[EURO Journal on Computational Optimization 9 (2021) 100020](https://doi.org/10.1016/j.ejco.2021.100020)

Contents lists available at [ScienceDirect](http://www.ScienceDirect.com/)

EURO Journal on Computational Optimization

journal homepage: [www.elsevier.com/locate/ejco](http://www.elsevier.com/locate/ejco)

Presolving linear bilevel optimization problems

Thomas Kleinert[a](#_bookmark0),[b](#_bookmark1), Julian Manns [a](#_bookmark0), Martin Schmidt[c](#_bookmark2),[∗](#_bookmark3), Dieter Weninger [a](#_bookmark0)

a *Friedrich-Alexander-Universität Erlangen-Nürnberg, Discrete Optimization, Cauerstr. 11, Erlangen 91058, Germany*

b *Energie Campus Nürnberg, Fürther Str. 250, Nürnberg 90429, Germany*

c *Trier University, Department of Mathematics, Universitätsring 15, Trier 54296, Germany*

a r t i c l e i n f o a b s t r a c t

*2010 MSC:*

90-08

90C11

90C30

90C26

90C46

*Keywords:*

Linear bilevel optimization Presolve

Computational analysis

Linear bilevel optimization problems are known to be strongly NP-hard and the computational techniques to solve these problems are often motivated by techniques from single-level mixed-integer optimization. Thus, during the last years and decades many branch-and-bound methods, cutting planes, or heuristics have been proposed. On the other hand, there is almost no literature on presolving linear bilevel problems although presolve is a very important ingredient in state-of-the-art mixed-integer optimization solvers. In this paper, we carry over standard presolve techniques from single-level optimization to bilevel problems and show that this needs to be done with great caution since a naive application of well-known techniques does often not lead to correctly presolved bilevel models. Our numerical study shows that presolve can also be very beneficial for bilevel problems but also highlights that these methods have a more heterogeneous effect on the solution process compared to what is known from single-level optimization. As a side result, our numerical experiments reveal that there is an urgent need for better and more heterogeneous test instance libraries to further propel the field of computational bilevel optimization.

# Introduction

In the last years and decades, bilevel optimization has received in- creasing attention. In particular, the progress of computational mixed- integer bilevel optimization has been significant. This progress resem- bles the development in mixed-integer single-level optimization for two reasons. First, many algorithms for (mixed-integer) linear bilevel prob- lems rely on highly-developed mixed-integer solvers as the working horse. Second, game-changing developments from mixed-integer opti- mization are also used in bilevel optimization such as tailored branch- [and-bound methods (Bard and Moore, 1990; Moore and Bard 1990;](#_bookmark41) [DeNegre,](#_bookmark61) [2011), cutting planes (](#_bookmark41)[Fischetti](#_bookmark57) [et al., 2018a; Tahernejad et al.,](#_bookmark41) [2020](#_bookmark54)[;](#_bookmark61) [Audet](#_bookmark42) [et al., 2007;](#_bookmark61) [Kleinert](#_bookmark69) [et al., 2021), and heuristics (DeNegre,](#_bookmark61) [2011; Fischetti et al., 2018b; Kleinert and Schmidt, 2020)—to name](#_bookmark54) only a few references for each of the fields. These developments allow to tackle bilevel problems of significant size today that have been far out of reach one or two decades ago. Nevertheless, one main driver of the success of mixed-integer solvers has hardly seen any attention in the bilevel literature so far: *presolve*.

As usual, presolve for bilevel problems should be applied, in general, to reduce the size of the problem in terms of variables and constraints as well as to obtain a tighter formulation of the problem. For bilevel problems, however, it is particularly important to reduce the number of

follower constraints since classic reformulations such as the one using the Karush–Kuhn–Tucker (KKT) conditions lead to harder problems if the number of follower constraints is large.

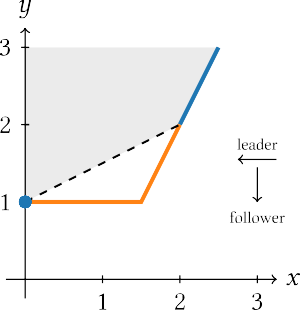
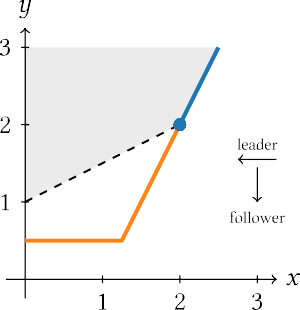
To the best of our knowledge, [Fischetti et al. (2017)](#_bookmark55) is the only work that computationally analyzes a general-purpose presolve method for bilevel problems. The authors show that applying duality fixing in the lower level of a mixed-integer bilevel problem can be very effective. We will later numerically confirm this result for linear bilevel problems as well. One reason for the sparse literature on bilevel presolve might be the inherent nonconvex structure of bilevel problems. This structure yields some surprising properties that render presolving bilevel prob- lems a very challenging task. In [Macal and Hurter (1997)](#_bookmark50), it is discussed that bilevel problems do not posses the independence of irrelevant con- straints (IIC) property known from single-level optimization. Essentially, the consequence is that adding inequalities to the lower-level problem of a bilevel problem that are inactive at the optimal bilevel solution may change the bilevel feasible set and, ultimately, the optimal solution as well. In [Dempe and Lohse (2011)](#_bookmark52), it is shown that under certain assump- tions on the bilevel problem, a global solution remains locally optimal when adding irrelevant constraints. The two results render cutting-plane algorithms, which subsequently develop the lower-level problem, infea- sible. This is in contrast to single-level optimization, where a solution of a relaxed problem (in which, e.g., some constraints are left out) that

∗ Corresponding author.

*E-mail addresses:* [thomas.kleinert@fau.de](mailto:thomas.kleinert@fau.de) (T. Kleinert), [julian.manns@fau.de](mailto:julian.manns@fau.de) (J. Manns), [martin.schmidt@uni-trier.de](mailto:martin.schmidt@uni-trier.de) (M. Schmidt), [dieter.weninger@fau.de](mailto:dieter.weninger@fau.de) (D. Weninger).

<https://doi.org/10.1016/j.ejco.2021.100020> Received 19 July 2021; Accepted 7 November 2021

2192-4406/© 2021 The Author(s). Published by Elsevier Ltd on behalf of Association of European Operational Research Societies (EURO). This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

where S(*𝑥*) denotes the set of optimal solutions or the rational reaction set of the so-called follower (or lower-level) problem

min

*𝑦*∈ℝ

*𝑓 ⊤𝑦* s*.*t*. 𝐷𝑦* ≥ *𝑏* − *𝐶𝑥, 𝑦𝐹* ≤ *𝑦* ≤ *𝑦̄𝐹 ,* (2)

**Fig. 1.** Feasible set and optimal solution without (left) and with (right) bound tightening applied. The dashed black line corresponds to the upper-level cou- pling constraint, the orange lines correspond to the lower-level constraints, the gray area is the joint feasible set w.r.t. the upper- and lower-level constraints, and the blue lines denote the bilevel feasible set with the dot marking the opti-

with *𝑐* ∈ ℝ*𝑛*, *𝑑, 𝑓* ∈ ℝ*𝑚*, *𝐴* ∈ ℝ*𝑘*×*𝑛*, *𝐵* ∈ ℝ*𝑘*×*𝑚*, *𝑎* ∈ ℝ*𝑘*, *𝐶* ∈ ℝ𝓁×*𝑛*, *𝐷* ∈ ℝ𝓁×*𝑚*, and *𝑏* ∈ ℝ𝓁 . Further, we denote variable bounds by *𝑥𝐿, 𝑥̄𝐿* ∈ ℝ*𝑛* and *𝑦𝐿, 𝑦̄𝐿* ∈ ℝ*𝑚* as well as *𝑦𝐹 , 𝑦̄𝐹* ∈ ℝ*𝑚*. Note that we explicitly distin- guish leader and follower bounds on the follower variables *𝑦*. We exploit

these bounds later in various presolve techniques.

lution, i.e., in case the set of optimal follower solutions S(*𝑥*) is not a The formulation [(1)](#_bookmark9) establishes the optimistic (or cooperative) so- singleton, the reaction *𝑦* ∈ S(*𝑥*) is chosen in favor of the objective func-

[tion of the so-called leader (or upper-level) problem](#_bookmark53) [(1)](#_bookmark9)[; see, e.g., Dempe (2002). We denote the shared constraint set and its projection onto the](#_bookmark53) decision space of the leader by

{

mal solution. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

is also feasible for the omitted constraints is the global optimum. Addi- tionally, the results also indicate that applying common mixed-integer presolve techniques such as, e.g., bound strengthening, to bilevel prob-

Ω ∶=

and

*𝐿 𝑥* ∈ Ω *,*

(*𝑥, 𝑦*) ∶ *𝐴𝑥* + *𝐵𝑦* ≥ *𝑎,*

}

*𝐶 𝑥* + *𝐷𝑦* ≥ *𝑏, 𝑥𝐿* ≤ *𝑥* ≤ *𝑥̄𝐿, 𝑦𝐿* ≤ *𝑦* ≤ *𝑦̄𝐿, 𝑦𝐹* ≤ *𝑦* ≤ *𝑦̄𝐹*

lems may not be without obstacles. To shed some light on this, let us consider the linear bilevel problem

Ω ∶= { ∶ ∃*𝑦* with (*𝑥, 𝑦*) }

min

*𝑥,𝑦*∈ℝ

*𝑥* s*.*t*. 𝑦* ≥ 0*.*5*𝑥* + 1*, 𝑥* ≥ 0*, 𝑦* ∈ arg min {*𝑦̄* ∶ *𝑦̄* ≥ 2*𝑥* − 2*, 𝑦̄* ≥ 0*.*5}*,*

*𝑦̄*∈ℝ

respectively. Finally, the bilevel feasible set, often also called the in-

ducible region, is denoted by

with optimal solution (2,2); see [Fig. 1](#_bookmark4) (left).

When strengthening the bound *𝑦̄* ≥ 0*.*5 in the lower-level problem

T ∶=

{ }

(*𝑥, 𝑦*) ∶ *𝐴𝑥* + *𝐵𝑦* ≥ *𝑎, 𝑥𝐿* ≤ *𝑥* ≤ *𝑥̄𝐿, 𝑦𝐿* ≤ *𝑦* ≤ *𝑦̄𝐿, 𝑦* ∈ S(*𝑥*) *.*

using the constraint *𝑦* ≥ 0*.*5*𝑥* + 1 of the upper-level problem, one finds that the minimum value of 0*.*5*𝑥* + 1 is 1 due to *𝑥* ≥ 0, which increases the bound of *𝑦̄* to *𝑦̄* ≥ 1. This yields the problem

It is well known that the set {(*𝑥, 𝑦*) ∶ *𝑥* ∈ ℝ*𝑛, 𝑦* ∈ S(*𝑥*)} is nonconvex

as well as that the bilevel feasible set T is nonconvex and, in general,

disconnected; see, e.g., [Bard (1998)](#_bookmark43) or [Dempe (2002)](#_bookmark53). This is also il-

min

*𝑥,𝑦*∈ℝ

*𝑥* s*.*t*. 𝑦* ≥ 0*.*5*𝑥* + 1*, 𝑥* ≥ 0*, 𝑦* ∈ arg min{*𝑦̄* ∶ *𝑦̄* ≥ 2*𝑥* − 2*, 𝑦̄* ≥ 1}*,*

*𝑦̄*∈ℝ

lustrated in [Fig. 1](#_bookmark4) (right). Consequently, even linear bilevel problems [are intrinsically nonconvex and nonsmooth. It is shown in Hansen et al.](#_bookmark66)

having the optimal solution (0*,* 1) ≠ (2*,* 2); see [Fig. 1](#_bookmark4) (right). See also the

thesis [Manns (2020)](#_bookmark52) for further examples. It can thus not be expected

that presolve techniques known from single-level mixed-integer linear optimization can be applied directly to bilevel problems.

In this paper we formally and computationally analyze presolve methods for the easiest variant of bilevel problems, i.e., problems with a linear upper and lower level. To this end, we mainly carry over classic presolve techniques from mixed-integer optimization to the bilevel set- ting, which has—to the best of our knowledge—not been done before

[(1992)](#_bookmark64) [that linear bilevel problems are strongly NP-hard and, in Vicente](#_bookmark66) [et al. (1994), that even checking local optimality of a given point is NP-](#_bookmark64) hard.

