Noname manuscript No.
(will be inserted by the editor)

A Single-Level Reformulation of Integer Bilevel Programs using Decision Diagrams

Sebastián Vásquez · Leonardo Lozano · Willem-Jan van Hoeve.

Received: date / Accepted: date

Abstract Integer bilevel programs are notoriously difficult to solve due to the absence of strong and efficiently computable relaxations. In this work, we introduce a novel single-level reformulation of these programs by leveraging a network flow-based representation of the follower's value function, utilizing decision diagrams and linear programming duality. This approach enables the development of scalable relaxations by applying it to a restricted solution set, which in turn provides dual bounds. We obtain an exact method by iteratively solving and strengthening the relaxation. We further extend the reformulation to address the pessimistic version of the original bilevel problem. We experimentally compare our method with state-of-the-art bilevel programming solvers, demonstrating competitive performance. Specifically, on the BOBILib benchmark set our approach provides new or improved bounds for six instances and closes two open instances for the first time. We also show experimentally that the decision diagram reformulation can be particularly effective when it can leverage the combinatorial structure of the follower's problem.

Keywords Bilevel Optimization · Integer Programming · Decision Diagrams

Mathematics Subject Classification (2020) 90C10 · 91A65 · 90C46 · 90C35

Sebastián Vásquez
Carnegie Mellon University, Pittsburgh, PA 15213, USA
E-mail: savasque@andrew.cmu.edu.com

Leonardo Lozano
University of Cincinnati, Cincinnati, OH 45221, USA
E-mail: leolozano@uc.edu

Willem-Jan van Hoeve
Carnegie Mellon University, Pittsburgh, PA 15213, USA
E-mail: vanhoeve@andrew.cmu.edu

1 Introduction

Bilevel optimization problems naturally arise when two players, a leader and a follower, interact and aim to optimize their separate objective functions subject to joint as well as separate constraints [9]. It is assumed that there is no private information; the entire model is visible to both the leader and the follower. However, the leader only controls its own decisions, and the follower will optimize its decisions based on the leader's decisions. When the follower has multiple optimal responses to the leader's solution, the optimistic setting assumes that the follower's response benefits the leader, while the pessimistic setting assumes that the follower's response disfavors the leader. The goal is to find a solution that optimizes the leader's objective. Bilevel optimization problems have many important practical applications, including transportation network design [33,27], management science [2], supply chain management [16], and finance [26]. Many such problems contain integer decisions; a well-known special case are network interdiction problems for which the leader and follower decisions are all binary [35]. Additional applications of bilevel programming are discussed in [3,15,25,5].
For continuous bilevel programs, single-level reformulations can be derived using strong duality or the Karush-Kuhn-Tucker (KKT) conditions, leading to nonlinear programs or mathematical programs with complementarity constraints [20,4,43]. However, these techniques cannot be directly applied to the integer case, making it more challenging to reformulate and solve. Current state-of-the-art exact solvers extend mixed-integer programming techniques, either by using known integer programming valid cuts [17,18,19,36] or by sampling and enumerating solutions [31,29,30]. For most of these approaches, a key challenge is repeatedly evaluating the follower's response, which typically requires solving a separate integer program.
In this work, we introduce a new single-level linear reformulation for integer bilevel programming problems using decision diagrams, which encodes the follower's solution space as a directed acyclic graph. This approach eliminates the need to repeatedly solve the follower's response. It comes at a price however, as the exact reformulation may grow to an exponential size. To address this, we introduce a polynomial-sized decision diagram relaxation that provides a dual bound. Stronger bounds can be achieved with larger diagrams, allowing for a trade-off between computational time and bound quality.

Contributions Our first contribution is a new single-level linear reformulation for integer bilevel programming problems. It replaces the well-known value function representation for the follower's problem by a reformulation that encodes all solutions to the follower's problem in a decision diagram. The reformulation consists of a network flow model defined over the decision diagram, with capacity constraints parameterized by the leader's solution, and new variables and constraints derived from linear programming duality to ensure the follower selects an optimal solution. Combined with the original leader problem formulation, this results in an exact reformulation. When the decision diagram size is small enough, the reformulation can be directly solved within reasonable time. Our second contribution is the development of dual bounds based on polynomial-sized restricted decision diagrams. Standard decision diagram relaxations do not yield valid bounds for bilevel programs. However, we show that valid dual bounds can be obtained by using restricted decision diagrams that include only a subset of the follower's decisions. In addition, we can iteratively strengthen this formulation to obtain an exact solution method. As our third contribution, we extend our approach to the pessimistic version of the bilevel program, showing that the reformulation can be adapted to this variant, resulting in a strengthened relaxation. Lastly, our fourth contribution is an experimental comparison with state-of-the-art mixed-integer bilevel programming solvers, where we achieve competitive results for certain problem classes and outperform existing solvers on instances where the follower's problem has a combinatorial structure suitable for decision diagrams.

We note three closely related approaches leveraging decision diagrams for bilevel optimization. The first one, proposed in [29], considers bilevel interdiction problems, in which both the leader variables x and the follower variables y are binary, and the follower problem constraints are of the specific form $x \leqslant 1 - y$. The authors provide a single-level reformulation of the bilevel problem using exact decision diagrams and exploiting the special structure of the follower's constraints. We use a similar technique and extend it to a much more general setting while also constructing relaxations and considering the pessimistic version of the problem. The second one, presented in [30], focuses on general linear binary bilevel programs, and uses decision diagrams to represent leader's decisions associated with the same follower's problem optimal value. To the best of our knowledge, this is the only single-level reformulation for general linear binary bilevel programs available in the literature. This work is related to ours in the sense that both approaches seek to provide single-level reformulations and relaxations of the original bilevel problem. However, our approach is based on constructing parametric decision diagrams that serve as convexification devices for the follower problem, allowing us to exploit duality arguments, while their approach is based on enumerative and combinatorial arguments resulting in considerably different reformulations. The third one, introduced by [11], leverages decision diagrams to model a pricing problem in which the leader maximizes a revenue function by setting tolls on items, resulting in modified prices perceived by a follower aiming to minimize a cost function. Here, the decision diagram encodes the follower's feasible decisions for fixed tolls. This approach is similar to ours since both rely on restricted decision diagrams to approximate the follower problem. However, the problem setting, the construction of the decision diagrams, and the resulting reformulations are widely different.

Organization The remainder of this work is organized as follows. Section 2 introduces the definitions and relevant notation. Section 3 presents our single-level reformulation and explains how to use decision diagrams to model the follower's value function. Section 4 shows a procedure to generate valid dual bounds through approximate decision diagrams. It also describes an iterative compilation procedure that yields an exact method. Section 5 explains how to contract decision diagram layers to achieve smaller reformulations. Section 6 extends our reformulation to address the pessimistic version of the original bilevel program. Section 8 presents numerical results and a comparison of our reformulation with state-of-the-art solvers. Finally, we provide a conclusion in Section 9.

2 Integer Bilevel Programming Definitions

In this section we first provide the definition of integer linear bilevel programming problems. We then introduce the high-point relaxation and the value-function reformulation. We conclude this section with a binary expansion of the integer model, which will be used for our decision diagram construction.

