & \cdots & 0 \\ A_3 & 0 & A_4 & \cdots & 0 \\ \vdots & & & \ddots & \\ A_3 & 0 & 0 & \cdots & A_4 \end{pmatrix},$$

where 0 is a matrix containing only zeros (of appropriate size). One can ask whether replacing A in the definition of the set of feasible solutions (SSol) can give us an algorithmic advantage leading to an efficient algorithm for solving such SIPs. We call such an SIP an *n-fold 4-block IP*. We derive two special cases of the n-fold 4-block IP with respect to special cases for the matrix E (see monographs [4,17] for more information). If both A_1 and A_3 are void (not present at all), then the result of replacing A with $E^{(n)}$ in (SIP) yields the *n-fold IP*. Similarly, if A_1 and A_2 are void, we obtain the *2-stage stochastic IP*.

The first, up to our knowledge, pioneering algorithmic work on n-fold 4-block IPs is due to Hemmecke et al. [9]. They gave an algorithm that given n, the 2×2 block matrix E, and vectors $\mathbf{w}, \mathbf{b}, \mathbf{l}, \mathbf{u}$ finds an integral vector $\mathbf{x}$ with $E^{(n)}\mathbf{x} = \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$ minimizing $\mathbf{wx}$. The algorithm of Hemmecke et al. [9] runs in time $n^{g(r,s,\|E\|_\infty)}L$, where r is the number of rows of E, s is the number of columns of E, L is the size of the input, and $g\colon \mathbb{N} \to \mathbb{N}$ is a computable function. Thus, from the parameterized complexity viewpoint this is an XP algorithm for parameters $r, s, \|E\|_\infty$. This algorithm has been recently improved by Chen et al. [3] who give better bounds on the function g; it is worth noting that Chen et al. [3] study also the special case where A_1 is a zero matrix and even in that case present an XP algorithm. Since the work of Hemmecke et al. [9] the question of whether it is possible to improve the algorithm to run in time $g'(r, s, \|E\|_\infty) \cdot n^{O(1)}L$ or not has become a major open question in the area of mathematical programming.

Of course, the complexity of the two aforementioned special cases of n-fold 4-block IP are extensively studied as well. The first FPT algorithm[1] for the n-fold IPs (for parameters $r, s, \|E\|_\infty$) is due to Hemmecke et al. [10]. Their algorithm has been subsequently improved [7,14]. Altmanová et al. [1] implemented the algorithm of Hemmecke et al. [10] and improved the polynomial factor (achieving the same running time as Eisenbrand et al. [7]) the above algorithms (from cubic dependence to $n^2 \log n$). The best running time of an algorithm solving n-fold IP is due to Jansen et al. [12] and runs in nearly linear time in terms of n.

Last but not least, there is an FPT algorithm for solving the 2-stage stochastic IP due to Hemmecke and Schultz [11]. This algorithm is, however, based on a well quasi ordering argument yielding a bound on the size of the Graver basis for these IPs. Very recently Klein [13] presented a constructive approach using Steinitz lemma and give the first explicit (and seemingly optimal) bound on the size of the Graver basis for 2-stage (and multistage) IPs. It is worth noting that possible applications of 2-stage stochastic IP are much less understood than those of its counterpart n-fold IP.

In the past few years, algorithmic research in this area has been mainly application-driven. Substantial effort has been taken in order to find the right formalism that is easier to understand and yields algorithms having the best possible ratio between their generality and the achieved running time. It turned out that the right formalism is connected with variants of the Gaifman graph (see e.g. [5]) of the matrix A (for the definitions see the Preliminaries section).

Our Contribution. In this paper we focus on the incidence (Gaifman) graph. We investigate the (negative) effect of the treedepth of the incidence Gaifman graph on tractability of ILP feasibility.

Theorem 1. *Given a matrix* $A \in \{-1, 0, 1\}^{m \times n}$ *and vectors* $\mathbf{l}, \mathbf{u} \in \mathbb{Z}^n_\infty$. *Deciding whether the set defined by* (SSol) is non-empty is NP-hard even if $\mathbf{b} = \mathbf{0}$ and $\mathrm{td}_I(A) \leq 5$.