These unpleasant properties of linear bilevel problems are well known and it is thus clear that one faces different challenging aspects when solving bilevel problems. However, it is still surprising that ap- plying classic presolve techniques like, e.g., bound tightening, fails for bilevel problems; see [Fig. 1](#_bookmark4). We highlight that it is in particular invalid to exploit the so-called high-point relaxation

in the literature with the only exception being a single technique dis- cussed in [Fischetti et al. (2017)](#_bookmark55). We introduce the relevant notation and

min

*𝑥,𝑦*

*𝑐⊤𝑥* + *𝑑⊤𝑦* s*.*t*.* (*𝑥, 𝑦*) ∈ Ω (3)

theory in [Section 2](#_bookmark8). We then introduce several presolve methods for lin- ear bilevel problems in [Section 3](#_bookmark14) by carrying over classic presolve ideas from (single-level) linear and mixed-integer optimization to the field of bilevel problems. Afterward, we evaluate these methods in a computa- tional study in [Section 4](#_bookmark25). These numerical experiments reveal two main insights. First, we observe that presolving linear bilevel problems can be very beneficial for some instances whereas it is rather harming for the solution process of other instances. Second, although we have a test set of more than 2500 instances, only very few of them are affected by some of the presolve methods discussed in this paper. We conclude, thus, that there is an urgent need for more realistic and more heterogeneous test instance libraries to further propel the field of computational bilevel optimization; see [Section 5](#_bookmark40).

# Notation and theoretical background

In this paper, we consider linear bilevel problems of the form

to presolve the original bilevel problem [(1)](#_bookmark9). As briefly discussed in the introduction, this is related to the independence of irrelevant constraints property. For single-level problems, this property states that adding any constraint that is satisfied at the global optimal solution does not change the global optimum. In [Macal and Hurter (1997)](#_bookmark50), the IIC property for linear bilevel problems of the form [(1)](#_bookmark9) is defined as follows.

**Definition 1.** Let Ω∗ be the set of optimal solutions of a linear bilevel problem *𝑃* of the form [(1)](#_bookmark9). Further, let *𝑃̃* ∶= *𝑃* (*𝑢, 𝑣, 𝑤*) be the modi- fied problem, in which the inequality *𝑢⊤𝑥* + *𝑣⊤𝑦* ≥ *𝑤* is added to the follower problem of *𝑃* and let Ω*̃* ∗ be its set of optimal solutions. *𝑃* is called *independent of irrelevant constraints*, if for any (*𝑢, 𝑣, 𝑤*) ∈ ℝ*𝑛*+*𝑚*+1 with *𝑢⊤𝑥*∗ + *𝑣⊤𝑦*∗ ≥ *𝑤* it holds (*𝑥*∗*, 𝑦*∗) ∈ Ω*̃* ∗ for every (*𝑥*∗*, 𝑦*∗) ∈ Ω∗.

It is shown in [Macal and Hurter (1997)](#_bookmark50) that bilevel problems can only possess the IIC property if the solution of the high-point relaxation [(3)](#_bookmark6) is also a solution of the bilevel problem [(1)](#_bookmark9). Consequently, most practical bilevel problems for which the objectives of the leader and the follower

min

*𝑥*∈ℝ*𝑛 ,𝑦*∈ℝ*𝑚*

*𝑐⊤𝑥* + *𝑑⊤𝑦*

are not aligned, lack the IIC property.

s*.*t*. 𝐴𝑥* + *𝐵𝑦* ≥ *𝑎,*

*𝑥𝐿* ≤ *𝑥* ≤ *𝑥̄𝐿, 𝑦𝐿* ≤ *𝑦* ≤ *𝑦̄𝐿,*

*𝑦* ∈ S(*𝑥*)*,*

(1)

On the other hand, [Definition 1](#_bookmark7) of the IIC property for bilevel prob- lems is not entirely in line with the IIC property known for single-level problems. In order to see this, we consider the equivalent single-level

reformulation this computation of tighter bounds is computationally costly. For more

min

*𝑥,𝑦*

*𝑐⊤𝑥* + *𝑑⊤𝑦* (4a)

details on this approach applied to single-level problems we refer to [Gleixner et al. (2017)](#_bookmark65). In practice, there exist different approaches

s*.*t*.* (*𝑥, 𝑦*) ∈ Ω*,* (4b)

*𝑓 ⊤𝑦* ≤ *𝜑*(*𝑥*)*,* (4c)

of Problem [(1)](#_bookmark9), which makes use of the optimal-value function

to compute tight bounds eﬃciently for single-level optimization prob- lems and, usually, there is a trade-off between the tightness of the de- rived bounds and computation times. In the simplest case, one consid- ers each constraint separately and tries to derive tighter bounds for the

*𝜑*(*𝑥*) ∶= min

*𝑦*∈ℝ*𝑚*

{ }

*𝑓 ⊤𝑦* ∶ *𝐷𝑦* ≥ *𝑏* − *𝐶𝑥, 𝑦𝐹* ≤ *𝑦* ≤ *𝑦̄𝐹*

[variables occurring in it, which is doable in linear time; see Achterberg et al. (2020) and](#_bookmark45) [Martin](#_bookmark54) [(2001). Approaches that consider more than one](#_bookmark45) [constraint simultaneously, see, e.g.,](#_bookmark51) [Achterberg et al. (2014)](#_bookmark46) [and Chen](#_bookmark51)

lem by (*𝑥*∗*, 𝑦*∗). It is clear that any inequality valid for (*𝑥*∗*, 𝑦*∗) can safely of the follower problem. We denote the optimal solution of this prob-

be added to Problem [(4)](#_bookmark10), i.e., the single-level problem [(4)](#_bookmark10) possesses the (single-level) IIC property. Adding such a constraint to Problem [(4)](#_bookmark10) how- ever corresponds to adding an irrelevant constraint to the leader prob- lem of the bilevel problem [(1)](#_bookmark9). On the contrary, adding a constraint that is valid for the bilevel optimal solution to the follower problem of

[(1)](#_bookmark9) *may* change the rational reaction set S(*𝑥*), respectively, the optimal-

value function *𝜑*(*𝑥*). In the point of view of Problem [(4)](#_bookmark10), adding such

to the set Ω but *may* also change the right-hand side of the existing a constraint to the follower problem thus not only adds this constraint

Constraint [(4c)](#_bookmark11). In this case, the concept of the (single-level) IIC prop- erty is not applicable. Detecting such constraints that do not affect Con- straint [(4c)](#_bookmark11) is a delicate task and will play a crucial role in the presolve of linear bilevel problems.

In general, the goal of presolving optimization problems is to trans-

[et al. (2020), usually yield better bounds than approaches that consider](#_bookmark51)

only one constraint. However, an increased running time is to be ex- pected. Naturally, this procedure can be continued until all constraints are considered, which results in the best possible bounds. Obviously, the computational effort involved is also the largest. In this work, we focus on the analysis of the effectiveness of bound strengthening by using the best available bounds and, thus, we refrain from using other methods as to solve [(5)](#_bookmark16) to compute these bounds.

The strengthening of follower bounds *𝑦𝐹* and *𝑦̄𝐹* located in the fol-

lower problem is not that straightforward. We already illustrated in the

the shared constraint set Ω may result in wrong “solutions” of Prob- introduction that a naive strengthening of follower bounds that exploits

lem [(1)](#_bookmark9). The reason is that the shared constraint set also contains the leader constraints. This suggests to tighten follower bounds by disre- garding the leader constraints, i.e., by solving

form the given problem into an easier-to-solve problem that has the following properties:

min

*𝑥,𝑦*

and

*𝑦𝑖* s*.*t*. 𝐶𝑥* + *𝐷𝑦* ≥ *𝑏, 𝑦𝐹* ≤ *𝑦* ≤ *𝑦̄𝐹* (6)

1. It is infeasible (unbounded) if and only if the original problem is infeasible (unbounded).

max

*𝑥,𝑦*

*𝑦𝑖*

s*.*t*. 𝐶𝑥* + *𝐷𝑦* ≥ *𝑏, 𝑦𝐹* ≤ *𝑦* ≤ *𝑦̄𝐹 ,* (7)

1. Every feasible solution of the presolved problem can be transformed into a feasible solution of the original problem.
2. The objective value of every optimal solution of the presolved prob- lem matches (at least after possibly required post-processing steps) the optimal objective value of the original problem.

This gives rise to two different presolve strategies. First, we can mod- ify Problem [(1)](#_bookmark9) in a way that leaves the bilevel feasible set T unchanged, [i.e., we can apply a feasibility-based presolve; see, e.g., Belotti et al. (2010) and](#_bookmark48) [Belotti](#_bookmark44) [et al. (2012) for single-level problems. Second, we](#_bookmark48) can apply an optimality-based presolve by modifying the bilevel feasi- ble set T in a way to ensure that at least one bilevel optimal solution [remains feasible for the reduced problem; see, e.g., Ryoo and Sahinidis (1996) for single-level problems or the removal of dominated columns](#_bookmark60) in mixed-integer programming ([Gamrath et al. 2015](#_bookmark62)). In the following section, we introduce several such presolve methods. We also evaluate these methods in a computational study in [Section 4](#_bookmark25).

# Presolve methods

In this section, we present presolve methods for linear bilevel prob-

for *𝑖* = 1*,* … *, 𝑚*. These problems can be further tightened by considering

ables *𝑥*, i.e., leader constraints with *𝐵𝑖*⋅ = 0 or leader bounds *𝑥𝐿* ≤ *𝑥* ≤ follower-independent leader constraints that only depend on leader vari-

*𝑥̄𝐿*. We show in the following theorem that follower-independent leader

constraints can be moved to the follower problem without changing the bilevel feasible set T .

**Theorem 1.** *Consider a leader constraint 𝑢⊤𝑥* ≥ *𝑤 with 𝑢* ∈ ℝ*𝑛 and 𝑤* ∈ ℝ

*Problem* [*(1)*](#_bookmark9) *by moving the constraint 𝑢⊤𝑥* ≥ *𝑤 to the follower problem. We of the bilevel problem* [*(1)*](#_bookmark9)*. Further, consider the bilevel problem obtained from denote the feasible set of the latter problem by* T *𝑀 . Then,* T = T *𝑀 holds.*

**Proof.** The shared constraint set for both problems is given by

Ω*̃* = Ω∩ {(*𝑥, 𝑦*) ∶ *𝑢⊤𝑥* ≥ *𝑤*}*.*

the constraint is moved to the follower, by S *𝑀* . We then have Further, we denote the rational reaction set of Problem [(1)](#_bookmark9), for which

T *𝑀* = Ω*̃* ∩ {(*𝑥, 𝑦*) ∶ *𝑥* ∈ ℝ*𝑛, 𝑦* ∈ S *𝑀* (*𝑥*)}*.*

Thus, for any point (*𝑥, 𝑦*) in T *𝑀* it holds *𝑢⊤𝑥* ≥ *𝑤*. Further, for *𝑥* ∈ ℝ*𝑛* it

holds

lems. { *⊤*

* 1. *Bound strengthening*

S *𝑀* (*𝑥*) =

S(*𝑥*)*,* if *𝑢 𝑥* ≥ *𝑤,*

∅*,* if *𝑢⊤𝑥 < 𝑤.*

A well-known presolve technique for single-level optimization is bound strengthening ([Savelsbergh, 1994; Achterberg et al., 2020](#_bookmark63)). This approach tries to strengthen variable bounds by domain propagation without changing the set of feasible solutions.

Type [(1)](#_bookmark9). Leader bounds *𝑥𝐿* and *𝑥̄𝐿* can be tightened by exploiting the We now consider bound strengthening for bilevel problems of shared constraint set Ω and, thus, by solving

*𝑖 𝑖*

min { } { }

Consequently,

T *𝑀* = Ω*̃* ∩ {(*𝑥, 𝑦*) ∶ *𝑥* ∈ ℝ*𝑛, 𝑦* ∈ S *𝑀* (*𝑥*)}

= Ω*̃* ∩ {(*𝑥, 𝑦*) ∶ *𝑥* ∈ ℝ*𝑛, 𝑦* ∈ S(*𝑥*)} = T *.* □

This theorem can be exploited to tighten the Problems [(6)](#_bookmark12) and [(7)](#_bookmark13).

to Problem [(6)](#_bookmark12) and [(7)](#_bookmark13). In particular, this includes the bounds *𝑥𝐿* ≤ *𝑥* ≤ **Observation 1.** We can add all follower-independent leader constraints

*𝑥̄𝐿* on the leader variables.