2.1 Integer Linear Bilevel Programming Problem

We consider integer linear bilevel programming problems of the following form. Let $x \in \mathbb{Z}^{n_L}$ and $y \in \mathbb{Z}^{n_F}$ be the leader's and the follower's decision variables, respectively, where $n_L \in \mathbb{N}_+$ and $n_F \in \mathbb{N}_+$ are given constants. Let $m_L \in \mathbb{N}_+$ and $m_F \in \mathbb{N}_+$ respectively denote the number of constraints for the leader and follower. We let $c_L \in \mathbb{R}^{n_L}$, $c_F, d \in \mathbb{R}^{n_F}$, $a \in \mathbb{R}^{m_L}$, $b \in \mathbb{R}^{m_F}$, $A \in \mathbb{R}^{m_L \times n_L}$, $C \in \mathbb{R}^{m_F \times n_L}$, $B \in \mathbb{R}^{m_L \times n_F}$, and $D \in \mathbb{R}^{m_F \times n_F}$. The canonical integer linear bilevel programming problem is formulated as:

$$\begin{aligned} f(x,y) = \min \ & c_L^\top x + c_F^\top y && \text{(1a)} \\ \text{s.t.} \ & Ax + By \leqslant a && \text{(1b)} \\ & x \in \mathbb{Z}^{n_L} && \text{(1c)} \\ & y \in \operatorname{argmin}_{\bar{y} \in \mathbb{Y}(x)} \left\{ d^\top \bar{y} \right\}, && \text{(1d)} \end{aligned}$$

where (1b) are called *linking constraints*, (1c) indicate integrality requirement for the leader's variables, and (1d) forces the follower to best respond with a solution y solving the follower's problem over its own domain given a vector x, defined as

$$\mathbb{Y}(x) := \{ y \in \mathbb{Z}^{n_F} : \ Dy \leqslant b - Cx \}.$$

As it is common in the literature, we define $f(x,y) = +\infty$ if (x,y) violates constraints (1b), or $\mathbb{Y}(x) = \varnothing$. Any feasible solution to problem (1) is called *bilevel-feasible*. Additionally, we denote as $\mathbb{Y}(\cdot)$ the set of feasible solutions (x,y) to the follower's problem, *i.e.*,

$$\mathbb{Y}(\cdot) = \{ (x,y) \in \mathbb{Z}^{n_L} \times \mathbb{Z}^{n_F} : Cx + Dy \leqslant b \}.$$

Importantly, as the follower may have alternative optimal responses for a given leader's solution, the literature presents two ways of breaking ties: the optimistic and the pessimistic setting. Under the optimistic setting, the follower's response benefits the leader, *i.e.*, it minimizes $c_L^\top x + c_F^\top y$ among alternative optimal follower solutions. Under the pessimistic setting, the follower's response disfavors the leader, *i.e.*, it maximizes $c_L^\top x + c_F^\top y$ among alternative optimal follower solutions. Throughout this work we will consider the optimistic setting; we discuss the extension to the pessimistic setting in Section 6.

2.2 High Point Relaxation

A natural relaxation of the set of feasible solutions to (1) is the *High Point Relaxation* (HPR), which is obtained by removing the optimality condition for the follower response [34]. Formally, it is defined as

$$H := \left\{ (x, y) \in \mathbb{Z}^{n_L} \times \mathbb{Z}^{n_F} : \begin{array}{c} Ax + By \leqslant a, \\ Cx + Dy \leqslant b \end{array} \right\},$$

noting that $\min \left\{ c_L^\top x + c_F^\top y : \ (x, y) \in H \right\}$ is a valid lower bound for problem (1). The HPR is embedded as the principal relaxation for state-of-the-art mixed-integer bilevel programming solvers, often strengthened with additional cuts [19,36,25].

2.3 Value Function Reformulation

The set containing all solutions feasible to problem (1) is called the *bilevel-feasible set*, which can be represented as

$$\left\{ (x, y) \in H : \ y \in \operatorname*{argmin}_{\bar{y} \in \mathbb{Y}(x)} \left\{ d^\top \bar{y} \right\} \right\}.$$

This representation motivates the *value function reformulation* of model (1). Given x satisfying (1b)-(1c), the *value function* $\phi : \mathbb{Z}^{n_L} \mapsto \mathbb{R}$ is defined as

$$\phi(x) := \min \left\{ d^\top y : \ y \in \mathbb{Y}(x) \right\}. \tag{2}$$

Then, the value function reformulation of (1) is

$$\begin{aligned} \min \ & c_L^\top x + c_F^\top y && \text{(3a)} \\ \text{s.t.} \ & (x, y) \in H && \text{(3b)} \\ & d^\top y \leqslant \phi(x). && \text{(3c)} \end{aligned}$$

This description forms the basis of our work, as we will compute ϕ using a decision diagram.

$$\begin{aligned} \min\ & -x-2y \\ \text{s.t.}\ & x+y \leqslant 5 \\ & 0 \leqslant x \leqslant 4 \\ & x \in \mathbb{Z} \\ & y \in \operatorname{argmin}_{\bar{y}\in\mathbb{Y}(x)} \{\bar{y}\} \end{aligned}$$

a. BIP model

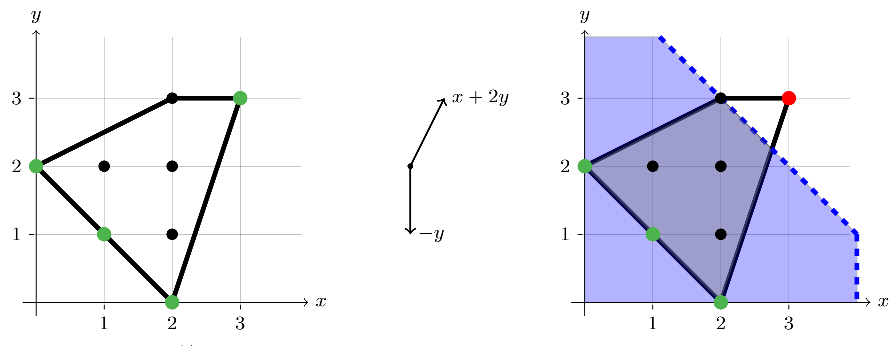
$$\mathbb{Y}(\cdot) := \left\{ \begin{array}{l} 2y \leqslant 4+x \\ y \geqslant 2-x \\ y \geqslant -6+3x \\ 0 \leqslant y \leqslant 3 \\ (x,y) \in \mathbb{Z}^2 \end{array} \right\}$$

b. Follower's domain

$$\begin{aligned} \min\ & -x-2y \\ \text{s.t.}\ & x+y \leqslant 5 \\ & 0 \leqslant x \leqslant 4 \\ & x \in \mathbb{Z} \\ & y \in \mathbb{Y}(x) \end{aligned}$$

c. High Point Relaxation

Fig. 1: Example integer bilevel programming model (a), the follower's feasible set (b), and the associated HPR (c).



a. $\mathbb{Y}(\cdot)$ b. Intersection with leader's domain

Fig. 2: The follower's feasible set $\mathbb{Y}(\cdot)$ for the example model in Fig. 1. Green points depict optimal follower's responses given specific x values. The blue-shaded polyhedron represents the leader's domain.

Example 1 As a running example, consider the integer bilevel programming problem in Fig. 1a with the follower's feasible set $\mathbb{Y}(\cdot)$ in Fig. 1b. Fig. 2 depicts the set $\mathbb{Y}(\cdot)$, where green points denote parametric optimal responses by the follower. Note that $x = 4$ is a feasible leader's decision, but infeasible to the overall model because $\mathbb{Y}(4) = \varnothing$. Also, given $x = 3$, the corresponding follower's optimal response is $y = 3$. However, $(x, y) = (3, 3)$ violates the linking constraint $x+y \leqslant 5$. Therefore, the leader cannot make decision $x = 3$. The set of bilevel-feasible solutions is $\{(0, 2),\ (1, 1),\ (2, 0)\}$ and the optimal solution to the overall model is $(x, y) = (0, 2)$, achieving an optimal value of -4. On the other hand, the optimal solution to the HPR, depicted in Fig. 1(c), is $(x, y) = (2, 3)$, with associated objective value -8. □

2.4 Binary Expansion

Because our representation requires all variables to be binary, we will make the following assumptions in the remainder of the paper. First, we assume that all integer variables have a bounded domain (which is reasonable for a computational method). We can then assume, without loss of generality, that $x \in \{0,1\}^{n_L}$ and $y \in \{0,1\}^{n_F}$ in program (1). Namely, we can first apply a linear transformation to make the bounds of each integer variable nonnegative. Then we apply a standard binary expansion: Each integer variable $z \in \{l, \dots, u\}$, with $0 < l < u$, is binarized by introducing $\lfloor \log(u-l) \rfloor + 1$ new binary variables z'_j and defining $z = l + \sum_{j=0}^{\lfloor \log(u-l) \rfloor} 2^j z'_j$. As a result, we focus henceforth in the binary version of the problem and assume that $x \in \{0,1\}^{n_L}$ and $y \in \{0,1\}^{n_F}$.

3 Exact Single Level Reformulation

We next describe how the value function ϕ can be computed using a decision diagram, leading to an exact single-level reformulation of problem (1). Our approach is based on three components: 1) a decision diagram to encode the domain of ϕ, 2) an associated network flow formulation that connects the decision diagram to the original problem, and 3) optimality conditions to enforce an optimal follower's response.