[1] That is, an algorithm running in time $f(r, s, \|E\|_\infty) \cdot n^{O(1)}L$.

Preliminaries

For integers $m < n$ by $[m : n]$ we denote the set $\{m, m+1, \ldots, n\}$ and $[n]$ is a shorthand for $[1 : n]$. We use bold face letters for vectors and normal font when referring to their components, that is, $\mathbf{x}$ is a vector and x_3 is its third component. For vectors of vectors we first use superscripts to access the "inner vectors", that is, $\mathbf{x} = (\mathbf{x}^1, \ldots, \mathbf{x}^n)$ is a vector of vectors and $\mathbf{x}^3$ is the third vector in this collection.

From Matrices to Graphs. Let A be an $m \times n$ integer matrix. The *incidence Gaifman graph* of A is the bipartite graph $G_I = (R \cup C, E)$, where $R = \{r_1, \ldots, r_m\}$ contains one vertex for each row of A and $C = \{c_1, \ldots, c_n\}$ contains one vertex for each column of A. There is an edge $\{r, c\}$ between the vertex $r \in R$ and $c \in C$ if $A(r, c) \neq 0$, that is, if row r contains a nonzero coefficient in column c. The *primal Gaifman graph* of A is the graph $G_P = (C, E)$, where C is the set of columns of A and $\{c, c'\} \in E$ whenever there exists a row of A with a nonzero coefficient in both columns c and c'. The *dual Gaifman graph* of A is the graph $G_D = (R, E)$, where R is the set of rows of A and $\{r, r'\} \in E$ whenever there exists a column of A with a nonzero coefficient in both rows r and r'.

Treedepth. Undoubtedly, the most celebrated structural parameter for graphs is treewidth, however, in the case of ILPs bounding treewidth of any of the graphs defined above does not lead to tractability (even if the largest coefficient in A is bounded as well see e.g. [14, Lemma 18]). Treedepth is a structural parameter which is useful in the theory of so-called sparse graph classes, see e.g. [16]. Let $G = (V, E)$ be a graph. The treedepth of G, denoted $\operatorname{td}(G)$, is defined by the following recursive formula:

$$\operatorname{td}(G) = \begin{cases} 1 & \text{if } |V(G)| = 1, \\ 1 + \min_{v \in V(G)} \operatorname{td}(G - v) & \text{if } G \text{ is connected with } |V(G)| > 1, \\ \max_{i \in [k]} \operatorname{td}(G_i) & \text{if } G_1, \ldots, G_k \text{ are connected components of } G. \end{cases}$$

Let A be an $m \times n$ integer matrix. The *incidence treedepth* of A, denoted $\operatorname{td}_I(A)$, is the treedepth of its incidence Gaifman graph G_I. The *dual treedepth* of A, denoted $\operatorname{td}_D(A)$, is the treedepth of its dual Gaifman graph G_D. The *primal treedepth* is defined similarly.

The following two well-known theorems will be used in the proof of Theorem 1.

Theorem 2 (Chinese Remainder Theorem). *Let $p_1, \ldots, p_n$ be pairwise coprime integers greater than 1 and let $a_1, \ldots, a_n$ be integers such that for all $i \in [n]$ it holds $0 \leq a_i < p_i$. Then there exists exactly one integer x such that*

1. $0 \leq x < \prod_{i=1}^{n} p_i$ *and*
2. $\forall i \in [n] : \; x \equiv a_i \mod p_i$.

Theorem 3 (Prime Number Theorem). *Let $\pi(n)$ denote the number of primes in $[n]$, then $\pi(n) \in \Theta(\frac{n}{\log n})$.*

It is worth pointing out that, given a positive integer n encoded in unary, it is possible to the n-th prime in polynomial time.

2 Proof of Theorem 1

Before we proceed to the proof of Theorem 1 we include a brief sketch of its idea. To prove NP-hardness, we will give a polynomial time reduction from 3-SAT which is well known to be NP-complete [8]. The proof is inspired by the NP-hardness proof for ILPs given by a set of inequalities, where the primal graph is a star, of Eiben et al. [6].