*𝑥𝑖* ∶ (*𝑥, 𝑦*) ∈ Ω and max *𝑥𝑖* ∶ (*𝑥, 𝑦*) ∈ Ω *,* (5)

respectively, for *𝑖* = 1*,* … *, 𝑛*. Tight follower bounds *𝑦𝐿* and *𝑦̄𝐿* located

in the leader problem can be derived analogously. We emphasize that

Let us also note that we can handle tightened bounds also slightly different. In practice, linear bilevel problems [(1)](#_bookmark9) are mostly solved by a

single-level reformulation that replaces the follower problem with its KKT conditions or with the strong-duality condition. In this view, it makes sense to move follower bounds to the leader instead of tightening them. This results in a smaller single-level reformulation. For instance, in the case that the KKT reformulation is used, moving bounds to the leader results in fewer KKT complementarity conditions. However, it is well known that moving constraints between the leader and the fol- lower problem is not without obstacles and, in general, it may change the optimal solution. Nevertheless, we make the following observation.

**Observation 2.** Consider a follower variable *𝑦* with a lower bound *𝑦𝐹*

From now on, we consider the bilevel setting again. A straightfor- ward idea is to apply the approach stated above only to the parametric

follower problem [(2)](#_bookmark5). In other words, we have two follower variables *𝑦𝑖*

and *𝑦𝑗* for some *𝑖* ≠ *𝑗*, *𝑖, 𝑗* ∈ {1*,* … *, 𝑚*}, with parallel columns in the fol-

necessarily be parallel, i.e., we might have *𝐵*⋅*𝑖* ≠ *𝜇𝐵*⋅*𝑗* . In this setting, we lower problem. However, the corresponding leader columns need not

This removes the two variables *𝑦𝑖* and *𝑦𝑗* from the follower problem can crush the two variables in the follower problem according to [(8)](#_bookmark18).

such that they only appear in the leader problem. In order to still re- spect the (crushed) optimal follower solution in the leader problem, we

*𝑖*

that can be tightened, i.e., the objective value *𝑦*∗

*𝑖*

*𝑖*

of Problem [(6)](#_bookmark12) is strictly

add the constraint *𝑦*new = *𝑦𝑗* + *𝜇𝑦𝑖* to the leader problem. This leaves the

bilevel feasible set, after uncrushing, unchanged.

larger than *𝑦𝐹* . Instead of tightening the bound, we can move the origi-

*𝑖*

nal bound *𝑦𝐹* from the follower to the leader problem. Since the bound is

*𝑖*

tional reaction set S(*𝑥*) of the follower. The same holds for upper bounds. implied by the follower constraints anyway, this does not change the ra-

Note that this is in contrast to general lower-level constraints (i.e., constraints that are more involved than simple bounds for the lower- level variables), which can be added to the leader without problems but not moved from the follower to the leader without altering the bilevel- feasible set T . We finally note that one can also simply remove the (any-

way redundant) bound *𝑦𝐹* from the follower problem without adding

A similar approach can be applied if two follower columns are par- allel in both the leader and the follower problem, i.e.,

*𝑑𝑖* = *𝜇𝑑𝑗 , 𝑓𝑖* = *𝜇𝑓𝑗 , 𝐵*⋅*𝑖* = *𝜇𝐵*⋅*𝑗 , 𝐷*⋅*𝑖* = *𝜇𝐷*⋅*𝑗*

for *𝑖* ≠ *𝑗*, *𝑖, 𝑗* ∈ {1*,* … *, 𝑚*}, and some *𝜇* ≠ 0. We can now crush *𝑦𝑖* and *𝑦𝑗*

into *𝑦*new . For the follower, we proceed according to [(8)](#_bookmark18) and [(9)](#_bookmark19), For the

leader we apply

*𝑑*new = *𝑑𝑗 , 𝐵*⋅new = *𝐵*⋅*𝑗*

and

{ *𝐿 𝐿* {

*𝑖* ∗

*𝑦* + *𝜇𝑦*

for *𝜇 >* 0*,*

*𝑦̄𝐿* + *𝜇𝑦̄𝐿*

for *𝜇 >* 0*,*

it to the leader problem, or one can add the updated bound *𝑦*

to the

*𝑦𝐿* = *𝑗 𝑖*

*𝑦̄𝐿* = *𝑗 𝑖*

*𝑖* new

*𝑦𝐿* + *𝜇𝑦̄𝐿*

for *𝜇 <* 0*,*

new

*𝑦̄𝐿* + *𝜇𝑦𝐿*

for *𝜇 <* 0*.*

leader problem. From single-level optimization it is known to be un-

*𝑗 𝑖*

*𝑗 𝑖*

clear, which strategy “helps” the solver and which one is rather harmful to the solution process. Thus, these different strategies need to be eval- uated numerically.

* 1. *Parallel columns*

We first briefly explain the presolve of parallel columns in the single-level context; see also, e.g., [Andersen and Andersen (1995)](#_bookmark47) and [Achterberg et al. (2020)](#_bookmark45). In [Achterberg et al. (2020)](#_bookmark45) it is both shown that parallel columns are common in public and commercial single-level problems and that their treatment helps for solving the instances more eﬃciently. To this end, consider the single-level follower problem [(2)](#_bookmark5) for

a fixed leader decision *𝑥* = *𝑥̄* as well as two follower variables *𝑦𝑖* and *𝑦𝑗*

for some *𝑖* ≠ *𝑗*, *𝑖, 𝑗* ∈ {1*,* … *, 𝑚*}. The two columns *𝐷*⋅*𝑖* and *𝐷*⋅*𝑗* are called

parallel if *𝐷*⋅*𝑖* = *𝜇𝐷*⋅*𝑗* holds for some *𝜇* ≠ 0. An algorithm for determin-

As above, we argue that an uncrushed solution is feasible for the follower problem [(2)](#_bookmark5) if and only if the crushed solution is feasible. Also, following the discussion for the single-level case above, the crushed and uncrushed solutions yield the same objective value for the follower problem. Fur- thermore, with the same arguments as for the follower-only case, the uncrushed solution is feasible for the leader if and only if the crushed solution is feasible for the leader. Consequently, the uncrushed solution is bilevel feasible if and only if the crushed solution is bilevel feasible. Finally, the crushed and uncrushed solution have the same leader objec- tive value, which means that the uncrushed solution is bilevel optimal if and only if the crushed solution is bilevel optimal.

We now consider two parallel leader columns in Problem [(1)](#_bookmark9), i.e.,

*𝑐𝑖* = *𝜇𝑐𝑗 , 𝐴*⋅*𝑖* = *𝜇𝐴*⋅*𝑗 , 𝐶*⋅*𝑖* = *𝜇𝐶*⋅*𝑗*

for *𝑖* ≠ *𝑗*, *𝑖, 𝑗* ∈ {1*,* … *, 𝑛*}, and some *𝜇* ≠ 0. We can crush *𝑥𝑖* and *𝑥𝑗* into

ing parallel columns with runtime *𝑂*(∑ *𝑙* log *𝑙* ), where *𝑙* represents the

*𝑥* across all leader and follower constraints and the two objectives,

*𝑖 𝑖 𝑖 𝑖*

new

number of non-zeros in column *𝐷*⋅*𝑖* , is given in [Bixby and Wagner (1987)](#_bookmark49).

If, in addition, *𝑓𝑖* = *𝜇𝑓𝑗* holds, we can merge, or “crush”, the two

similarly to the parallel follower case. Now, let *𝑥*∗ be part of an optimal

solution of the presolved, i.e., crushed, problem. We uncrush this solu-

variables *𝑦𝑖* and *𝑦𝑗* into a new variable *𝑦*new by setting

tion by setting *𝑥*∗ = *𝑥*∗ + *𝜇𝑥*∗. This does not affect the right-hand sides

new *𝑗 𝑖*

*𝑦*new ∶= *𝑦𝑗* + *𝜇𝑦𝑖 .* (8)

ing the two original variables *𝑦𝑖* and *𝑦𝑗* and by instead adding the vari- Consequently, we obtain a presolved version of Problem [(2)](#_bookmark5) by remov- able *𝑦*new with

*𝑏* − *𝐶𝑥* of the follower constraints. In addition, the uncrushed solution

is feasible for the leader constraints if and only if the crushed solution

is feasible. Thus, the uncrushed solution is bilevel feasible if and only if the crushed solution is bilevel feasible for presolved bilevel problem.

Up to now, we only considered the case in which either both columns

*𝑓*new = *𝑓𝑗 , 𝐷*

⋅new

= *𝐷*

⋅*𝑗*

are leader columns or both columns are follower columns. Unfortu- nately, we cannot presolve a leader column that is parallel to a follower

as well as bounds

{*𝑦𝐹* + *𝜇𝑦𝐹*

*𝑦𝐹* = *𝑗 𝑖*

for *𝜇 >* 0*,*

{

*𝑦̄𝐹* + *𝜇𝑦̄𝐹*

*𝑦̄* = *𝑗 𝑖*

*𝐹*

for *𝜇 >* 0*,*

(9)

column, as the following example shows.

**Example 1.** Consider the linear bilevel problem

new

*𝑦𝐹* + *𝜇𝑦̄𝐹*

for *𝜇 <* 0*,*

new

*𝑦̄𝐹* + *𝜇𝑦𝐹*

for *𝜇 <* 0*.* { }

*𝑗 𝑖*

Now, let ∗

*𝑦*

be an optimal solution of the presolved problem. Then, we

*𝑗 𝑖*

min

*𝑥*∈ℝ*,𝑦*∈ℝ2

*𝑥* + *𝑦*1 s*.*t*.* 0 ≤ *𝑥* ≤ 1*, 𝑦* ∈ arg max

*𝑦̄*∈ℝ2

*𝑦̄*2 ∶ *𝑦̄*2 ≤ *𝑥* + *𝑦̄*1 *,* 0 ≤ *𝑦̄*1 ≤ 1 *,*

can post-process, or “uncrush”, the value *𝑦*∗

into values *𝑦*∗ and *𝑦*∗ by

which has the unique optimal solution *𝑥*∗ = 0, *𝑦*∗ = 1, and *𝑦*∗ = 1. Note

new

*𝑖 𝑗*

that the columns of *𝑥* and *𝑦*

1 2

are parallel with = 1

using [Equation (8)](#_bookmark18) and by paying attention to the bounds [(9)](#_bookmark19). It holds

1 *𝜇*

. If we crush them

*𝐷 𝑦*∗

= *𝐷* (*𝑦*∗ + *𝜇𝑦*∗) = *𝐷 𝑦*∗ + *𝜇𝐷 𝑦*∗ = *𝐷 𝑦*∗ + *𝐷 𝑦*∗

into a leader variable, we obtain

new new

⋅*𝑗 𝑗 𝑖*

⋅*𝑗 𝑗*

⋅*𝑗 𝑖*

⋅*𝑗 𝑗*

⋅*𝑖 𝑖*

min

*𝑥*new *,𝑦*2 ∈ℝ

*𝑥*new

s*.*t*.* 0 ≤ *𝑥*

new

≤ 2*, 𝑦*2

∈ arg max {*𝑦̄*

∶ *𝑦̄*2

≤ *𝑥*

}

new *,*

so that the uncrushed solution is feasible for the original problem [(2)](#_bookmark5) if

and only if *𝑦*∗ is feasible for the presolved problem. In addition, we have

which has the unique optimal solution *𝑥*∗

*𝑦̄*2 ∈ℝ

= 0 and *𝑦*∗ = 0. By uncrush-

2

∗ ∗ ∗ ∗

∗ ∗ ∗

ing *𝑥*∗

= *𝑥*∗ + *𝑦* , we obtain *𝑥*∗ = 0

new 2

= 0

*𝑓*new *𝑦*new = *𝑓𝑗* (*𝑦𝑗* + *𝜇𝑦𝑖* ) = *𝑓𝑗 𝑦𝑗* + *𝜇𝑓𝑗 𝑦𝑖* = *𝑓𝑗 𝑦𝑗* + *𝑓𝑖 𝑦𝑖 .*

new 1

and *𝑦*1

. Contrary, if we crush

the two variables into a follower variable, we obtain

Thus, the uncrushed solution is optimal for Problem [(2)](#_bookmark5) if and only if

min *𝑦*

s*.*t*.* (*𝑦*

{

*, 𝑦* ) ∈ arg max *𝑦̄*

∶ *𝑦̄*

≤ *𝑦̄*

*,* 0 ≤ *𝑦̄* ≤ 2}

∗ new

*𝑦*

is optimal for the presolved problem.