3.1 Decision Diagram Definitions

We follow the terminology from [8,29] to introduce decision diagrams. Given a set of binary points $\mathcal{X} \subseteq \{0,1\}^n$, a binary decision diagram is a directed acyclic graph $\mathcal{D} = (\mathcal{N}, \mathcal{A}, l, \nu)$ encoding every solution in $\mathcal{X}$, where $\mathcal{N}$ is a set of nodes, $\mathcal{A} \subset \mathcal{N} \times \mathcal{N}$ a set of arcs, $l \in \mathbb{R}^{|\mathcal{A}|}$ a vector of arc-costs, and $\nu \in \{0,1\}^{|\mathcal{A}|}$ a vector of arc-values representing "decisions". We define a "root node" $r \in \mathcal{N}$ and a "sink node" $t \in \mathcal{N}$. Node set $\mathcal{N}$ is partitioned into $n+1$ layers $\mathcal{N}_1, \dots, \mathcal{N}_{n+1}$, where $\mathcal{N}_1 = \{r\}$ and $\mathcal{N}_{n+1} = \{t\}$. Assuming a fixed but arbitrary ordering of the binary points in $\mathcal{X}$, we introduce a mapping $i : [n] \to [n]$ such that layer $\ell \in [n]$ is uniquely associated with index $i(\ell) \in [n]$ representing the $i(\ell)$-th component of a vector in $\mathcal{X}$. Every arc $a = (u,v) \in \mathcal{A}$ connects consecutive ascending layers, *i.e.*, if $u \in \mathcal{N}_\ell$, then $v \in \mathcal{N}_{\ell+1}$. For notational convenience, we define $\mathcal{A}_\ell \subset \mathcal{A}$ as the set of arcs leaving layer $\ell \in [n]$, and $\ell(a)$ as the layer from which arc $a \in \mathcal{A}$ leaves. Arcs with value 0 are called 0-arcs and arcs with value 1 are called 1-arcs. For any node $u \in \mathcal{N}$, $\delta^+(u)$ and $\delta^-(u)$ denote the subsets of $\mathcal{A}$ containing all arcs leaving and entering node u, respectively.

A diagram $\mathcal{D}$ encodes $\mathcal{X}$ through r-t paths in the following way. An arc-specified r-t path $(a_1, a_2, \dots, a_n)$ in $\mathcal{D}$ encodes $\hat{x} = (\nu_{a_1}, \nu_{a_2}, \dots, \nu_{a_n})$ in $\mathcal{X}$, and vice versa, where $\hat{x}$ follows the ordering induced by the diagram. Moreover, if x has an associated cost vector $c \in \mathbb{R}^n$, then for every layer $\ell \in [n]$ and arc $a \in \mathcal{A}_\ell$, we set $l_a = \nu_a c_{i(\ell)}$. This way, we obtain $c^\top \hat{x} = \sum_{\ell \in [n]} l_{a_\ell}$, *i.e.*, the length of the path encoding $\hat{x}$ is equal to the cost of $\hat{x}$. This implies that a shortest r-t path in $\mathcal{D}$ corresponds to an optimal solution to $\min_{x \in \mathcal{X}} \{c^\top x\}$. To simplify notation, we will use $i(a)$ to denote $i(\ell(a))$. Finally, a diagram $\mathcal{D}$

is called *exact* if it encodes every solution in $\mathcal{X}$, it is called *relaxed* if it encodes a superset of $\mathcal{X}$, and it is called *restricted* if it encodes a subset of $\mathcal{X}$.

The benefit of using decision diagrams for representing solutions is that *equivalent* nodes, *i.e.*, nodes with the same set of partial paths reaching the sink node, can be merged. A decision diagram is called *reduced* if no two nodes in the same layer are equivalent. An efficient procedure to make a given decision diagram reduced is presented in [10]. Given a fixed variable ordering, the procedure will find the unique reduced diagram of minimum size. Aside from its efficiency for representing circuits and Boolean functions [28,1,10,41], decision diagrams have proven beneficial in discrete optimization for designing cut generation/separation algorithms [39,12], generating bounds [40,21,6], and modeling/solving combinatorial problems [14,23,24,37]; see [8,13,22] and the references therein.

3.2 Decision Diagram for the Value Function

To compute $\phi(x)$ for any $x \in \{0,1\}^{n_L}$, we define a decision diagram $\mathcal{D} = (\mathcal{N}, \mathcal{A}, l, \nu)$ with the following structure:

- It has $n_F + n_L + 1$ layers split into two disjoint subsets: the *follower layer*s go from layer 1 to layer $n_F + 1$ and encode y values, while the *leader layers* go from layer $n_F + 2$ to layer $n_F + n_L + 1$ and encode x values. Similarly, we call any arc leaving a node in a follower layer a *follower arc*, and any arc leaving a node in a leader layer a *leader arc*.
- The diagram encodes all vectors in $\mathbb{Y}(\cdot)$.
- The arc-length vector l is defined as $l_a = d_{i(a)} \cdot \nu_a$ for any follower arc a, and $l_a = 0$ for any leader arc a.
- We additionally impose capacity constraints on the arcs, similar to [32]. Follower arcs have infinite capacity, while leader 1-arcs and 0-arcs in layer ℓ have capacity $x_{i(\ell)}$ and $\left(1 - x_{i(\ell)}\right)$, respectively.

Example 2 We continue Example 1. Fig. 3(a) shows a decision diagram encoding all solutions for the follower's feasible set $\mathbb{Y}(\cdot)$. Fig. 3(b) shows the unique reduced decision diagram representing the same solution set. $\square$

We let $\mathcal{A}^0 \subset \mathcal{A}$ and $\mathcal{A}^1 \subset \mathcal{A}$ be the set of leader 0-arcs and 1-arcs, respectively. We next use $\mathcal{D}$ to compute the value function through a network flow model.

Proposition 1 *Consider $\widehat{x} \in \{0,1\}^{n_L}$ such that $\mathbb{Y}(\widehat{x}) \neq \varnothing$. We have*

$$
\begin{aligned}
\phi(\widehat{x}) \; = \min \; & \sum_{a \in \mathcal{A}} l_a w_a && && \text{(4a)}\\
\text{s.t.} \; & \sum_{a \in \delta^+(u)} w_a - \sum_{a \in \delta^-(u)} w_a = \begin{cases} 1, & \textit{if } u = r \\ -1, & \textit{if } u = t \\ 0, & \textit{if } u \neq r, t \end{cases} && u \in \mathcal{N} && \text{(4b)}\\
& w_a \leqslant 1 - \widehat{x}_{i(a)} && a \in \mathcal{A}^0 && \text{(4c)}\\
& w_a \leqslant \widehat{x}_{i(a)} && a \in \mathcal{A}^1 && \text{(4d)}\\
& w \in \mathbb{R}_+^{|\mathcal{A}|}. && && \text{(4e)}
\end{aligned}
$$

Proof By construction, there is a one-to-one correspondence between binary points in $\mathbb{Y}(\cdot)$ and paths in the diagram, *i.e.*, a flow vector $\widehat{w}$ corresponds to exactly one pair $(\widehat{x}, \widehat{y})$ and vice versa. The length of the path represented by flow vector $\widehat{w}$ is

$$
\sum_{a \in \mathcal{A}} l_a \widehat{w}_a = d^\top \widehat{y}, \tag{5}
$$

where $\widehat{y}$ is the follower portion of the binary vector encoded by $\widehat{w}$.
Since the diagram encodes all vectors in $\mathbb{Y}(\cdot)$, then for a fixed vector $\widehat{x}$, constraints (4c) and (4d) ensure that the follower's solutions represented by all feasible paths exactly describe set $\mathbb{Y}(\widehat{x})$. As a result, for a shortest path represented by w^* we have