Proof Idea. Let φ be a 3-CNF formula. We encode an assignment into a variable y. With every variable v_i of the formula φ we associate a prime number p_i. We make $y \bmod p_i$ be the boolean value of the variable v_i; i.e., using auxiliary gadgets we force $y \bmod p_i$ to always be in $\{0, 1\}$. Further, if for a clause $C \in \varphi$ by $\|C\|$ we denote the product of all of the primes associated with the variables occurring in C, then, by Chinese Remainder Theorem, there is a single value in $[\|C\|]$, associated with the assignment that falsifies C, which we have to forbid for $y \bmod \|C\|$. We use the box constraints, i.e., the vectors $\mathbf{l}, \mathbf{u}$, for an auxiliary variable taking the value $y \bmod \|C\|$ to achieve this. For example let $\varphi = (v_1 \vee \neg v_2 \vee v_3)$ and let the primes associated with the three variables be $2, 3$, and 5, respectively. Then we have $\|(v_1 \vee \neg v_2 \vee v_3)\| = 30$ and, since $v_1 = v_3 = \texttt{false}$ and $v_2 = \texttt{true}$ is the only assignment falsifying this clause, we have that 21 is the forbidden value for $y \bmod 30$. Finally, the (SIP) constructed from φ is feasible if and only if there is a satisfying assignment for φ.

Proof (of Theorem 1). Let φ be a 3-CNF formula with n' variables $v_1, \ldots, v_{n'}$ and m' clauses $C_1, \ldots, C_{m'}$ (an instance of 3-SAT). Note that we can assume that none of the clauses in φ contains a variable along with its negation. We will define an SIP, that is, vectors $\mathbf{b}, \mathbf{l}, \mathbf{u}$, and a matrix A with $\mathcal{O}((n' + m')^5)$ rows and columns, whose solution set is non-empty if and only if a satisfying assignment exists for φ. Furthermore, we present a decomposition of the incidence graph of the constructed SIP proving that its treedepth is at most 5. We naturally split the vector $\mathbf{x}$ of the SIP into subvectors associated with the sought satisfying assignment, variables, and clauses of φ, that is, we have $\mathbf{x} = \left(y, \mathbf{x}^1, \ldots, \mathbf{x}^{n'}, \mathbf{z}^1, \ldots, \mathbf{z}^{m'}\right)$. Throughout the proof p_i denotes the i-th prime number.

Variable Gadget. We associate the $\mathbf{x}^i = \left(x^i_0, \ldots, x^i_{p_i}\right)$ part of $\mathbf{x}$ with the variable v_i and bind the assignment of v_i to y. We add the following constraints

$$x^i_1 = x^i_\ell \qquad \forall \ell \in [2 : p_i] \tag{1}$$

$$x^i_0 = y + \sum_{\ell=1}^{p_i} x^i_\ell \tag{2}$$

and box constraints

$$-\infty \le x_\ell^i \le \infty \qquad \forall \ell \in [p_i] \tag{3}$$

$$0 \le x_0^i \le 1 \tag{4}$$

to the SIP constructed so far.

Claim. For given values of x_0^i and y, one may choose the values of x_ℓ^i for $\ell \in [p_i]$ so that (1) and (2) are satisfied if and only if $x_0^i \equiv y \mod p_i$.

Proof. By (1) we know $x_1^i = \cdots = x_{p_i}^i$ and thus by substitution we get the following equivalent form of (2)

$$x_0^i = y + p_i \cdot x_1^i \,. \tag{5}$$

But this form is equivalent to $x_0^i \equiv y \mod p_i$. ⌟

Note that by (the proof of) the above claim the conditions (1) and (2) essentially replace the large coefficient (p_i) used in the condition (5). This is an efficient trade-off between large coefficients and incidence treedepth which we are going to exploit once more when designing the clause gadget.