*𝑦*new *,𝑦*2 ∈ℝ

new

new 2

*𝑦̄*new *,𝑦̄*2 ∈ℝ 2 2

new

new *,*

**Table 1**

Implications of parallel rows depending on *𝜇* and the right-hand sides *𝑎̃𝑞* and *𝑎̃𝑟* .

*3.4. Duality fixing*

*𝑎̃𝑞 < 𝜇𝑎̃𝑟*

*𝑎̃𝑞* = *𝜇𝑎̃𝑟*

*𝑎̃𝑞 > 𝜇𝑎̃𝑟*

This well-known single-level presolve method tries to detect vari-

ables that can be fixed to certain values based on optimality conditions.

*𝜇 >* 0 inequality with index *𝑞* both inequalities inequality with index *𝑟*

is irrelevant are equivalent is irrelevant

*𝜇 <* 0 both inequalities are relevant infeasible model

Duality fixing is often very eﬃcient in practice and can be applied to both continuous and integer variables. Details on this method are given in [Martin (2001)](#_bookmark54) and [Achterberg et al. (2020)](#_bookmark45).

In the following, we briefly describe duality fixing for single-level

which has the unique optimal solution *𝑦*∗

= 2 and *𝑦*∗ = 2. Uncrushing

problems by considering the follower problem [(2)](#_bookmark5) for a fixed leader de-

yields *𝑥*∗ = 1 and *𝑦*∗ = 1.

1

* 1. *Parallel rows*

new 2

cision *𝑥* = *𝑥̄*. We now consider variables *𝑦𝑖* that fulfill one of the two

conditions:

1. *𝑓𝑖* ≥ 0 and *𝐷𝑗𝑖* ≤ 0 for all *𝑗* ∈ {1*,* … *,* 𝓁},
2. *𝑓𝑖* ≤ 0 and *𝐷𝑗𝑖* ≥ 0 for all *𝑗* ∈ {1*,* … *,* 𝓁}.

This presolve technique is also known from single-level optimization.

We briefly explain it based on the high-point relaxation [(3)](#_bookmark6) and use the notation

[ ] [ ] ( )

*𝐴̃* = *𝐴 , 𝐵̃* = *𝐵 , 𝑎̃* = *𝑎 .*

*𝐶 𝐷 𝑏*

Assume that the index *𝑖* fulfills Condition (i) and that *𝑦𝐹 >* holds.

*𝑖*

−∞

*𝑖*

Then we can fix *𝑦𝑖* = *𝑦𝐹* . For *𝑓𝑖 >* 0, this variable fixing is given directly as a necessary optimality condition. For *𝑓𝑖* = 0, there exists at least one optimal solution that satisfies this variable fixing. If we assume *𝑦𝐹* = −∞

instead of *𝑦𝐹 >* −∞, we have to distinguish two cases. For *𝑓𝑖 >* 0, the

*𝑖*

We call two inequalities with indices *𝑞, 𝑟* ∈ {1*,* … *, 𝑘* + 𝓁} parallel, if *𝑖*

*𝐴̃*

*𝑞*⋅

= *𝜇𝐴̃*

*𝑟*⋅

and *𝐵̃*

*𝑞*⋅

= *𝜇𝐵̃*

*𝑟*⋅

hold for *𝜇* ≠ 0. The sign of *𝜇* and the right-

problem is infeasible or unbounded and for *𝑓𝑖* = 0, the variable *𝑦𝑖* and all

constraints, for which *𝑦𝑖* has a nonzero coeﬃcient, can be removed from

hand sides *𝑎̃𝑞* and *𝑎̃𝑟* can be used to determine infeasibility of the entire

problem or that one of the two constraints is redundant. More precisely,

we distinguish the cases shown in [Table 1](#_bookmark20).

Detailed descriptions of this procedure for single-level optimiza- tion are given in [Andersen and Andersen (1995), Martin (2001)](#_bookmark47) and [Achterberg et al. (2020)](#_bookmark45). Although the occurrence of parallel rows can be seen as bad modeling, it is shown in [Achterberg et al. (2020)](#_bookmark45) that parallel rows occur rather frequently in real-world single-level instances and that their treatment contributes to a more eﬃcient solution process [for these instances. Note that the same algorithm as in Bixby and Wag- ner (1987) to detect parallel columns can also be used to detect parallel](#_bookmark49) rows.

We now turn to the linear bilevel problem [(1)](#_bookmark9). The two inequalities can now either be two leader inequalities, two follower inequalities, or one leader and one follower inequality. In the first two cases, we can directly apply the rules given in [Table 1](#_bookmark20). The third case requires a more

detailed discussion. We assume w.l.o.g. that *𝑞* is an index of a leader

inequality and *𝑟* is an index of a follower inequality. In case *𝜇 <* 0, the rules in [Table 1](#_bookmark20) still apply, because for *𝑎̃𝑞 > 𝜇𝑎̃𝑟* the model is infeasible and for *𝑎̃𝑞* ≤ *𝜇𝑎̃𝑟* no deduction can be made. However, in case *𝜇 >* 0, we

obtain different rules:

the problem. Condition (ii) can be tackled analogously. If *𝑦̄𝐹 <* ∞, we fix

*𝑦𝑖* = *𝑦̄𝐹* . For *𝑦̄𝐹* = ∞ we distinguish the same cases as for Condition (i).

*𝑖*

*𝑖 𝑖*

As shown in [Fischetti et al. (2017)](#_bookmark55), this single-level duality fixing can

into account the leader decision *𝑥*. be applied almost directly to the follower problem [(2)](#_bookmark5) without taking

**Theorem 2** ([Fischetti et al., 2017](#_bookmark55))**.** *For every follower variable 𝑦𝑖, 𝑖* ∈

{1*,* … *, 𝑚*}*, the following fixing is correct:*

1. *If 𝑓𝑖 >* 0 *and 𝐷𝑗𝑖* ≤ 0 *for all 𝑗* ∈ {1*,* … *,* 𝓁}*, fix 𝑦𝑖* = *𝑦𝐹 ,*
2. *If 𝑓𝑖 <* 0 *and 𝐷𝑗𝑖* ≥ 0 *for all 𝑗* ∈ {1*,* … *,* 𝓁}*, fix 𝑦𝑖* = *𝑦̄𝐹 .*

*𝑖*

*𝑖*

Note that in [Theorem 2](#_bookmark22) the case *𝑓𝑖* = 0 is not taken into account. The reason might be that this case is somehow problematic. For *𝑓𝑖* = 0, the optimal choice of *𝑦𝑖* by the follower might be ambiguous. Under the

follower solutions the one that she prefers. Thus, fixing *𝑦𝑖* must preserve optimistic assumption, the leader is free to choose among the optimal this optimistic solution. Since the case *𝑓𝑖* = 0 does hardly appear in our

instance set that we use in the computational study in [Section 4](#_bookmark25), we do

not further investigate this issue.

Until now, we have applied duality fixing to the follower problem. We now present an example that illustrates the occurring diﬃculties in

* 1. If *𝑎̃*

*< 𝜇𝑎̃* , then *𝐴̃ 𝑥* + *𝐵̃ 𝑦* ≥ *𝑎̃*

is implied by *𝑦* ∈ S(*𝑥*) and the

the application of duality fixing to the leader problem or to the leader

*𝑞 𝑟*

*𝑞 𝑞 𝑞*

and follower problem simultaneously.

leader inequality with index *𝑞* is redundant.

* 1. If *𝑎̃𝑞 > 𝜇𝑎̃𝑟*, then the follower constraint may only be active for a point (*𝑥, 𝑦*) that violates the leader constraint. Thus, remov-

**Example 2.** Consider the bilevel problem

ing this constraint only changes the rational reaction *𝑦* ∈ S(*𝑥*) for leader decisions *𝑥* ∉ Ω*𝐿*. Consequently, the follower inequality with index *𝑟* must be inactive for every bilevel feasible point and

min

*𝑥,𝑦*

*𝑥* − *𝑦* s*.*t*.* − 3*𝑥* + *𝑦* ≥ −3*, 𝑥* ≥ 0*, 𝑦* ∈ arg min{*𝑦̄* ∶ −2*𝑥* + *𝑦̄* ≥ 0}*.*

*𝑦̄*

(10)

we can safely remove it.

* 1. If *𝑎̃𝑞* = *𝜇𝑎̃𝑟*, then the inequalities are equivalent. For the follower inequality with index *𝑟*, the argument of Case (ii) does not hold,

because it can be active at a bilevel feasible point. Thus, remov- ing the follower constraint might change the response of the fol- lower in a way that renders the follower response infeasible for the leader constraints. However, the leader inequality with in-

dex *𝑞* is redundant and we can remove it.

able *𝑥* is negative. In addition, the coeﬃcient in the upper-level objective For all leader and follower constraints, the coeﬃcient of the leader vari-

single-level optimization, we would fix *𝑥* to its lower bound, i.e., *𝑥* = 0. function is positive. If we naively apply duality fixing as we would do in

This yields the point (0,0) with objective function value 0.

However, for every leader decision *𝑥*, the optimal solution of the follower is given by *𝑦* = 2*𝑥*. Thus, we can substitute *𝑦* with 2*𝑥* in the

leader problem of Problem [(10)](#_bookmark23) to obtain

In summary, this means that we can detect parallel rows using the

min

*𝑥*

−*𝑥* s*.*t*.* − *𝑥* ≥ −3*, 𝑥* ≥ 0*.*

shared constraint set Ω as long as we pay attention to Case (iii).

Until now, we only covered parallel inequalities. Equality constraints

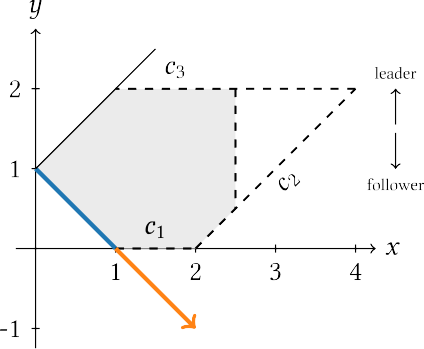
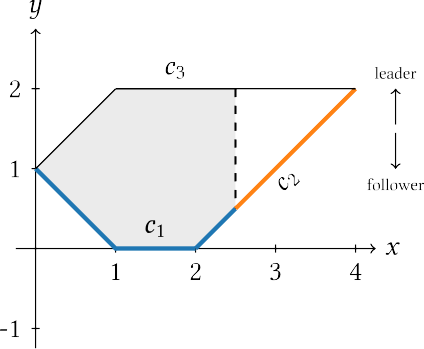
can, in principle, be reformulated as two inequalities, such that we may apply the techniques above. From a practical point of view, however, it is often not a good idea to divide an equation into two inequalities. Keep- ing the equation results in further cases that have to be distinguished; see, e.g., [Achterberg et al. (2020)](#_bookmark45).

This reformulated problem has the optimal solution *𝑥* = 3, such that we

value −3. Hence, the point (0,0) we obtained by fixing *𝑥* = 0 is not an obtain the optimal solution (3,6) of Problem [(10)](#_bookmark23) with objective function

optimal solution of Problem [(10)](#_bookmark23).

The reason for the observation in the example is that we omit the optimality of the follower in the fixing step. In other words, we applied

**Fig. 2.** Moving follower constraints *𝑐*1 , *𝑐*2 , and *𝑐*3

(left) that are inactive at the bilevel optimal solu-

tion (0,1) to the leader (right) changes the reaction of the follower (blue and orange) and the bilevel feasible set (blue), but retains the optimal solution. (For inter- pretation of the references to color in this figure leg- end, the reader is referred to the web version of this article.)

duality fixing to the high-point relaxation [(3)](#_bookmark6), which is not suﬃcient to guarantee correctness.