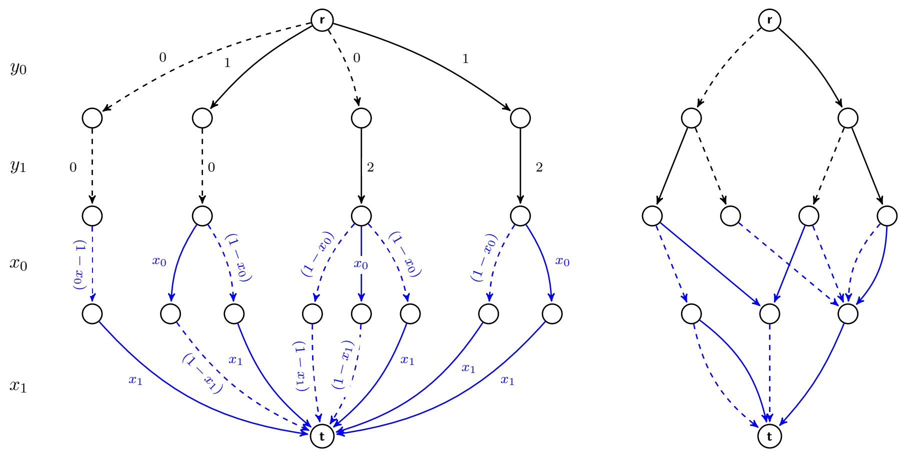
$$
\sum_{a \in \mathcal{A}} l_a w_a^* = d^\top y^* = \min \left\{ d^\top y : \; y \in \mathbb{Y}(\widehat{x}) \right\} = \phi(\widehat{x}), \tag{6}
$$

where y^* is the follower's portion of the binary vector encoded by w^*. □

3.3 Single Level Reformulation

Because the optimal value of the linear program (4) corresponds to the value function given a leader's decision $x \in \mathbb{Z}^{n_L}$, we can create a single-level reformulation of (1) using linear programming duality. The dual of (4) is:

$$
\begin{aligned}
\max \; & g(\widehat{x}, \xi) && && \text{(7a)}\\
\text{s.t.} \; & \pi_u - \pi_v \leqslant l_{uv} && (u,v) = a \in \mathcal{A} \setminus \left(\mathcal{A}^0 \cup \mathcal{A}^1 \right) && \text{(7b)}\\
& \pi_u - \pi_v - \lambda_{uv} \leqslant l_{uv} && (u,v) = a \in \mathcal{A}^0 && \text{(7c)}\\
& \pi_u - \pi_v - \beta_{uv} \leqslant l_{uv} && (u,v) = a \in \mathcal{A}^1 && \text{(7d)}\\
& \pi_t = 0 && && \text{(7e)}\\
& \lambda \in \mathbb{R}_+^{|\mathcal{A}^0|}, \; \beta \in \mathbb{R}_+^{|\mathcal{A}^1|} && && \text{(7f)}\\
& \xi = (\pi, \lambda, \beta), && && \text{(7g)}
\end{aligned}
$$



a. Decision diagram representing all vectors in $\mathbb{Y}(\cdot)$ b. Reduced decision diagram

Fig. 3: a) Decision diagram encoding solutions in $\mathbb{Y}(\cdot)$ for the example in Fig. 1, using a binary expansion. Nodes with incoming black arcs are in follower layers, while nodes with blue incoming arcs are in leader layers. Dashed arcs represent 0-arcs and solid arcs represent 1-arcs. Black arcs are follower arcs with arc-costs as labels, and blue arcs are leader arcs with arc-capacities as labels. b) The reduced decision diagram encoding the same solution set.

where

$$g(x, \xi) = \pi_r - \sum_{a \in \mathcal{A}^0} \left(1 - x_{i(a)}\right) \lambda_a - \sum_{a \in \mathcal{A}^1} x_{i(a)} \beta_a.$$

Moreover, we set $\pi_t = 0$ because the corresponding flow-balance constraint is linearly dependent on the remaining constraints. Let $P(\mathcal{D})$ denote the feasible solution set of model (7). We can now reformulate problem (1) as

$$\min\ c_L^\top x + c_F^\top y \tag{8a}$$

$$\text{s. t. } (x, y) \in H \tag{8b}$$

$$d^\top y \leqslant g(x, \xi) \tag{8c}$$

$$\xi \in P(\mathcal{D}). \tag{8d}$$

Proposition 2 *Model* (8) *is an exact single-level reformulation of the integer bilevel program* (1)*.*

Proof We show that (8) is equivalent to model (3), which in turn is a well-known exact single-level reformulation of the binary bilevel programming problem (1).
We first consider a feasible solution $(\hat{x}, \hat{y}, \hat{\xi})$ to model (8) and show that $(\hat{x}, \hat{y})$ is feasible to model (3). Constraints (3b) and (8b) are identical and thus $(\hat{x}, \hat{y})$ satisfies (3b). By Proposition 1 and weak duality, we obtain

$$\phi(\hat{x}) \geqslant g\left(\hat{x}, \hat{\xi}\right).$$

Therefore, constraint (8c) ensures that

$$d^\top \widehat{y} \leqslant \phi(\widehat{x}),$$

and thus $(\widehat{x}, \widehat{y})$ satisfies (3c). We conclude that $(\widehat{x}, \widehat{y})$ is feasible to model (3).
We now show that for any solution $(\widehat{x}, \widehat{y})$ feasible to model (3), there exists ξ such that $(\widehat{x}, \widehat{y}, \xi)$ is feasible to model (8). Let $\widehat{w}$ be the flow vector associated with the unique path in the diagram encoding $(\widehat{x}, \widehat{y})$. By construction, we have that $\widehat{w}$ is feasible for model (4) and

$$\sum_{a \in \mathcal{A}} l_a \widehat{w}_a = d^\top \widehat{y}.$$

Furthermore, constraint (3c) and Proposition 1 ensure that $\widehat{w}$ is an optimal solution to model (4) with arc capacities induced by $\widehat{x}$, *i.e.*,

$$\sum_{a \in \mathcal{A}} l_a \widehat{w}_a = \phi(\widehat{x}).$$

Now, let ξ' be an optimal solution to model (7). By definition, ξ' satisfies (8d) and, by strong duality, satisfies

$$\phi(\widehat{x}) = g(\widehat{x}, \xi').$$

We conclude that $(\widehat{x}, \widehat{y}, \xi')$ is feasible to model (8).
Because the objective functions for models (3) and (8) are the same, any feasible solution to one of them can be transformed into a feasible solution to the other with the same objective value. □

Observe that constraint (8c) contains the bilinear terms $x_{i(a)}\lambda_a$ and $x_{i(a)}\beta_a$. The following result shows how the constraint can be linearized.

Proposition 3 *Constraint* (8c) *can be replaced with the following system*

$$d^\top y \leqslant \pi_r, \tag{9a}$$
$$\lambda_a \leqslant M x_{i(a)} \qquad a \in \mathcal{A}^0, \tag{9b}$$
$$\beta_a \leqslant M\left(1 - x_{i(a)}\right) \qquad a \in \mathcal{A}^1, \tag{9c}$$

for a sufficiently large constant M.

Proof Since any feasible solution $(\widehat{x}, \widehat{y}, \widehat{\xi})$ to model (8) satisfies the KKT conditions for model (4) through constraint (8c), it must satisfy complementary slackness conditions.
Let $\widehat{\xi} = (\widehat{\pi}, \widehat{\lambda}, \widehat{\beta})$ and $\widehat{w}$ be the flow vector associated with the unique path in the diagram encoding $(\widehat{x}, \widehat{y})$. By construction, we have that $\widehat{w}$ is feasible for model (4). Now, consider an equivalent formulation of (4), in which the capacity constraints (4c) and (4d) are alternatively written as:

$$w_a \leqslant K - K x_{i(a)} \qquad a \in \mathcal{A}^0 \tag{10a}$$
$$w_a \leqslant K x_{i(a)} \qquad a \in \mathcal{A}^1 \tag{10b}$$

for some $K > 1$. By complementary slackness, we have

$$\left(K - K\widehat{x}_{i(a)} - \widehat{w}_a\right)\widehat{\lambda}_a = 0 \qquad \forall a \in \mathcal{A}^0 . \tag{11}$$

We examine two cases. If $\widehat{x}_{i(a)} = 1$, then $\widehat{w}_a = 0$ because of (10a), and (11) is satisfied for any value of $\widehat{\lambda}_a$. If $\widehat{x}_{i(a)} = 0$, then (11) becomes

$$\left(K - \widehat{w}_a\right)\widehat{\lambda}_a = 0 \qquad \forall a \in \mathcal{A}^0,$$

which can only be satisfied when $\widehat{\lambda}_a = 0$ since $K > 1$ and $\widehat{w}_a \leqslant 1$. Hence, we have

$$\left(\widehat{x}_{i(a)} = 0\right) \implies \left(\widehat{\lambda}_a = 0\right) \qquad \forall a \in \mathcal{A}^0 . \tag{12}$$