By the above claim we get an immediate correspondence between y and truth assignments for $v_1, \dots, v_{n'}$. For an integer w and a variable v_i we define the following mapping

$$\text{assignment}(w, v_i) = \begin{cases} \texttt{true} & \text{if } w \equiv 1 \mod p_i \\ \texttt{false} & \text{if } w \equiv 0 \mod p_i \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Notice that (4) implies that the the mapping $\text{assignment}(y, v_i) \in \{\texttt{true}, \texttt{false}\}$ for $i \in [n']$. We straightforwardly extend the mapping $\text{assignment}(\cdot, \cdot)$ for tuples of variables as follows. For a tuple $\mathbf{a}$ of length ℓ, the value of $\text{assignment}(w, \mathbf{a})$ is $(\text{assignment}(w, a_1), \dots, \text{assignment}(w, a_\ell))$ and we say that $\text{assignment}(w, \mathbf{a})$ is defined if all of its components are defined.

Clause Gadget. Let C_j be a clause with variables v_e, v_f, v_g. We define $\|C_j\|$ as the product of the primes associated with the variables occurring in C_j, that is, $\|C_j\| = p_e \cdot p_f \cdot p_g$. We associate the $\mathbf{z}^j = \left(z_0^j, \dots, z_{\|C_j\|}^j\right)$ part of $\mathbf{x}$ with the clause C_j. Let d_j be the unique integer in $[\|C_j\|]$ for which $\text{assignment}(d_j, (v_e, v_f, v_g))$ is defined and gives the falsifying assignment for C_j. The existence and uniqueness of d_j follows directly from the Chinese Remainder Theorem. We add the following constraints

$$z_1^j = z_\ell^j \qquad \forall \ell \in [2 : \|C_j\|] \tag{6}$$

$$z_0^j = y + \sum_{1 \le \ell \le \|C_j\|} z_\ell^j \tag{7}$$

and box constraints

$$-\infty \leq z^j_\ell \leq \infty \qquad \forall \ell \in [\|C_j\|] \tag{8}$$

$$d_j + 1 \leq z^j_0 \leq \|C_j\| + d_j - 1 \tag{9}$$

to the SIP constructed so far.

Claim. Let C_j be a clause in φ with variables v_e, v_f, v_g. For given values of y and z^j_0 such that assignment$(y, (v_e, v_f, v_g))$ is defined, one may choose the values of z^j_ℓ for $\ell \in [\|C_j\|]$ so that (6), (7), (8) and (9) are satisfied if and only if assignment$(y, (v_e, v_f, v_g))$ satisfies C_j.

Proof. Similarly to the proof of the first claim, (6) and (7) together are equivalent to $z^j_0 \equiv y \mod \|C_j\|$. Finally, by (9) we obtain that $z^j_0 \neq d_j$ which holds if and only if assignment$(y, (v_e, v_f, v_g))$ satisfies C_j. ⌟

Let $A\mathbf{x} = \mathbf{0}$ be the SIP with constraints (1), (2), (6), and (7) and box constraints $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$ given by (3), (4), (8), (9), and $-\infty \leq y \leq \infty$. By the first claim, constraints (1), (2), (3), (4), are equivalent to the assertion that assignment$(y, (v_1, \ldots, v_{n'}))$ is defined. Then by the second claim, constraints (6), (7), (8), (9) are equivalent to checking that every clause in φ is satisfied by assignment$(y, (v_1, \ldots, v_{n'}))$. This finishes the reduction and the proof of its correctness.

In order to finish the proof we have to bound the number of variables and constraints in the presented SIP and to bound the incidence treedepth of A. It follows from the Prime Number Theorem that $p_i = \mathcal{O}(i \log i)$. Hence, the number of rows and columns of A is at most $(n' + m')p^3_{n'} = \mathcal{O}((n' + m')^5)$.

Claim. It holds that $\mathrm{td}_I(A) \leq 5$.