*3.5. Optimality-based presolve*

Up to now, we presented feasibility-based presolve methods that do not change the bilevel-feasible set T . In order to further simplify the solution of Problem [(1)](#_bookmark9), it might be desirable to tighten T in a way that retains at least one bilevel optimal solution. In particular, it would be beneficial to exclude bilevel-feasible points that can be proven to be not bilevel optimal. In general, this requires to respect bilevel optimality, i.e., to deal with the nonconvexity of the problem. One the one hand, optimality-based presolve can thus be expected to be a very diﬃcult task. On the other hand, one might be able to identify non-optimal fea- sible points without knowledge of the bilevel-optimal solution. In this section, we present a first step in this direction.

In [Section 3.1](#_bookmark15), we discussed to move constraints to tighten the linear problems used for the bound strengthening. To be specific, we moved

follower-independent leader constraints to the follower problem. This is a feasible approach because in this special case, the bilevel feasible set T

remains unchanged. However, it is well-known that moving constraints between the two levels changes the bilevel feasible set in general. In the following paragraphs we propose a criterion that can be used to detect follower constraints that can be safely moved to the leader prob- lem without changing the set of bilevel-optimal points—although the bilevel-feasible set T might be changed. We illustrate this approach in [Fig. 2](#_bookmark24), in which upper-level constraints correspond to dashed lines and lower-level constraints to solid lines. The specific problem can be found

constraint, i.e., (*𝑥, 𝑦*) ∈ T*𝑀* implies (*𝑥, 𝑦*) ∈ T . (ii) we do not obtain additional bilevel feasible points by moving the

We first show (i). Since *𝐶𝑖*⋅*𝑥*∗ + *𝐷𝑖*⋅*𝑦*∗ *> 𝑏𝑖* , the reaction *𝑦*∗ of the fol- lower must still be optimal for the leader decision *𝑥*∗ after removing the constraint from the follower problem. In addition, the point (*𝑥*∗*, 𝑦*∗) is

feasible for all original leader constraints as well as for the constraint

*𝐶𝑖*⋅*𝑥* + *𝐷𝑖*⋅*𝑦* ≥ *𝑏𝑖* . Thus, (*𝑥*∗*, 𝑦*∗) ∈ T*𝑀* holds.

We now turn to (ii). Let (*𝑥̄, 𝑦̄*) ∈ T*𝑀* , i.e., *𝑦̄* is optimal for the relaxed follower problem [(2)](#_bookmark5) for fixed *𝑥* = *𝑥̄*, in which the *𝑖*th follower constraint is removed. Since (*𝑥̄, 𝑦̄*) ∈ T*𝑀* , *𝐶𝑖*⋅*𝑥̄* + *𝐷𝑖*⋅*𝑦̄* ≥ *𝑏𝑖* holds and *𝑦̄* is also optimal for Problem [(2)](#_bookmark5) with fixed *𝑥* = *𝑥̄*. Thus, (*𝑥̄, 𝑦̄*) ∈ T . □

In order to turn [Theorem 3](#_bookmark29) into a presolve method, it is crucial to detect such inactive constraints. One approach might be to exploit the high-point relaxation in special cases. For example, in min-max prob- lems the objective functions of the leader and the follower point into opposite directions. Follower constraints that are binding for the solu- tion of the high-point relaxation are canonical candidates to be inactive at the bilevel-optimal solution—although, of course, easy examples can be constructed for which this does not hold. It remains an open question and subject of future research whether this technique can be carried out eﬃciently.

# Computational experiments

In this section, we try to shed some light on the effectiveness of the proposed presolve techniques. To this end, we consider the following single-level reformulation

on Page 33 of [Manns (2020)](#_bookmark52).

In addition, the shared constraint set Ω is colored gray, the bilevel

feasible set is colored in blue, and the set of optimal follower solu-

min

*𝑥,𝑦,𝜆,𝜆̄,𝜆*

*𝑐⊤𝑥* + *𝑑⊤𝑦* (11a)

tions lifted to the *𝑥*–*𝑦*-space is the union of the blue and orange lines.

attained at the point (0,1). We see that the lower-level constraints *𝑐*1, [Fig. 2](#_bookmark24) (left) shows the original problem, in which the bilevel solution is

*𝑐*2, and *𝑐*3 are not active in (0,1). In [Fig. 2](#_bookmark24) (right), we moved these con-

straints to the leader. This changes the optimal reaction of the follower

and, thus, also changes the bilevel feasible set. However, the bilevel optimal solution (0,1) is retained. We formalize this in the following theorem.

**Theorem 3.** *Let* (*𝑥*∗*, 𝑦*∗) *be an optimal solution of Problem* [*(1)*](#_bookmark9) *and consider* *a follower constraint with index 𝑖* ∈ {1*,* … *,* 𝓁} *such that 𝐶𝑖*⋅*𝑥*∗ + *𝐷𝑖*⋅*𝑦*∗ *> 𝑏𝑖 holds. Then,* (*𝑥*∗*, 𝑦*∗) *is also optimal for the problem that we obtain by moving*

*𝐶𝑖*⋅*𝑥* + *𝐷𝑖*⋅*𝑦* ≥ *𝑏𝑖 from the follower to the leader.*

**Proof.** We denote the bilevel feasible set of Problem [(1)](#_bookmark9) by T and the bilevel feasible set after moving the constraint by T*𝑀* . We prove the theorem by showing that

(i) the optimal solution (*𝑥*∗*, 𝑦*∗) remains bilevel feasible when moving the constraint, i.e., (*𝑥*∗*, 𝑦*∗) ∈ T*𝑀* , and

s*.*t*.* (*𝑥, 𝑦*) ∈ Ω*,* (11b)

*𝜆, 𝜆, 𝜆̄* ≥ 0*,* (11c)

*𝐷⊤𝜆* + *𝜆* − *𝜆̄* = *𝑓 ,* (11d)

*𝜆⊤*(*𝐶𝑥* + *𝐷𝑦* − *𝑏*) = 0*,* (11e)

*𝜆⊤*(*𝑦* − *𝑦𝐹* ) = 0*,* (11f)

*𝜆̄⊤*(*𝑦̄𝐹* − *𝑦*) = 0*.* (11g)

of Problem [(1)](#_bookmark9). This well-known reformulation has been proposed in [Fortuny-Amat and McCarl (1981)](#_bookmark60) and can be derived by replacing the lower-level problem [(2)](#_bookmark5) with its Karush–Kuhn–Tucker (KKT) condi- tions, i.e., primal feasibility that is contained in [(11b)](#_bookmark26), nonnegativity of the dual variables [(11c)](#_bookmark27), stationarity [(11d)](#_bookmark28), and complementarity [(11e)](#_bookmark30)– [(11g)](#_bookmark31). Due to the latter conditions, Problem (11) is a mathematical prob- lem with complementarity constraints (MPCC). The complementarity

constraints [(11e)](#_bookmark30) can be replaced using the mixed-integer reformula- tion

*𝑢* ∈ {0*,* 1}𝓁 *, 𝜆* ≤ *𝑀𝑢, 𝐶𝑥* + *𝐷𝑦* − *𝑏* ≤ *𝑀* (1 − *𝑢*);

see [Fortuny-Amat and McCarl (1981)](#_bookmark60) as well. The complementarity con-

requires additional binary variables *𝑢* and a suﬃciently large value *𝑀* . straints [(11f)](#_bookmark32) and [(11g)](#_bookmark31) can be treated in the same way. This formulation

without obstacles. Choosing *𝑀* too large may cause numerical insta- It is pointed out in [Pineda and Morales (2019)](#_bookmark58) that this approach is not

bilities and choosing it too small may result in suboptimal solutions of Problem [(1)](#_bookmark9). Further, in [Kleinert et al. (2020)](#_bookmark68), it is shown that verify-

ing the correctness of a given value *𝑀* is as hard as solving the orig-

inal bilevel problem. Still, this approach is by far the most frequently

used approach to solve linear bilevel problems in practice, which is why we test our presolve techniques for this approach.[1](#_bookmark34) Throughout all

tests, we set *𝑀* = 106. Although this might exclude some bilevel-optimal

points from the feasible set of the bilevel problem’s reformulation (11),

it still allows for a proper computational evaluation of the speed-ups





**Fig. 3.** Speed-up factors *𝑠𝑖* for duality fixing.

The effect of the presolve method is measured by the speed-up factor

obtained by applying presolve techniques, which is what we do in the

{ wo w

wo w

following.

*𝑠𝑖* =

*𝑡𝑖* ∕*𝑡𝑖 ,* if *𝑡𝑖* ≥ *𝑡𝑖 ,*

−*𝑡*w∕*𝑡*wo*,* else.

*𝑖 𝑖*

* 1. *Computational setup and test sets*

The big-*𝑀* based single-level reformulation as well as all presolve techniques have been implemented in C++-11 and have been compiled with GCC 7.5.0. We solved all mixed-integer problems with Gurobi

9.0.2. The computational experiments have been executed on a compute cluster using compute nodes with Intel Xeon E3-1240 v6 CPUs with 4 [cores, 3.7GHz, and 32GB RAM; see Regionales Rechenzentrum Erlangen (2020) for more details. Note the we limited the number of threads to 1](#_bookmark59) in all computations.

Our initial test set contains more than 2500 instances and mainly consists of the mixed-integer linear bilevel instances that are used in [Kleinert and Schmidt (2020)](#_bookmark48). This set consists of mixed-integer bilevel instances from the literature, for which the integrality conditions are re- laxed to obtain continuous bilevel problems. From this set, we removed all infeasible instances and all instances that we solve in less than 1s and that we thus consider as too easy. The resulting test set only con- tains roughly 600 instances, on which we test the proposed presolve methods in the following. Let us already comment that—although we started with a rather large instance set that contains (to the best of our knowledge) all available (mixed-integer) linear bilevel instances from the literature—most presolve methods can only be applied to a very small amount of instances. The reason is that not too many instances indeed have, e.g., parallel rows or columns. We will dis- cuss this test-library specific problem later on again in our conclusion in [Section 5](#_bookmark40). Since the number of relevant instances is rather small, we specify the instances that we use for each presolve method along with a reference to its origin in the literature in each of the following sections.

* 1. *Parallel rows*

In [Table 2](#_bookmark35), we compare the running times *𝑡*wo for solving an instance *𝑖* We detected parallel rows in only 8 out of the roughly 600 instances. without presolve with the running time *𝑡*w obtained if parallel-rows pre- solve is applied; see [Section 3.3](#_bookmark21). In addition, we also specify the time *𝑡𝑝*

*𝑖*

*𝑖*

*𝑖*

that is needed for detecting and removing parallel rows. Note that *𝑡*w

We see that removing parallel rows has a very significant effect on the neos-1109824 instance, which is solved almost 60 times faster if parallel-rows presolve is applied. The reason for this is an enormous reduction of constraints: around one third of all constraints can be re- moved; see the “# Rows” column in [Table 2](#_bookmark35). On the other instances, pre- solving has mixed effects. On some instances (gmu-35-40, acc-tight5, and rocI-4-11), we observe a moderate reduction in running time and of the number of constraints. However, on other instances, removing parallel rows has none or a rather negative effect. The latter is hard to explain on the data basis we have—especially because different aspects seem to be the reason for the observed results. For example, the instance bab5 is solved by a root-node heuristic if no presolve is applied, which is not the case if presolve is applied, whereas the instance gmu-35- 50 requires less branch-and-bound nodes to be solved if parallel-rows presolve is not used.

* 1. *Parallel columns*

We detected parallel columns in 7 instances. The results are similar to the case of parallel-rows presolve; see [Table 3](#_bookmark36).

Detecting parallel columns has a significantly positive effect on the instance tanglegram1, on which we observe a speed-up of over 8, which is due to 126 presolved columns. On all other instances, we ob- serve only moderate or no effects. This is surprising for some instances. For example, we remove more than 40% of the variables of the in- stance gmu-35-50. Still, the speed-up of 1.13 is rather small. More- over, we observed that the root node model after Gurobi’s own pre- solve is smaller if we apply our parallel-columns presolve beforehand. However, this is not reflected in significant speed-ups for the running time.