Following a similar argument, we obtain

$$\left(\widehat{x}_{i(a)} = 1\right) \implies \left(\widehat{\beta}_a = 0\right) \qquad \forall a \in \mathcal{A}^1 . \tag{13}$$

A big-M formulation for (12) and (13) is given by

$$\widehat{\lambda}_a \leqslant M\widehat{x}_{i(a)} \qquad a \in \mathcal{A}^0 \tag{14a}$$

$$\widehat{\beta}_a \leqslant M\left(1 - \widehat{x}_{i(a)}\right) \qquad a \in \mathcal{A}^1, \tag{14b}$$

for a sufficiently large M value. Moreover, (12) and (13) imply that

$$\sum_{a\in\mathcal{A}^0}\left(1 - \widehat{x}_{i(a)}\right)\widehat{\lambda}_a = \sum_{a\in\mathcal{A}^1}\widehat{x}_{i(a)}\widehat{\beta}_a = 0. \tag{15}$$

Replacing the above in (8c) yields

$$\sum_{a\in\mathcal{A}} l_a\widehat{w}_a = \widehat{\pi}_r. \tag{16}$$

□

Furthermore, we can determine a valid value M using the following proposition.

Proposition 4 *A valid value M for* (9b)-(9c) *is given by $M := \theta - l^\top\bar{w}$, where $\theta = \max\left\{d^\top y : (x, y) \in \mathbb{Y}(\cdot)\right\}$ and $\bar{w}$ is a flow vector associated to a shortest path in $\mathcal{D}$.*

Proof Consider a feasible solution $(\widehat{x}, \widehat{y}, \widehat{\xi})$ to model (8). We show that there exists $\xi = (\pi, \lambda, \beta)$ such that $(\widehat{x}, \widehat{y}, \xi)$ is also feasible to model (8), and

$$\lambda_a \leqslant M \qquad \forall a \in \mathcal{A}^0,$$

$$\beta_a \leqslant M \qquad \forall a \in \mathcal{A}^1,$$

for $M := \theta - l^\top\bar{w}$.

We iteratively build a vector $\xi' = (\pi', \lambda', \beta')$. Let $\widehat{w}$ be the flow vector associated with the unique path in the diagram encoding $(\widehat{x}, \widehat{y})$. Start by defining ξ' such that

$$\begin{aligned}
&\pi'_r = l^\top \widehat{w}, \\
&\pi'_t = 0, \\
&\pi'_v = \max_{(u,v)\in\delta^-(v)} \left\{\pi'_u - l_{uv}\right\} && \forall v \in \mathcal{N}\backslash\{r,t\} \\
&\lambda' = \beta' = 0.
\end{aligned}$$

Observe that, for any node $u \in \mathcal{N}_{n_F+1}$, we have $l^\top \widehat{w} - \pi'_u \leqslant M$. Moreover, for any r-t path with a length strictly lower than $l^\top \widehat{w}$, there exists an arc $a \in \mathcal{A}^0 \cup \mathcal{A}^1$ such that either $(a \in \mathcal{A}^0 \wedge \widehat{x}_{i(a)} = 1)$ or $(a \in \mathcal{A}^1 \wedge \widehat{x}_{i(a)} = 0)$. Otherwise, $l^\top \widehat{w}$ would not be the shortest length induced by $\widehat{x}$, $\widehat{\xi}$ would not be an optimal solution to model (7) and, thus, $(\widehat{x}, \widehat{y}, \widehat{\xi})$ would not be feasible to model (8). Now, for each such arc a, update the value of λ'_a to M if the first condition is met, or the value of β'_a to M if the second one is met. Finally, for each r-t path with a length strictly lower than $l^\top \widehat{w}$, identify the node v that is closest to t for which there is an arc $a \in \delta^-(v)$ with $\lambda'_a = M$ or $\beta'_a = M$, and update π'_k to 0 for all nodes k in the v-t path.
By construction, ξ' is a feasible solution to model (7) and, as $\pi'_r = l^\top \widehat{w}$, it is also optimal for $\widehat{x}$. Thus, $(\widehat{x}, \widehat{y}, \xi')$ is feasible for model (8). We conclude that $M = \theta - l^\top \bar{w}$ is a valid value for (9b)-(9c). □

We remark that from the proof of Proposition 4, one can compute θ by solving a longest path problem over $\mathcal{D}$, and one can also define more refined values for M for sub-parts of the diagram, depending on the value of the shortest path into the nodes in the last follower layer.

4 Dual Bounds from Restricted Decision Diagrams

Because an exact decision diagram to compute ϕ may grow exponentially large, we next propose a more scalable approach that uses an approximate decision diagram of polynomial size. Two types of approximate decision diagrams have been proposed in the literature [7,8]. The first are *relaxed* decision diagrams that encode a superset of a given set of solutions. The relaxation is obtained by merging nodes in the diagram that are not necessarily equivalent, inducing possibly infeasible r-t paths. The second are *restricted* decision diagrams that encode a subset of feasible solutions. Restricted diagrams can be obtained by discarding nodes (and the associated arcs) from the diagram. In both cases, the approximate diagrams are typically obtained by merging/discarding nodes until each layer contains at most a given maximum number of nodes, referred to as the decision diagram *width*.

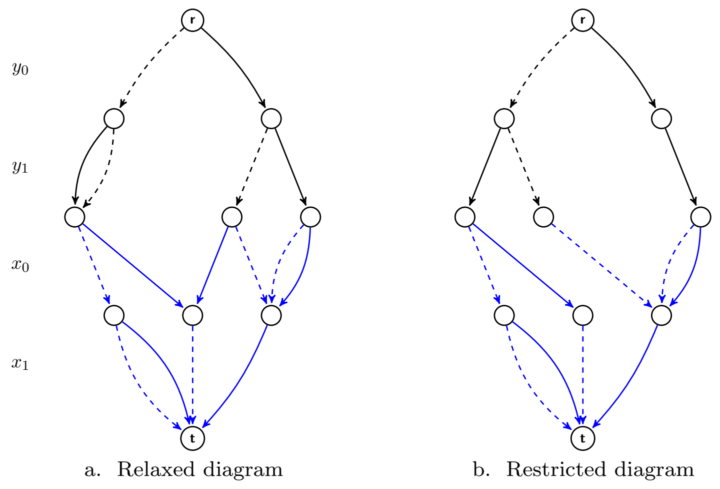


Fig. 4: Examples of approximate decision diagrams when forcing the width of the compiled diagram to be at most 3. a) A relaxed decision diagram for the running example. b) a restricted decision diagram for the running example.

4.1 Obtaining a Dual Bound

For standard optimization problems, optimizing over a relaxed decision diagram provides a dual bound, while a restricted decision diagram provides a primal bound. For bilevel optimization problems, however, relaxing the follower's problem does not yield a valid lower bound because of the bilevel structure (in the same way as relaxing the follower's integrality constraints does not produce a valid relaxation [34]). Instead, restricted decision diagrams will help us achieve valid dual bounds. This is illustrated in the following example.

Example 3 We continue Example 2. Figure 4 illustrates two approximate decision diagrams for the running example by imposing the width of the compiled diagram to be at most 3. Figure 4a shows a relaxed decision diagram in which now $y = 0$ becomes a possible follower reaction to the leader's decisions $x \in \{0, 1\}$. This modification implies that the optimal solution to model (8) now becomes $(x, y) = (2, 0)$, with an optimal value of -2, which is not a valid lower bound. Figure 4b depicts a restricted decision diagram in which now solutions $(x, y) \in \{(1, 1), (2, 1)\}$ are not encoded. As a result, the new optimal solution to model (8) corresponds to $(x, y) = (1, 2)$, with an optimal value of -5, which is a valid lower bound. □

The purpose of the approximate decision diagram is to encode upper bounds on the value function $\phi(x)$ for a subset of possible leader solutions x. Indeed, any restricted diagram that encodes a subset of r-t paths fulfills this purpose: For the leader solutions x that are encoded, the restriction will provide an upper bound on $\phi(x)$, as not all follower responses to x may be encoded.