Proof. Let G be the incidence graph of the matrix A. It is easy to verify that y is a cut-vertex in G. Observe that each component of $G - y$ is now either a variable gadget for v_i with $i \in [n']$ (we call such a component a *variable component*) or a clause gadget for C_j with $j \in [m']$ (we call such a component a *clause component*). Let G^i_v be the variable component (of $G - y$) containing variables $\mathbf{x}^i$ and G^j_c be the clause component containing variables $\mathbf{z}^j$. Let $t_v = \max_{\ell \in [n']} \mathrm{td}(G^\ell_v)$ and $t_c = \max_{\ell \in [m']} \mathrm{td}(G^\ell_c)$. It follows that $\mathrm{td}(G) \leq 1 + \max(t_v, t_c)$.

Refer to Fig. 1. Observe that if we delete the variable x^i_1 together with the constraint (2) from G^i_v, then each component in the resulting graph contains at most two vertices. Each of these components contains either

- a variable x^i_ℓ and an appropriate constraint (1) (the one containing x^i_ℓ and x^i_0) for some $\ell \in [2 : p_i]$ or
- the variable x^i_0.

Since treedepth of an edge is 2 and treedepth of the one vertex graph is 1, we have that $t_v \leq 4$.

The bound on t_c follows the same lines as for t_v, since indeed the two gadgets have the same structure. Now, after deleting z_1^j and (7) in G_c^j we arrive to a graph with treedepth of all of its components again bounded by two (in fact, none of its components contain more than two vertices). Thus, $t_v \leq 4$ and the claim follows. $\lrcorner$

The theorem follows by combining the three above claims. $\square$

3 Incidence Treedepth of Restricted ILPs

It is worth noting that the proof of Theorem 1 crucially relies on having variables as well as constraints which have high degree in the incidence graph. Thus, it is natural to ask whether this is necessary or, equivalently, whether bounding the degree of variables, constraints, or both leads to tractability. It is well known that if a graph G has bounded degree and treedepth, then it is of bounded size, since indeed the underlying decomposition tree has bounded height and degree and thus bounded number of vertices. Let (SIP) with n variables be given. Let $\mathrm{maxdeg}_C(A)$ denote the maximum arity of a constraint in its constraint matrix A and let $\mathrm{maxdeg}_V(A)$ denote the maximum occurrence of a variable in constraints of A. In other words, $\mathrm{maxdeg}_C(A)$ denotes the maximum number of nonzeros in a row of A and $\mathrm{maxdeg}_V(A)$ denotes the maximum number of nonzeros in a column of A. Now, we get that ILP can be solved in time $f(\mathrm{maxdeg}_C(A), \mathrm{maxdeg}_V(A), \mathrm{td}_I(A))L^{O(1)}$, where f is some computable function and L is the length of the encoding of the given ILP thanks to Lenstra's algorithm [15].

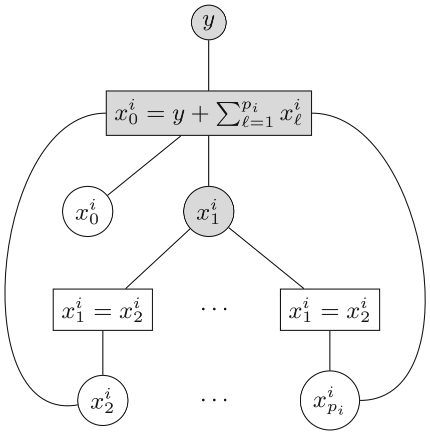

Fig. 1. The variable gadget for u_i of 3-SAT instance together with the global variable y. Variables (of the IP) are in circular nodes while equations are in rectangular ones. The nodes deleted in the proof of the third claim in the proof of Theorem 1 have light gray background.

The above observation can in fact be strengthened—namely, if the arity of all the constraints or the number of occurrences of all the variables in the given SIP is bounded, then we obtain a bound on either primal or dual treedepth. This is formalized by the following lemma.

Lemma 4. *For every (SIP) we have*

$$\mathrm{td}_P(A) \le \mathrm{maxdeg}_C(A) \cdot \mathrm{td}_I(A) \qquad \textit{and} \qquad \mathrm{td}_D(A) \le \mathrm{maxdeg}_V(A) \cdot \mathrm{td}_I(A).$$

The proof idea is to investigate the definition of the incidence treedepth of A, which essentially boils down to recursively eliminating either a row, or a column, or decomposing a block-decomposable matrix into its blocks. Then, say for the second inequality above, eliminating a column can be replaced by eliminating all the at most $\mathrm{maxdeg}_V(A)$ rows that contain non-zero entries in this column.