* 1. *Duality fixing*

We detected variables that can be fixed via duality fixing in 103 in- stances. Thus, duality fixing can be applied by far to the largest subset of instances of our test set. This is in line with the positive results reported

specifies the total running time including *𝑡𝑝*.

*𝑖*

*𝑖* in [Fischetti et al. (2017)](#_bookmark55), in which duality fixing is applied as well. Due

to the large number of instances, we illustrate the speed-up factors *𝑠*

1 Note that the presolve techniques evaluated in this section would also have a comparably beneficial effect on other solution approaches such as classic branch-and-bound for LP-LP bilevel problems ([Bard, 1998](#_bookmark43)) due to, e.g., a re- duced number of lower-level constraints.

*𝑖*

in a more condensed way in [Fig. 3](#_bookmark33). For the sake of completeness, we

also denote running times and the number of variables in [Table 6](#_bookmark39) in the appendix. We see that for the majority of instances, duality fixing is beneficial, and for many of them, the speed-up can be considered sig- nificant. For example, for 26 instances, we observe a speed-up of 1.5 or

**Table 2**

Running times for parallel row presolve. All times are given in seconds.

Running Time # Rows

*𝑖 𝑖*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instance *𝑖* | Ref. | *𝑡*wo | *𝑡*w | *𝑡𝑝 𝑠* |  | w/o | w |  |
| neos-1109824 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 63.51 | 1.06 | 0.11 59.92 |  | 28,979 | 9979 |  |
| gmu-35-40 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.28 | 0.86 | 0.01 1.49 |  | 424 | 419 |  |
| acc-tight5 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 12.37 | 9.55 | 0.03 1.30 |  | 3052 | 3045 |  |
| rocI-4-11 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 3.29 | 2.91 | 0.04 1.13 |  | 10,883 | 10,663 |  |
| neos-4647030-tutaki | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 206.08 | 209.87 | 3.08 -1.02 |  | 8382 | 8381 |  |
| unitcal\_7 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 82.29 | 84.94 | 0.14 -1.03 |  | 48,939 | 48,936 |  |
| bab5 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 17.21 | 28.45 | 0.14 -1.65 |  | 4964 | 4943 |  |
| gmu-35-50 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 2.00 | 3.30 | 0.01 -1.65 |  | 435 | 427 |  |

*𝑖 𝑖*

**Table 3**

Numerical results for parallel columns presolve. All times are given in seconds.

Running Time # Columns

*𝑖 𝑖*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instance *𝑖* | Ref. | *𝑡*wo | *𝑡*w | *𝑡𝑝 𝑠* |  | w/o | w |  |
| tanglegram1 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 42.54 | 5.19 | 0.31 8.20 |  | 34,759 | 34,633 |  |
| gmu-35-50 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 2.00 | 1.77 | 0.02 1.13 |  | 1919 | 1106 |  |
| unitcal\_7 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 82.49 | 81.42 | 0.20 1.01 |  | 25,755 | 24,747 |  |
| eilB101 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 23.84 | 23.88 | 0.03 -1.00 |  | 2818 | 2817 |  |
| istanbul-no-cutoff | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 24.18 | 24.36 | 0.08 -1.01 |  | 5282 | 5278 |  |
| tanglegram2 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.12 | 1.16 | 0.04 -1.04 |  | 4714 | 4680 |  |
| neos13 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.37 | 1.49 | 0.23 -1.09 |  | 1827 | 1826 |  |

*𝑖 𝑖*

**Table 4**

Numerical results for bound strengthening. All times are given in seconds.

Running Time

*𝑖 𝑖 𝑖 𝑖*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Instance *𝑖* | Ref. | *𝑡*wo | *𝑡*w | *𝑡𝑝* | *𝑠̃* | # Bounds |
| gmu-35-50 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 2.00 | 2.24 | 1.33 | 2.20 | 41 |
| gmu-35-40 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.28 | 1.35 | 0.56 | 1.62 | 17 |
| 2AP05-21 | ([DeNegre, 2011](#_bookmark54)) | 49.68 | 40.85 | 0.01 | 1.22 | 1 |
| n4-3 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.76 | 8.14 | 6.65 | 1.18 | 1637 |
| mcsched | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.61 | 3.10 | 1.50 | 1.01 | 24 |
| rococoB10-011000 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 1.78 | 8.51 | 6.74 | 1.01 | 385 |
| bnatt500 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 1.07 | 18.65 | 17.57 | -1.01 | 1055 |

**Table 5**

Numerical results for bound moving. All times are given in seconds.

Running Time

*𝑖 𝑖 𝑖 𝑖*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Instance *𝑖* | Ref. | *𝑡*wo | *𝑡*w | *𝑡𝑝* | *𝑠̃* | # Bounds |
| gmu-35-40 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.28 | 1.30 | 0.53 | 1.66 | 12 |
| 2AP05-21 | ([DeNegre, 2011](#_bookmark54)) | 49.68 | 38.09 | 0.01 | 1.30 | 1 |
| rococoB10-011000 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 1.78 | 10.96 | 9.20 | 1.01 | 385 |
| bnatt500 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 1.07 | 21.17 | 20.10 | -1.00 | 1055 |
| n4-3 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.76 | 6.36 | 4.59 | -1.01 | 13 |
| mcsched | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.61 | 3.88 | 1.48 | -1.49 | 24 |
| gmu-35-50 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 2.00 | 4.79 | 1.29 | -1.75 | 36 |

above, and for 55 instances, we observe a speed-up of 1.2 or above. In contrast, there are only very few instances, for which duality fixing has

a negative impact. Only 5 instances have a speed-up factor of −1*.*5 or

below.

* 1. *Bound strengthening and bound moving*

Finally, we consider bound strengthening (see [Section 3.1](#_bookmark15)) and bound moving (see [Remark 2](#_bookmark17)). These methods are applicable to 7 in- stances; see [Table 4](#_bookmark37) and [Table 5](#_bookmark38).

For both methods, the detection of implied tighter bounds is imple- mented in a computationally costly way by solving Problem [(6)](#_bookmark12) and [(7)](#_bookmark13). This could certainly be improved, e.g., by using internal solver infor-

mation that is not accessible when using commercial solvers. In or- der to measure the speed-up provided by the presolve method, we de-

note cleaned speed-up factors *𝑠̃𝑖* with respect to *𝑡̃*w = *𝑡*w − *𝑝𝑖* instead of

*𝑡𝑖* , i.e., we disregard the presolve time when computing the speed-up

w

*𝑖 𝑖*

factors.

For bound strengthening, we observe positive effects for the majority of instances. Comparing the two tables, we see that bound strengthen- ing clearly outperforms bound moving. One reason for this might be that for bound strengthening we may compute implied bounds even for fol- lower variables that are initially unbounded. In contrast, bound moving only shifts existing redundant bounds from the follower to the leader. Thus, bound strengthening affects more bounds as indicated in the final column of the two tables.

**Table 6**

Numerical results for duality fixing. All times are given in seconds.