Furthermore, for leader solutions that are not encoded we will not enforce an upper bound on the corresponding value function.
Formally, let $\mathcal{D}^{\mathsf{R}}$ denote a restricted diagram induced by $(\mathcal{N}^{\mathsf{R}}, \mathcal{A}^{\mathsf{R}})$, where $\mathcal{N}^{\mathsf{R}} \subseteq \mathcal{N}$ and $\mathcal{A}^{\mathsf{R}} \subseteq \mathcal{A}$. Furthermore, let $\mathcal{A}^{\mathsf{R}0}$ and $\mathcal{A}^{\mathsf{R}1}$ be the leader 0-arcs and 1-arcs in $\mathcal{A}^{\mathsf{R}}$, respectively. The following proposition states how to obtain dual bounds for the optimal value of problem 1 by compiling smaller restricted decision diagrams, where $P\left(\mathcal{D}^{\mathsf{R}}\right)$ is defined exactly as $P(\mathcal{D})$ but over the restricted decision diagram $\mathcal{D}^{\mathsf{R}}$ instead of an exact decision diagram $\mathcal{D}$.

Proposition 5 *The optimal value of*

$$\min\ c_L^\top x + c_F^\top y \tag{17a}$$
$$\text{s.t. } (x,y) \in H \tag{17b}$$
$$d^\top y \leqslant g^R(x,\xi) \tag{17c}$$
$$\xi = (\pi, \lambda, \beta) \in P\left(\mathcal{D}^R\right) \tag{17d}$$

where

$$g^R(x,\xi) = \pi_r - \sum_{a\in\mathcal{A}^{R0}} \left(1 - x_{i(a)}\right)\lambda_a - \sum_{a\in\mathcal{A}^{R1}} x_{i(a)}\beta_a,$$

provides a valid lower bound on the optimal value of the integer bilevel program (1).

Proof We prove that any feasible solution to model (8) maps to a feasible solution to model (17), concluding that model (17) corresponds to a relaxation of model (8), which in turn is an exact reformulation of problem (1) by Proposition 2.
Let $(\widehat{x}, \widehat{y}, \widehat{\xi})$ be a feasible solution to model (8). Consider a vector $\xi' = (\pi', \lambda', \beta')$ such that

$$\begin{aligned} \lambda'_a &= \widehat{\lambda}_a && \forall a \in \mathcal{A}^{\mathsf{R}0} \\ \beta'_a &= \widehat{\beta}_a && \forall a \in \mathcal{A}^{\mathsf{R}1} \\ \pi'_u &= \widehat{\pi}_u && \forall u \in \mathcal{N}^{\mathsf{R}}. \end{aligned}$$

By construction, ξ' satisfies (17d). Moreover, since $\mathcal{A}^{\mathsf{R}0} \subseteq \mathcal{A}^0$, $\mathcal{A}^{\mathsf{R}1} \subseteq \mathcal{A}^1$, and $\widehat{\lambda}, \widehat{\beta} \geqslant 0$, we have

$$\sum_{a\in\mathcal{A}^{\mathsf{R}0}} \left(1 - \widehat{x}_{i(a)}\right)\lambda'_a \leqslant \sum_{a\in\mathcal{A}^0} \left(1 - \widehat{x}_{i(a)}\right)\widehat{\lambda}_a,$$
$$\sum_{a\in\mathcal{A}^{\mathsf{R}1}} \widehat{x}_{i(a)}\beta'_a \leqslant \sum_{a\in\mathcal{A}^1} \widehat{x}_{i(a)}\widehat{\beta}_a.$$

Therefore, $(\widehat{x}, \widehat{y}, \xi')$ satisfies (17c). Since $(\widehat{x}, \widehat{y}) \in H$, we conclude that $(\widehat{x}, \widehat{y}, \xi')$ is a feasible solution to model (17), implying that model (17) is a relaxation of problem (1) as the objective functions are the same. □

We remark that constraint (17c) can be linearized according to Propositions 3 and 4.

4.2 Algorithm for Compiling a Restricted Diagram

Algorithm 1 presents a generic procedure to create a restricted decision diagram with maximum width W. It starts by initializing the diagram with a root and a sink node, and creates $n_F + n_L + 1$ queues, each associated with a layer. Then, it builds the diagram layer by layer, starting from the root node. At each layer, it selects an unvisited node u and tries to create two new nodes, expanding the associated encoded solutions with values $\nu = 0$ and 1. To create a new node, it first verifies whether the partial evaluation p^u of $Cx + Dy$ up to node u does not lead to infeasibility of the constraints. This is done by calling function notInfeasible (line 14), which determines for each constraint whether a feasible completion exists, by evaluating the minimum contribution of each of the remaining variables in the constraint (this is also known as bounds propagation). If setting value ν does not detect any infeasible constraint, notInfeasible returns True and a new node v is created. The function addNode$(v, Q_{\ell+1})$ (line 19) first determines whether v already exists in the queue $Q_{\ell+1}$ associated with the next layer, in which case v is merged with that node. Otherwise, v is added as a new node to $Q_{\ell+1}$. Lastly, the new arc (u, v) is built for the follower or leader case (lines 20-24). Finally, on line 25, the algorithm checks whether the length of a queue surpasses the maximum width W. If so, it reduces the queue by only keeping the W nodes with the best follower objective function values, discarding the remaining ones and their incoming arcs.

Proposition 6 *Algorithm 1 compiles a restricted diagram of size $O((n_F + n_L)W)$ in $O(m(n_F + n_L))$ time, where m is the number of constraints in the follower's problem.*

Proof By construction, the restricted diagram has size $O((n_F + n_L)W)$. Regarding the time complexity, the evaluation of notInfeasible in line 14 takes $O(m)$ steps for m constraints for each node in a layer. Because each node has at most two outgoing arcs, the maximum number of nodes in any layer is $2W$, for constant W. For $n_F + n_L + 1$ layers this gives $O(m(n_F + n_L))$ steps in total. All other steps in lines 7-24 take constant time. The sorting in line 26 takes $O(W \log W)$ time as there are at most $2W$ nodes in each layer. Because W is a constant, the overall time complexity follows. □

We remark that in certain cases, our relaxation can readily prove optimality for the original bilevel problem.

Proposition 7 *Consider a solution $(\widehat{x}, \widehat{y})$ obtained by solving model (17). If $d^\top \widehat{y} \leqslant \phi(\widehat{x})$, then $(\widehat{x}, \widehat{y})$ is optimal for problem (1).*

Proof By Proposition 5, $c_L^\top \widehat{x} + c_F^\top \widehat{y}$ is a lower bound on the optimal value. If $d^\top \widehat{y} \leqslant \phi(\widehat{x})$, then $(\widehat{x}, \widehat{y})$ is a bilevel-feasible solution and $c_L^\top \widehat{x} + c_F^\top \widehat{y}$ is also an upper bound on the optimal value. □

Algorithm 1: Procedure to generate a restricted decision diagram.

```
Parameters: maximum width $W$
Input: binary bilevel programming instance $I$
initialize diagram $\mathcal{D} = (\mathcal{N}, \mathcal{A}) \leftarrow (\{r, t\}, \varnothing)$ with root $r$ and sink $t$
initialize queues $Q_1 \leftarrow \{r\},\ Q_{n_F+n_L+1} \leftarrow \{t\},\ Q_\ell \leftarrow \varnothing \quad \forall \ell = 2, \ldots, n_F + n_L$
for $\ell = 1, \ldots, n_F + n_L$ do
    while $Q_\ell \neq \varnothing$ do
        select node $u \leftarrow Q_\ell[0]$ and remove it from $Q_\ell$
        set $p^u \leftarrow$ partial evaluation of $Cx + Dy$ at node $u$
        for $\nu = 0, 1$ do
            if $\ell \leqslant n_F$ then
                set $\hat{p} \leftarrow p^u + D_{i(\ell)} \cdot \nu$
            else
                set $\hat{p} \leftarrow p^u + C_{i(\ell)} \cdot \nu$
            if notInfeasible$(\ell, \hat{p}) = True$ then
                create node $v$
                if $\ell = n_F + n_L$ then
                    set $v \leftarrow t$                                  // Reached last layer
                else
                    addNode$(v, Q_{\ell+1})$
                if $\ell \leqslant n_F$ then
                    create follower $\nu$-arc $(u, v)$ with cost $\nu \cdot d_{i(\ell)}$
                else
                    create leader $\nu$-arc $(u, v)$ with capacity
                      $\nu \cdot x_{i(\ell)} + (1 - \nu) \cdot (1 - x_{i(\ell)})$
                add arc $(u, v)$ to $\mathcal{A}$
    if $|Q_{\ell+1}| > W$ then
        sort nodes in $Q_{\ell+1}$ by increasing follower objective function value
        discard all but the first $W$ nodes from $Q_{\ell+1}$
        remove the associated arcs from $\mathcal{A}$
    add all nodes from $Q_{\ell+1}$ to $\mathcal{N}_{\ell+1}$
return $\mathcal{D}$
```

4.3 An Exact Iterative Method

We propose an iterative procedure to extend the computation of the dual bound to an exact method for solving the integer bilevel programming problem (1), presented in Algorithm 2. We assume that an initial restricted decision diagram $\mathcal{D}^{\mathsf{R}}$ has been compiled. The algorithm iteratively solves relaxation (17) and checks whether the follower solution is optimal. If not, it adds the leader solution with the optimal follower response to the relaxation by incorporating the associated follower dual variables and constraints, and repeats these steps until the optimum is found.