It follows that if we bound either $\mathrm{maxdeg}_V(A)$ or $\mathrm{maxdeg}_C(A)$, that is, formally set $\mathrm{maxdeg}(A) = \min\{\mathrm{maxdeg}_V(A), \mathrm{maxdeg}_C(A)\}$, then the linear IP with such a solution set is solvable in time $f(\mathrm{maxdeg}(A), \|A\|_\infty) \cdot n^{O(1)} \cdot L$ thanks to results of Koutecký et al. [14]. Consequently, the use of high-degree constraints and variables in the proof of Theorem 1 is unavoidable.

4 Conclusions

We have shown that, unlike the primal and the dual treedepth, the incidence treedepth of a constraint matrix of (SIP) does not (together with the largest coefficient) provide a way to tractability. This shows our current understanding of the structure of the incidence Gaifman graph is not sufficient. Thus, the effect on tractability of some other "classical" graph parameters shall be investigated. For example we have some preliminary evidences that

– the vertex cover number of the incidence Gaifman graph together with the largest coefficient yields a tractable case and
– the graph in our reduction (Theorem 1) may admit a treecut decomposition of constant width.

We are going to investigate the two above claims in detail in the full version of this paper. Last but not least, all of the above suggest some open questions. Namely, whether ILP parameterized by the largest coefficient and treewidth and the maximum degree of the incidence Gaifman graph is in FPT or not. Furthermore, one may also ask about parameterization by the largest coefficient and the feedback vertex number of the incidence Gaifman graph.

Appendix

Proof of Lemma 4. We prove only the second inequality, as the first one is symmetric. The proof is by induction with respect to the total number of rows and columns of the matrix A. The base of the induction, when A has one row and one column, is trivial, so we proceed to the induction step.

Observe that $G_I(A)$ is disconnected if and only if $G_D(A)$ is disconnected if and only if A is a block-decomposable matrix. Moreover, the incidence treedepth of A is the maximum incidence treedepth among the blocks of A, and the same also holds for the dual treedepth. Hence, in this case we may apply the induction hypothesis to every block of A and combine the results in a straightforward manner.

Assume then that $G_I(A)$ is connected. Then

$$\mathrm{td}(G_I(A)) = 1 + \min_{v \in V(G_I(A))} \mathrm{td}(G_I(A) - v).$$

Let v be the vertex for which the minimum on the right hand side is attained. We consider two cases: either v is a row of A or a column of A.

Suppose first that v is a row of A. Then we have

$$\begin{aligned}\mathrm{td}(G_D(A)) &\leq 1 + \mathrm{td}(G_D(A) - v)\\ &\leq 1 + \mathrm{maxdeg}_V(A) \cdot \mathrm{td}(G_I(A) - v)\\ &= 1 + \mathrm{maxdeg}_V(A) \cdot (\mathrm{td}(G_I(A)) - 1)\\ &\leq \mathrm{maxdeg}_V(A) \cdot \mathrm{td}(G_I(A))\end{aligned}$$

as required, where the second inequality follows from applying the induction assumption to A with the row v removed.

Finally, suppose that v is a column of A. Let X be the set of rows of A that contain non-zero entries in column v; then $|X| \leq \mathrm{maxdeg}_V(A)$ and X is non-empty, because $G_I(A)$ is connected. If we denote by $A - v$ the matrix obtained from A by removing column v, then we have

$$\begin{aligned}\mathrm{td}(G_D(A)) &\leq |X| + \mathrm{td}(G_D(A) - X)\\ &\leq \mathrm{maxdeg}_V(A) + \mathrm{td}(G_D(A - v))\\ &\leq \mathrm{maxdeg}_V(A) + \mathrm{maxdeg}_V(A) \cdot \mathrm{td}(G_I(A - v))\\ &\leq \mathrm{maxdeg}_V(A) \cdot \mathrm{td}(G_I(A)),\end{aligned}$$

as required. Here, in the second inequality we used the fact that $G_D(A) - X$ is a subgraph of $G_D(A - v)$, while in the third inequality we used the induction assumption for the matrix $A - v$. □