Running Time # Variables

*𝑖 𝑖*

*𝑖 𝑖*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Instance *𝑖* | Ref. | *𝑡*wo | *𝑡*w | *𝑡𝑝 𝑠* |  | w/o | w |
| tanglegram1 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 42.57 | 10.36 | 0.07 4.11 |  | 34,759 | 17,968 |
| xuLarge900-1 | ([Fischetti et al., 2017](#_bookmark55)) | 70.74 | 23.41 | 0.26 3.02 |  | 1800 | 1355 |
| bmilplib\_310\_1 | ([Xu and Wang, 2014](#_bookmark67)) | 2.15 | 0.73 | 0.04 2.95 |  | 620 | 448 |
| bmilplib\_360\_9 | ([Xu and Wang, 2014](#_bookmark67)) | 3.19 | 1.15 | 0.05 2.77 |  | 720 | 550 |
| xuLarge800-4 | ([Fischetti et al., 2017](#_bookmark55)) | 7.25 | 2.67 | 0.21 2.72 |  | 1600 | 1227 |
| xuLarge500-9 | ([Fischetti et al., 2017](#_bookmark55)) | 6.12 | 2.35 | 0.09 2.60 |  | 1000 | 762 |
| xuLarge1000-8 | ([Fischetti et al., 2017](#_bookmark55)) | 107.76 | 42.13 | 0.32 2.56 |  | 2000 | 1476 |
| bmilplib\_360\_7 | ([Xu and Wang, 2014](#_bookmark67)) | 2.01 | 0.80 | 0.05 2.51 |  | 720 | 547 |
| bmilplib\_460\_6 | ([Xu and Wang, 2014](#_bookmark67)) | 5.68 | 2.31 | 0.07 2.46 |  | 920 | 683 |
| xuLarge900-4 | ([Fischetti et al., 2017](#_bookmark55)) | 70.74 | 29.97 | 0.26 2.36 |  | 1800 | 1341 |
| bmilplib\_410\_4 | ([Xu and Wang, 2014](#_bookmark67)) | 1.35 | 0.65 | 0.08 2.08 |  | 820 | 609 |
| xuLarge600-1 | ([Fischetti et al., 2017](#_bookmark55)) | 13.76 | 7.05 | 0.12 1.95 |  | 1200 | 887 |
| xuLarge500-4 | ([Fischetti et al., 2017](#_bookmark55)) | 8.40 | 4.36 | 0.08 1.93 |  | 1000 | 739 |
| bmilplib\_310\_2 | ([Xu and Wang, 2014](#_bookmark67)) | 1.88 | 1.01 | 0.04 1.86 |  | 620 | 458 |
| xuLarge800-5 | ([Fischetti et al., 2017](#_bookmark55)) | 5.60 | 3.14 | 0.21 1.78 |  | 1600 | 1189 |
| xuLarge800-9 | ([Fischetti et al., 2017](#_bookmark55)) | 20.68 | 11.60 | 0.21 1.78 |  | 1600 | 1197 |
| xuLarge700-5 | ([Fischetti et al., 2017](#_bookmark55)) | 9.27 | 5.30 | 0.16 1.75 |  | 1400 | 1027 |
| xuLarge800-1 | ([Fischetti et al., 2017](#_bookmark55)) | 49.98 | 29.00 | 0.21 1.72 |  | 1600 | 1215 |
| xuLarge900-5 | ([Fischetti et al., 2017](#_bookmark55)) | 21.15 | 12.70 | 0.26 1.67 |  | 1800 | 1342 |
| bmilplib\_460\_9 | ([Xu and Wang, 2014](#_bookmark67)) | 3.73 | 2.25 | 0.07 1.66 |  | 920 | 675 |
| bmilplib\_360\_2 | ([Xu and Wang, 2014](#_bookmark67)) | 4.25 | 2.65 | 0.05 1.60 |  | 720 | 546 |
| bmilplib\_310\_4 | ([Xu and Wang, 2014](#_bookmark67)) | 2.58 | 1.61 | 0.04 1.60 |  | 620 | 455 |
| xuLarge1000-4 | ([Fischetti et al., 2017](#_bookmark55)) | 6.14 | 3.86 | 0.32 1.59 |  | 2000 | 1479 |
| xuLarge600-5 | ([Fischetti et al., 2017](#_bookmark55)) | 8.11 | 5.14 | 0.12 1.58 |  | 1200 | 877 |
| xuLarge1000-3 | ([Fischetti et al., 2017](#_bookmark55)) | 99.36 | 64.99 | 0.32 1.53 |  | 2000 | 1461 |
| xuLarge700-10 | ([Fischetti et al., 2017](#_bookmark55)) | 24.49 | 16.27 | 0.16 1.51 |  | 1400 | 1038 |
| xuLarge700-1 | ([Fischetti et al., 2017](#_bookmark55)) | 3.28 | 2.20 | 0.16 1.49 |  | 1400 | 1035 |
| xuLarge900-6 | ([Fischetti et al., 2017](#_bookmark55)) | 19.63 | 13.18 | 0.26 1.49 |  | 1800 | 1366 |
| bmilplib\_410\_8 | ([Xu and Wang, 2014](#_bookmark67)) | 2.02 | 1.37 | 0.06 1.47 |  | 820 | 609 |
| xuLarge1000-5 | ([Fischetti et al., 2017](#_bookmark55)) | 35.67 | 24.72 | 0.32 1.44 |  | 2000 | 1488 |
| bmilplib\_460\_2 | ([Xu and Wang, 2014](#_bookmark67)) | 5.57 | 3.86 | 0.07 1.44 |  | 920 | 680 |
| bmilplib\_260\_2 | ([Xu and Wang, 2014](#_bookmark67)) | 1.14 | 0.80 | 0.03 1.42 |  | 520 | 394 |
| xuLarge1000-9 | ([Fischetti et al., 2017](#_bookmark55)) | 18.40 | 13.15 | 0.32 1.40 |  | 2000 | 1494 |
| bmilplib\_310\_6 | ([Xu and Wang, 2014](#_bookmark67)) | 1.36 | 0.97 | 0.04 1.40 |  | 620 | 456 |
| xuLarge600-3 | ([Fischetti et al., 2017](#_bookmark55)) | 5.31 | 3.83 | 0.12 1.39 |  | 1200 | 893 |
| xuLarge900-8 | ([Fischetti et al., 2017](#_bookmark55)) | 47.64 | 34.32 | 0.26 1.39 |  | 1800 | 1332 |
| xuLarge700-8 | ([Fischetti et al., 2017](#_bookmark55)) | 19.37 | 13.90 | 0.16 1.39 |  | 1400 | 1049 |
| bmilplib\_410\_9 | ([Xu and Wang, 2014](#_bookmark67)) | 5.50 | 4.03 | 0.06 1.36 |  | 820 | 620 |
| bmilplib\_310\_8 | ([Xu and Wang, 2014](#_bookmark67)) | 1.21 | 0.89 | 0.04 1.36 |  | 620 | 446 |
| bmilplib\_360\_5 | ([Xu and Wang, 2014](#_bookmark67)) | 2.37 | 1.77 | 0.05 1.34 |  | 720 | 548 |
| bmilplib\_460\_8 | ([Xu and Wang, 2014](#_bookmark67)) | 7.64 | 5.78 | 0.08 1.32 |  | 920 | 682 |
| xuLarge800-7 | ([Fischetti et al., 2017](#_bookmark55)) | 7.85 | 5.98 | 0.21 1.31 |  | 1600 | 1199 |
| bmilplib\_310\_7 | ([Xu and Wang, 2014](#_bookmark67)) | 1.38 | 1.07 | 0.04 1.29 |  | 620 | 485 |
| xuLarge600-8 | ([Fischetti et al., 2017](#_bookmark55)) | 6.41 | 4.98 | 0.12 1.29 |  | 1200 | 917 |
| bmilplib\_310\_10 | ([Xu and Wang, 2014](#_bookmark67)) | 3.20 | 2.48 | 0.04 1.29 |  | 620 | 474 |
| bmilplib\_410\_2 | ([Xu and Wang, 2014](#_bookmark67)) | 3.08 | 2.41 | 0.06 1.28 |  | 820 | 597 |
| xuLarge700-3 | ([Fischetti et al., 2017](#_bookmark55)) | 56.34 | 44.84 | 0.16 1.26 |  | 1400 | 1051 |
| xuLarge900-3 | ([Fischetti et al., 2017](#_bookmark55)) | 55.40 | 43.87 | 0.30 1.26 |  | 1800 | 1333 |
| xuLarge600-2 | ([Fischetti et al., 2017](#_bookmark55)) | 7.60 | 6.02 | 0.12 1.26 |  | 1200 | 881 |
| xuLarge800-2 | ([Fischetti et al., 2017](#_bookmark55)) | 100.68 | 80.77 | 0.21 1.25 |  | 1600 | 1206 |
| xuLarge500-5 | ([Fischetti et al., 2017](#_bookmark55)) | 7.63 | 6.17 | 0.09 1.24 |  | 1000 | 759 |
| bmilplib\_360\_4 | ([Xu and Wang, 2014](#_bookmark67)) | 4.57 | 3.70 | 0.05 1.24 |  | 720 | 536 |
| bmilplib\_360\_3 | ([Xu and Wang, 2014](#_bookmark67)) | 2.66 | 2.17 | 0.05 1.23 |  | 720 | 549 |
| bmilplib\_460\_4 | ([Xu and Wang, 2014](#_bookmark67)) | 3.67 | 3.06 | 0.07 1.20 |  | 920 | 689 |
| bmilplib\_460\_1 | ([Xu and Wang, 2014](#_bookmark67)) | 4.85 | 4.03 | 0.08 1.20 |  | 920 | 699 |
| xuLarge600-10 | ([Fischetti et al., 2017](#_bookmark55)) | 6.13 | 5.14 | 0.12 1.19 |  | 1200 | 895 |
| xuLarge600-6 | ([Fischetti et al., 2017](#_bookmark55)) | 7.51 | 6.33 | 0.12 1.19 |  | 1200 | 914 |
| xuLarge500-1 | ([Fischetti et al., 2017](#_bookmark55)) | 10.97 | 9.24 | 0.09 1.19 |  | 1000 | 755 |
| xuLarge1000-1 | ([Fischetti et al., 2017](#_bookmark55)) | 23.32 | 19.78 | 0.32 1.18 |  | 2000 | 1482 |
| xuLarge800-6 | ([Fischetti et al., 2017](#_bookmark55)) | 28.41 | 24.11 | 0.21 1.18 |  | 1600 | 1205 |
| xuLarge500-7 | ([Fischetti et al., 2017](#_bookmark55)) | 6.83 | 5.93 | 0.08 1.15 |  | 1000 | 753 |
| bmilplib\_410\_1 | ([Xu and Wang, 2014](#_bookmark67)) | 4.79 | 4.16 | 0.06 1.15 |  | 820 | 625 |
| xuLarge500-10 | ([Fischetti et al., 2017](#_bookmark55)) | 3.10 | 2.69 | 0.09 1.15 |  | 1000 | 739 |
| bmilplib\_310\_9 | ([Xu and Wang, 2014](#_bookmark67)) | 1.97 | 1.71 | 0.04 1.15 |  | 620 | 471 |
| xuLarge700-7 | ([Fischetti et al., 2017](#_bookmark55)) | 12.97 | 11.58 | 0.16 1.12 |  | 1400 | 1040 |
| bmilplib\_410\_6 | ([Xu and Wang, 2014](#_bookmark67)) | 2.41 | 2.15 | 0.06 1.12 |  | 820 | 609 |
| bmilplib\_410\_3 | ([Xu and Wang, 2014](#_bookmark67)) | 1.72 | 1.55 | 0.06 1.11 |  | 820 | 614 |
| gmu-35-50 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 2.00 | 1.80 | 0.01 1.11 |  | 1919 | 1769 |
| xuLarge600-7 | ([Fischetti et al., 2017](#_bookmark55)) | 15.23 | 13.72 | 0.12 1.11 |  | 1200 | 895 |
| xuLarge500-8 | ([Fischetti et al., 2017](#_bookmark55)) | 4.05 | 3.68 | 0.09 1.10 |  | 1000 | 759 |
| bmilplib\_460\_3 | ([Xu and Wang, 2014](#_bookmark67)) | 3.32 | 3.04 | 0.07 1.09 |  | 920 | 695 |
| wachplan | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 1.32 | 1.22 | 0.03 1.08 |  | 3361 | 2870 |
| cvs16r128-89 | ([MIPLIB2017, 2017; Kleinert and Schmidt, 2020](#_bookmark56)) | 15.27 | 14.41 | 0.01 1.06 |  | 3472 | 3160 |
| bmilplib\_360\_10 | ([Xu and Wang, 2014](#_bookmark67)) | 2.17 | 2.10 | 0.05 1.03 |  | 720 | 541 |

(*continued on next page*)

**Table 6** (*continued*)

Running Time # Variables

Instance *𝑖* Ref. *𝑡*wo w

*𝑖*

*𝑡*

*𝑖*

*𝑝*

*𝑖 𝑖*

*𝑡*

*𝑠*

w/o w

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| xuLarge700-4 | ([Fischetti et al., 2017](#_bookmark55)) | 21.16 | 20.54 | 0.16 | 1.03 | 1400 | 1059 |
| xuLarge700-6 | ([Fischetti et al., 2017](#_bookmark55)) | 11.23 | 11.04 | 0.16 | 1.02 | 1400 | 1040 |
| bmilplib\_360\_1 | ([Xu and Wang, 2014](#_bookmark67)) | 2.36 | 2.31 | 0.05 | 1.02 | 720 | 533 |
| bmilplib\_410\_5 | ([Xu and Wang, 2014](#_bookmark67)) | 2.49 | 2.46 | 0.06 | 1.01 | 820 | 600 |
| bmilplib\_360\_8 | ([Xu and Wang, 2014](#_bookmark67)) | 3.73 | 3.71 | 0.05 | 1.01 | 720 | 543 |
| tanglegram2 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.12 | 1.12 | 0.01 | -1.00 | 4714 | 2581 |
| bmilplib\_410\_10 | ([Xu and Wang, 2014](#_bookmark67)) | 9.56 | 9.62 | 0.06 | -1.01 | 820 | 631 |
| xuLarge800-10 | ([Fischetti et al., 2017](#_bookmark55)) | 24.10 | 24.39 | 0.21 | -1.01 | 1600 | 1200 |
| bmilplib\_460\_5 | ([Xu and Wang, 2014](#_bookmark67)) | 10.60 | 10.68 | 0.07 | -1.01 | 920 | 702 |
| gmu-35-40 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 1.28 | 1.35 | 0.01 | -1.05 | 1205 | 1055 |
| xuLarge800-3 | ([Fischetti et al., 2017](#_bookmark55)) | 92.04 | 97.73 | 0.21 | -1.06 | 1600 | 1195 |
| xuLarge900-9 | ([Fischetti et al., 2017](#_bookmark55)) | 51.79 | 54.74 | 0.26 | -1.06 | 1800 | 1330 |
| xuLarge600-9 | ([Fischetti et al., 2017](#_bookmark55)) | 12.76 | 13.54 | 0.12 | -1.06 | 1200 | 891 |
| bmilplib\_460\_7 | ([Xu and Wang, 2014](#_bookmark67)) | 2.94 | 3.11 | 0.08 | -1.06 | 920 | 684 |
| xuLarge700-2 | ([Fischetti et al., 2017](#_bookmark55)) | 26.89 | 29.05 | 0.16 | -1.08 | 1400 | 1061 |
| satellites1-25 | ([Koch et al., 2011](#_bookmark49); [Kleinert and Schmidt, 2020](#_bookmark48)) | 59.16 | 64.37 | 0.03 | -1.09 | 9013 | 9012 |
| xuLarge1000-10 | ([Fischetti et al., 2017](#_bookmark55)) | 48.11 | 54.28 | 0.33 | -1.13 | 2000 | 1516 |
| xuLarge600-4 | ([Fischetti et al., 2017](#_bookmark55)) | 6.54 | 7.86 | 0.12 | -1.20 | 1200 | 919 |
| xuLarge500-6 | ([Fischetti et al., 2017](#_bookmark55)) | 7.12 | 8.86 | 0.09 | -1.24 | 1000 | 756 |
| xuLarge500-3 | ([Fischetti et al., 2017](#_bookmark55)) | 4.67 | 5.84 | 1.89 | -1.25 | 1000 | 749 |
| xuLarge1000-2 | ([Fischetti et al., 2017](#_bookmark55)) | 13.10 | 16.95 | 0.32 | -1.29 | 2000 | 1513 |
| xuLarge900-7 | ([Fischetti et al., 2017](#_bookmark55)) | 89.33 | 116.28 | 0.27 | -1.30 | 1800 | 1361 |
| xuLarge500-2 | ([Fischetti et al., 2017](#_bookmark55)) | 12.64 | 16.58 | 0.09 | -1.31 | 1000 | 761 |
| xuLarge800-8 | ([Fischetti et al., 2017](#_bookmark55)) | 15.86 | 22.97 | 0.21 | -1.45 | 1600 | 1181 |
| xuLarge1000-7 | ([Fischetti et al., 2017](#_bookmark55)) | 116.16 | 177.57 | 0.32 | -1.53 | 2000 | 1460 |
| xuLarge900-2 | ([Fischetti et al., 2017](#_bookmark55)) | 58.69 | 92.74 | 0.26 | -1.58 | 1800 | 1341 |
| xuLarge1000-6 | ([Fischetti et al., 2017](#_bookmark55)) | 18.48 | 32.71 | 0.32 | -1.77 | 2000 | 1488 |
| xuLarge900-10 | ([Fischetti et al., 2017](#_bookmark55)) | 123.22 | 221.17 | 0.26 | -1.79 | 1800 | 1338 |
| bmilplib\_410\_7 | ([Xu and Wang, 2014](#_bookmark67)) | 2.90 | 6.41 | 0.06 | -2.21 | 820 | 619 |

# Conclusion

In this paper, we systematically studied presolve methods for linear bilevel problems. Due to the fact that the IIC property does not hold for bilevel problems in general, this is a delicate task and the application of standard presolve methods from single-level optimization thus has to be done with great caution. Fortunately, we are able to carry over bound strengthening, the handling of parallel rows and columns, duality fixing, and a special type of an optimality-based presolve to the field of bilevel optimization.