Proposition 8 *Algorithm 2 returns the optimal solution to the integer bilevel programming problem* (1) *in a finite number of iterations, assuming that the leader and follower problems have a finite number of integer solutions.*

The proof follows immediately because, in the worst case, Algorithm 2 may enumerate all possible leader solutions with all associated optimal follower responses. It also shows that the restricted decision diagram can potentially

Algorithm 2: Iterative compilation procedure.

```
1 Input: restricted decision diagram $\mathcal{D}^{\mathsf{R}}$
2 while True do
3     solve model (17) to obtain solution $(\hat{x}, \hat{y})$ and dual bound $\hat{z}$
4     solve the optimal follower's response to $\hat{x}$ to obtain solution $y'$ and optimal
        objective value $z'$
5     if $\hat{z} = z'$ then
6         terminate and return the optimal solution $(\hat{x}, \hat{y})$
7     else
8         add the path encoding $(y', \hat{x})$ to $\mathcal{D}^{\mathsf{R}}$
```

accelerate this process if it represents the "right" subset of leader and follower solutions.
As an alternative to explicitly adding the leader path and follower solution to the diagram in each iteration, we can equivalently represent this using a "no-good" cut formulation encoding the paths, similar to the approach by Lozano and Smith [31]. By doing so, we avoid introducing more dual variables. In our computational analysis, adding the no-good cuts formulation to a fixed initial decision diagram generally yielded more efficient solving times than explicitly adding the paths to the diagram.

5 Layer Contraction

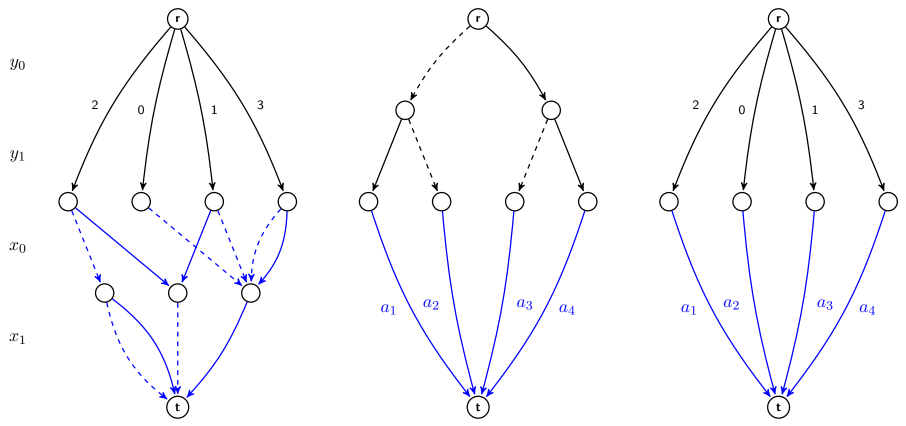
Even after compiling a restricted decision diagram, the resulting relaxation (17) can become hard to solve. We now propose two contraction procedures that help diminish this burden, substantially reducing the number of variables and constraints without losing any relevant information.

Follower layers contraction Variables π in model (8) enforce the selection of a feasible shortest r-u path entering any node $u \in \mathcal{N}_{n_F+1}$. This implies that any path reaching a node $u \in \mathcal{N}_{n_F+1}$ with a greater length than the shortest r-u path becomes redundant, *i.e.*, can be removed from the diagram without loss of optimality. Thus, after compiling layer $\mathcal{N}_{n_F+1}$, we can collapse all the follower layers but layer 1 into a single layer. This way, any node $u \in \mathcal{N}_{n_F+1}$ will have only one incoming arc (r, u), leaving the root node r with a length equivalent to the shortest r-u path, and infinite capacity. Figure 5a exemplifies this contraction.

Leader layers contraction Consider any node $u \in \mathcal{N}_{n_F+1}$ and let p^u be the partial evaluation of $Cx + Dy$ according to Algorithm 1 (line 8). Observe that, given $x \in \{0,1\}^{n_L}$, there will be at least one u-t path with positive capacity if and only if $Cx \leqslant b - p^u$. Moreover, no matter what u-t path is considered, the follower's cost remains defined by the length of the follower arcs. Therefore, without loss of optimality, we can replace all u-t paths with an arc (u, t) with length 0 and capacity $\mathbb{1}(Cx \leqslant b - p^u)$, where $\mathbb{1}(\cdot)$ denotes the indicator function. Doing this for every node $u \in \mathcal{N}_{n_F+1}$ corresponds to contracting all leader layers into one layer, as illustrated in Figure 5b. In this example, the capacities of new arcs are given by

$$a_1 = \mathbb{1}\begin{pmatrix} x \geqslant 0, \\ 3x \leqslant 8 \end{pmatrix}, a_2 = \mathbb{1}\begin{pmatrix} x \geqslant 2, \\ 3x \leqslant 6 \end{pmatrix}, a_3 = \mathbb{1}\begin{pmatrix} x \geqslant 1, \\ 3x \leqslant 7 \end{pmatrix}, a_4 = \mathbb{1}\begin{pmatrix} x \geqslant 2, \\ 3x \leqslant 9 \end{pmatrix}.$$

We remark that, in the case of using restricted diagrams, this contraction also helps yield new bounds for x values that may not have been initially encoded. Moreover, for each node $u \in \mathcal{N}_{n_F+1}$, function $\mathbb{1}(Cx \leqslant b - p^u)$ can be linearized through additional variables and big-M constraints. Finally, observe that, since both contractions are applied to separate sets of layers, we can execute both without loss of optimality, as illustrated in Figure 5c.



a. Follower layers contraction b. Leader layers contraction c. Both contractions

Fig. 5: Contractions applied to diagram in Figure 3b. a) Contraction of the follower layers. b) Contraction of the leader layers. c) Contraction of the follower and leader layers.

6 Extension to the Pessimistic Setting

We now consider the integer linear version of the pessimistic bilevel program defined in [31,42]. Similar to the procedure exposed in Section 3, based on a value function reformulation, we show how to extend model (8) to obtain a relaxation for this setting. Recall that, in the pessimistic case, the follower aims to maximize the leader's objective function, while keeping the solution bilevel-feasible. Similar to model (1), the pessimistic version can be reformulated using the value function ϕ as

$$\min_x \left\{ \max_y \left\{ c_L^\top x + c_F^\top y \right\} : \ (x, y) \in H, \ d^\top y \leqslant \phi(x) \right\}. \tag{18}$$

Observe that, given a leader's decision $\widehat{x}$, if there exists a vector $\widehat{y}$ making solution $(\widehat{x}, \widehat{y})$ infeasible for the leader's problem, then the follower will select such vector, achieving $c_L^\top \widehat{x} + c_F^\top \widehat{y} = +\infty$. Furthermore, models (1) and (18) share the same HPR H. Hence, any optimal solution to model (18) is also feasible to model (1), but potentially suboptimal. These two key ideas allow the use of model (8) as a base to obtain an enhanced relaxation of model (18). In a similar fashion to [31,42], we start from the value function reformulation of (18), which corresponds to

$$
\begin{aligned}
\min\ & c_L^\top x + c_F^\top y && && \text{(19a)}\\
\text{s.t.}\ & (x,y) \in H && && \text{(19b)}\\
& d^\top y \leqslant \phi(x) && && \text{(19c)}\\
& c_F^\top y \geqslant \eta(x, \phi(x)) && && \text{(19d)}\\
& Ax + B\widehat{y} \leqslant a && \forall \widehat{y} \in R(x). && \text{(19e)}
\end{aligned}
$$

where $\eta(x,z) = \max\{c_F^\top y : y \in \mathbb{Y}(x),\ d^\top y \leqslant z\}$. Constraint (19d) forces a feasible solution (x,y) to maximize $c_F^\top y$ over $\mathbb{Y}(x)$, and constraints (19e) force any leader decision vector x to be such that there is no feasible follower reaction y making (x,y) infeasible for the leader's problem.