References

1. Altmanová, K., Knop, D., Koutecký, M.: Evaluating and tuning n-fold integer programming. In: D'Angelo, G. (ed.) 17th International Symposium on Experimental Algorithms, SEA 2018, L'Aquila, Italy, 27–29 June 2018. LIPIcs, vol. 103, pp. 10:1–10:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2018). https://doi.org/10.4230/LIPIcs.SEA.2018.10
2. Chatzigiannakis, I., Kaklamanis, C., Marx, D., Sannella, D. (eds.): 45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, Prague, Czech Republic, 9–13 July 2018. LIPIcs, vol. 107. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2018). http://www.dagstuhl.de/dagpub/978-3-95977-076-7

3. Chen, L., Xu, L., Shi, W.: On the graver basis of block-structured integer programming. CoRR abs/1805.03741 (2018). http://arxiv.org/abs/1805.03741
4. De Loera, J.A., Hemmecke, R., Köppe, M.: Algebraic and Geometric Ideas in the Theory of Discrete Optimization. MOS-SIAM Series on Optimization, vol. 14. SIAM (2013). https://doi.org/10.1137/1.9781611972443
5. Dechter, R.: Chapter 7 - tractable structures for constraint satisfaction problems. In: Rossi, F., van Beek, P., Walsh, T. (eds.) Handbook of Constraint Programming, Foundations of Artificial Intelligence, vol. 2, pp. 209–244. Elsevier (2006). https://doi.org/10.1016/S1574-6526(06)80011-8
6. Eiben, E., Ganian, R., Knop, D., Ordyniak, S.: Unary integer linear programming with structural restrictions. In: Lang, J. (ed.) Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden, 13–19 July 2018, pp. 1284–1290. ijcai.org (2018). https://doi.org/10.24963/ijcai.2018/179
7. Eisenbrand, F., Hunkenschröder, C., Klein, K.: Faster algorithms for integer programs with block structure. In: Chatzigiannakis et al. [2], pp. 49:1–49:13. https://doi.org/10.4230/LIPIcs.ICALP.2018.49
8. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, New York (1979)
9. Hemmecke, R., Köppe, M., Weismantel, R.: A polynomial-time algorithm for optimizing over N-fold 4-block decomposable integer programs. In: Eisenbrand, F., Shepherd, F.B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 219–229. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13036-6_17
10. Hemmecke, R., Onn, S., Romanchuk, L.: N-fold integer programming in cubic time. Math. Program. **137**(1–2), 325–341 (2013). https://doi.org/10.1007/978-3-642-13036-6_17
11. Hemmecke, R., Schultz, R.: Decomposition of test sets in stochastic integer programming. Math. Program. **94**(2), 323–341 (2003). https://doi.org/10.1007/s10107-002-0322-1
12. Jansen, K., Lassota, A., Rohwedder, L.: Near-linear time algorithm for n-fold ILPs via color coding. CoRR abs/1811.00950 (2018)
13. Klein, K.: About the complexity of two-stage stochastic IPs. CoRR abs/1901.01135 (2019). http://arxiv.org/abs/1901.01135
14. Koutecký, M., Levin, A., Onn, S.: A parameterized strongly polynomial algorithm for block structured integer programs. In: Chatzigiannakis et al. [2], pp. 85:1–85:14. https://doi.org/10.4230/LIPIcs.ICALP.2018.85
15. Lenstra Jr., H.W.: Integer programming with a fixed number of variables. Math. Oper. Res. **8**(4), 538–548 (1983). https://doi.org/10.1287/moor.8.4.538
16. Nešetřil, J., Ossona de Mendez, P.: Sparsity - Graphs, Structures, and Algorithms. AC, vol. 28. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27875-4
17. Onn, S.: Nonlinear Discrete Optimization: An Algorithmic Theory (Zurich Lectures in Advanced Mathematics). European Mathematical Society Publishing House (2010)