Our numerical results indicate that the studied presolve methods can both be very beneficial for some instances but can also harm the solu- tion process for other problems of our test set. However, the number of instances to which, e.g., parallel-rows or parallel-columns, presolve could be applied in our numerical experiments is too small to discuss the general impact of presolve for linear bilevel problems. This, in particu- lar, reveals that there is an urgent need of better test instance libraries to further propel the field of computational bilevel optimization. The instance sets used in computational (mixed-integer) linear bilevel op- timization are mostly interdiction instances that (i) have a very special structure and that (ii) are too easy if their inherent integrality conditions are relaxed so that they can be used to test techniques to solve contin- uous bilevel problems. In particular, (i) is most likely the reason that all interdiction instances of our general test set do not play any role for the numerical experiments carried out in this paper. Moreover, there are almost no real-world test instances publicly available that usually possess much more structures such as parallel columns or rows. In our opinion—and we are rather convinced that the numerical study in this paper clearly reveals this aspect—the availability of better test instances is of crucial importance for the further development of the field. Finally, the field would also benefit from publicly available open-source solver frameworks for bilevel optimization so that one is not restricted to use commercial solvers as black-boxes.

# Declaration of Competing Interest

The authors declare that there is no conflict of interest.

# Acknowledgments

This research has been performed as part of the Energie Campus Nürnberg and is supported by funding of the Bavarian State Govern- ment. We also thank the Deutsche Forschungsgemeinschaft for their support within project A05, B08, and Z01 in the “Sonderforschungs- bereich/Transregio 154 Mathematical Modelling, Simulation and Opti- mization using the Example of Gas Networks”.

# Appendix A. Detailed Results for Duality Fixing

In Table 6, we list detailed results for all of the 103 instances to which we applied duality fixing.

# References

Achterberg, T., Bixby, R.E., Gu, Z., Rothberg, E., Weninger, D., 2014. Multi-row presolve reductions in mixed integer programming. In: Hosei University, T. (Ed.), Proceedings of the Twenty-Sixth RAMP Symposium (Tokyo), pp. 181–196. <http://www.orsj.or.jp/ramp/2014/paper/4-4.pdf>.

Achterberg, T., Bixby, R.E., Gu, Z., Rothberg, E., Weninger, D., 2020. Presolve reductions in mixed integer programming. INFORMS Journal on Computing 32 (2), 473–506. doi:[10.1287/ijoc.2018.0857](https://doi.org/10.1287/ijoc.2018.0857).

Andersen, E.D., Andersen, K.D., 1995. Presolving in linear programming. Mathematical Programming 71 (2), 221–245. doi:[10.1007/BF01586000](https://doi.org/10.1007/BF01586000).

Audet, C., Savard, G., Zghal, W., 2007. New branch-and-cut algorithm for bilevel linear programming. Journal of Optimization Theory and Applications 134 (2), 353–370. doi:[10.1007/s10957-007-9263-4](https://doi.org/10.1007/s10957-007-9263-4).

Bard, J.F., 1998. Practical bilevel optimization: algorithms and applications, 30. Springer Science & Business Media. doi:[10.1007/978-1-4757-2836-1](https://doi.org/10.1007/978-1-4757-2836-1).

Bard, J.F., Moore, J.T., 1990. A branch and bound algorithm for the bilevel program- ming problem. SIAM Journal on Scientific and Statistical Computing 11 (2), 281–292. doi:[10.1137/0911017](https://doi.org/10.1137/0911017).

Belotti, P., Cafieri, S., Lee, J., Liberti, L., 2012. On feasibility based bounds tighten- [ing. Tech. rep. Preprint. Optimization Online. http://www.optimization-online.org/ DB\_HTML/2012/01/3325.html.](http://www.optimization-online.org/DB_HTML/2012/01/3325.html)

[Belotti, P., Cafieri, S., Lee, J., Liberti, L., 2010. Feasibility-based bounds tightening via fixed points. In: Wu, W., Daescu. Berlin, O. (Eds.), Combinatorial optimization and applications. Springer Berlin Heidelberg, Heidelberg, pp. 65–76.](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0008)

Bixby, R.E., Wagner, D.K., 1987. A note on detecting simple redundancies in linear sys- tems. Operations Research Letters 6 (1), 15–17. doi:[10.1016/0167-6377(87)90004-6](https://doi.org/10.1016/0167-6377(87)90004-6). Chen, W.-K., Gemander, P., Gleixner, A., Gottwald, L., Martin, A., Weninger, D., 2020. Two-row and two-column mixed-integer presolve using hashing-based pair-

ing methods. EURO Journal on Computational Optimization 8 (3), 205–240. doi:[10.1007/s13675-020-00129-6](https://doi.org/10.1007/s13675-020-00129-6).

Dempe, S., 2002. Foundations of bilevel programming. Springer. doi:[10.1007/b101970](https://doi.org/10.1007/b101970). [Dempe, S., Lohse, S., 2011. Dependence of bilevel programming on irrelevant data. Tech.](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0012)

[rep. Preprint. TU Bergakademie Freiberg.](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0012)

[DeNegre, S.T., 2011. Interdiction and discrete bilevel linear programming. PhD Thesis.](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0013)

[Lehigh University.](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0013)

Fischetti, M., Ljubić, I., Monaci, M., Sinnl, M., 2017. A new general-purpose algorithm for mixed-integer bilevel linear programs. Operations Research 65 (6), 1615–1637. doi:[10.1287/opre.2017.1650](https://doi.org/10.1287/opre.2017.1650).

Fischetti, M., Ljubić, I., Monaci, M., Sinnl, M., 2018. On the use of intersec- tion cuts for bilevel optimization. Mathematical Programming 172 (1), 77–103. doi:[10.1007/s10107-017-1189-5](https://doi.org/10.1007/s10107-017-1189-5).

Fischetti, M., Monaci, M., Sinnl, M., 2018. A dynamic reformulation heuristic for general- ized interdiction problems. European Journal of Operational Research 267 (1), 40–51. doi:[10.1016/j.ejor.2017.11.043](https://doi.org/10.1016/j.ejor.2017.11.043).

Fortuny-Amat, J., McCarl, B., 1981. A Representation and economic interpretation of a two-level programming problem. The Journal of the Operational Research Society 32 (9), 783–792. doi:[10.1057/jors.1981.156](https://doi.org/10.1057/jors.1981.156).

Gamrath, G., Koch, T., Martin, A., Miltenberger, M., Weninger, D., 2015. Progress in pre- solving for mixed integer programming. Mathematical Programming Computation 7, 367–398. doi:[10.1007/s12532-015-0083-5](https://doi.org/10.1007/s12532-015-0083-5).

Gleixner, A.M., Berthold, T., Müller, B., Weltge, S., 2017. Three enhancements for optimization-based bound tightening. Journal of Global Optimization 67 (4), 731– 757. doi:[10.1007/s10898-016-0450-4](https://doi.org/10.1007/s10898-016-0450-4).

Hansen, P., Jaumard, B., Savard, G., 1992. New branch-and-bound rules for linear bilevel programming. SIAM Journal on Scientific and Statistical Computing 13 (5), 1194– 1217. doi:[10.1137/0913069](https://doi.org/10.1137/0913069).

Kleinert, T., Labbé, M., Plein, F., Schmidt, M., 2021. Closing the gap in linear bilevel op- timization: a new valid primal-dual inequality. Optimization Letters 15, 1027–1040. doi:[10.1007/s11590-020-01660-6](https://doi.org/10.1007/s11590-020-01660-6).

Kleinert, T., Labbé, M., Plein, F., Schmidt, M., 2020. Technical note—there’s no free lunch: on the hardness of choosing a correct big-M in bilevel optimization. Operations Re- search 68 (6), 1716–1721. doi:[10.1287/opre.2019.1944](https://doi.org/10.1287/opre.2019.1944).

Kleinert, T., Schmidt, M., 2020. Computing feasible points of bilevel problems with a penalty alternating direction method. INFORMS Journal on Computing doi:[10.1287/ijoc.2019.0945](https://doi.org/10.1287/ijoc.2019.0945).

Koch, T., Achterberg, T., Andersen, E., Bastert, O., Berthold, T., Bixby, R.E., Danna, E., Gamrath, G., Gleixner, A.M., Heinz, S., Lodi, A., Mittelmann, H., Ralphs, T., Sal- vagnin, D., Steffy, D.E., Wolter, K., 2011. MIPLIB 2010. Mathematical Programming Computation 3 (2), 103–163. doi:[10.1007/s12532-011-0025-9](https://doi.org/10.1007/s12532-011-0025-9), Online first.

Macal, C.M., Hurter, A.P., 1997. Dependence of bilevel mathematical programs on irrelevant constraints. Computers & Operations Research 24 (12), 1129–1140. doi:[10.1016/S0305-0548(97)00025-7](https://doi.org/10.1016/S0305-0548(97)00025-7).

[Manns,](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0026) [J., 2020. Presolve of linear bilevel programs. MA thesis. Friedrich-Alexander-Uni- versität Erlangen-Nürnberg.](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0026)

Martin, A., 2001. General mixed integer programming: computational issues for branchand-cut algorithms. In: Naddef, D., Jünger, M. (Eds.), Computational combina- torial optmization, 2241. Springer, Berlin, pp. 1–25. doi:[10.1007/3-540-45586-8\_1](https://doi.org/10.1007/3-540-45586-8_1).

MIPLIB 2017. <http://miplib.zib.de/>. Accessed: 2021-03-05. 2018.

Moore, J.T., Bard, J.F., 1990. The mixed integer linear bilevel programming problem.

Operations Research 38 (5), 911–921. doi:[10.1287/opre.38.5.911](https://doi.org/10.1287/opre.38.5.911).

Pineda, S., Morales, J.M., 2019. Solving linear bilevel problems using big-Ms: not all that glitters is gold. IEEE Transactions on Power Systems 34 (3), 2469–2471. doi:[10.1109/TPWRS.2019.2892607](https://doi.org/10.1109/TPWRS.2019.2892607).

Regionales [Rechenzentrum Erlangen, 2020. Woodcrest Cluster. https://www.anleitungen. rrze.fau.de/hpc/woody-cluster/ (visited on 11/30/2020).](https://www.anleitungen.rrze.fau.de/hpc/woody-cluster/)

Ryoo, H.S., Sahinidis, N.V., 1996. A branch-and-reduce approach to global optimization.

Journal of Global Optimization 8 (2), 107–138. doi:[10.1007/BF00138689](https://doi.org/10.1007/BF00138689).

[Savelsbergh,](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0033) [M.W.P., 1994. Preprocessing and probing techniques for mixed integer pro- gramming problems. ORSA Journal on Computing 6, 445–454.](http://refhub.elsevier.com/S2192-4406(21)00147-7/sbref0033)

Tahernejad, S., Ralphs, T.K., DeNegre, S.T., 2020. A branch-and-cut algorithm for mixed integer bilevel linear optimization problems and its implementation. Mathematical Programming Computation 1–40. doi:[10.1007/s12532-020-00183-6](https://doi.org/10.1007/s12532-020-00183-6).

Vicente, L., Savard, G., Júdice, J., 1994. Descent approaches for quadratic bilevel programming. Journal of Optimization Theory and Applications 81 (2), 379–399. doi:[10.1007/BF02191670](https://doi.org/10.1007/BF02191670).

Xu, P., Wang, L., 2014. An exact algorithm for the bilevel mixed integer linear program- ming problem under three simplifying assumptions. Computers & Operations Research 41, 309–318. doi:[10.1016/j.cor.2013.07.016](https://doi.org/10.1016/j.cor.2013.07.016).