Note that we can now reformulate model (19) by adding new variables and no-good cuts as in [31] to our decision diagram-based reformulation (8). However, we can also further exploit the compiled decision diagram $\mathcal{D}$ to obtain a stronger relaxation of (19). To do so, we reformulate the program obtained from (19) by relaxing (19e), *i.e.*, model

$$
\begin{aligned}
\min\ & c_L^\top x + c_F^\top y && \text{(20a)}\\
\text{s.t.}\ & (x,y) \in H && \text{(20b)}\\
& d^\top y \leqslant \phi(x) && \text{(20c)}\\
& c_F^\top y \geqslant \eta(x, \phi(x)). && \text{(20d)}
\end{aligned}
$$

We now use the components of $\mathcal{D} = (\mathcal{N}, \mathcal{A}, \nu, l)$ to model constraint (20d). For any $u \in \mathcal{N}$, let $\bar{l}_u$ denote the minimum length among all r-u paths in $\mathcal{D}$, and consider the additional arc-length vector l^{P} defined as $l_a^{\mathsf{P}} = \nu_a c_{F_{i(a)}}$ for any follower arc a and $l_a^{\mathsf{P}} = 0$ for any leader arc a. We next show how to use $\mathcal{D}$ to compute η.

Proposition 9 *Consider $\widehat{x} \in \{0,1\}^{n_L}$ such that $\mathbb{Y}(\widehat{x}) \neq \varnothing$ and $\widehat{z} \in \mathbb{R}$. Then,*

$$
\begin{aligned}
\eta(\widehat{x}, \widehat{z}) = \max\ & \sum_{a \in \mathcal{A}} l_a^{P} w_a && && \text{(21a)}\\
\text{s.t.}\ & \text{(4b)-(4d)} && && \text{(21b)}\\
& w_a \leqslant \mathbb{1}\left(\bar{l}_u \leqslant \widehat{z}\right) && u \in \mathcal{N}_{n_F+1},\ a \in \delta^+(u) && \text{(21c)}\\
& w \in \mathbb{R}_+^{|\mathcal{A}|}. && && \text{(21d)}
\end{aligned}
$$

Proof Similar to the proof of Proposition 1. In this case, due to constraints (21c), we also discard any path encoding a vector $(\widehat{x}, y)$ with a follower portion $y \in \mathbb{Y}(\widehat{x})$ violating $d^\top y \leqslant \widehat{z}$ as a feasible solution. Thus, given an optimal solution w^* to model (21), we have

$$\sum_{a \in \mathcal{A}} l_a^{\mathsf{P}} w_a^* = c_F^\top y^* = \max\left\{c_F^\top y :\ d^\top y \leqslant \widehat{z},\ y \in \mathbb{Y}(\widehat{x})\right\} = \eta(\widehat{x}, \widehat{z}),$$

where y^* is the follower portion of the binary vector encoded by w^*. □

Following the same procedure as in Section 3, we can use LP duality to compute $\eta(\widehat{x}, \widehat{z})$ by solving

$$\begin{aligned}
\min\ & g^{\mathsf{P}}(\widehat{x}, \widehat{z}, \zeta) && && \text{(22a)}\\
\text{s.t.}\ & \pi_u - \pi_v \geqslant l_{uv}^{\mathsf{P}} && (u,v) = a \in \mathcal{A} \backslash (\mathcal{A}^0 \cup \mathcal{A}^1) && \text{(22b)}\\
& \pi_u - \pi_v + \lambda_{uv} \geqslant l_{uv}^{\mathsf{P}} && (u,v) = a \in \mathcal{A}^0 \setminus \mathcal{A}_{n_F+1} && \text{(22c)}\\
& \pi_u - \pi_v + \beta_{uv} \geqslant l_{uv}^{\mathsf{P}} && (u,v) = a \in \mathcal{A}^1 \setminus \mathcal{A}_{n_F+1} && \text{(22d)}\\
& \pi_u - \pi_v + \lambda_{uv} + \gamma_{uv} \geqslant l_{uv}^{\mathsf{P}} && (u,v) = a \in \mathcal{A}^0 \cap \mathcal{A}_{n_F+1} && \text{(22e)}\\
& \pi_u - \pi_v + \beta_{uv} + \gamma_{uv} \geqslant l_{uv}^{\mathsf{P}} && (u,v) = a \in \mathcal{A}^1 \cap \mathcal{A}_{n_F+1} && \text{(22f)}\\
& \lambda \in \mathbb{R}_+^{|\mathcal{A}^0|},\ \beta \in \mathbb{R}_+^{|\mathcal{A}^1|} && && \text{(22g)}\\
& \gamma \in \mathbb{R}_+^{|\mathcal{A}_{n_F+1}|} && && \text{(22h)}\\
& \zeta = (\pi, \lambda, \beta, \gamma), && && \text{(22i)}
\end{aligned}$$

where

$$\begin{aligned}
g^{\mathsf{P}}(x, z, \zeta) = \pi_r &+ \sum_{a \in \mathcal{A}^0} \left(1 - x_{i(a)}\right) \lambda_a + \sum_{a \in \mathcal{A}^1} x_{i(a)} \beta_a \\
&+ \sum_{u \in \mathcal{N}_{n_F+1}} \mathbb{1}\left(\bar{l}_u \leqslant z\right) \sum_{a \in \delta^+(u)} \gamma_a.
\end{aligned}$$

Let $P^{\mathsf{P}}(\mathcal{D})$ denote the feasible solution set of model (22). We next show that constraint (20d) can be replaced with the system

$$\begin{cases} c_F^\top y \geqslant g^{\mathsf{P}}(x, d^\top y, \zeta), \\ \zeta = (\pi, \lambda, \beta, \gamma) \in P^{\mathsf{P}}(\mathcal{D}) \end{cases}.$$

Proposition 10 *Model*

$$\begin{aligned}
\min\ & c_L^\top x + c_F^\top y && \text{(23a)}\\
\text{s.t.}\ & (x, y) \in H && \text{(23b)}\\
& d^\top y \leqslant g(x, \xi) && \text{(23c)}\\
& \xi = (\pi, \lambda, \beta) \in P(\mathcal{D}) && \text{(23d)}\\
& c_F^\top y \geqslant g^P(x, d^\top y, \zeta) && \text{(23e)}\\
& \zeta = \left(\pi^P, \lambda^P, \beta^P\right) \in P\left(\mathcal{D}^P\right). && \text{(23f)}
\end{aligned}$$

is an exact reformulation of the relaxed pessimistic integer bilevel program (20)*.*

Proof From the proof of Proposition 2, we know that, for any $(x, y) \in H$, constraints (23c)-(23d) force $\xi \in P(\mathcal{D})$ to satisfy $d^\top y \leqslant g(x, \xi) \leqslant \phi(x)$. Therefore, by the definition of ϕ, we have $d^\top y = \phi(x)$ for all $(x, y) \in H$. We now follow the same logic to interpret constraint (23e).
We first consider a feasible solution $(\widehat{x}, \widehat{y}, \widehat{\xi}, \widehat{\zeta})$ to model (23) and show that $(\widehat{x}, \widehat{y})$ is feasible to model (20). For any real value $z \in \mathbb{R}$, by Proposition 9 and weak duality, we have

$$g^{\mathsf{P}}\left(\widehat{x}, z, \widehat{\zeta}\right) \geqslant \eta\left(\widehat{x}, z\right).$$

Therefore, constraints (23e)-(23f) ensure that

$$\begin{aligned} c_F^\top \widehat{y} &\geqslant g^{\mathsf{P}}\left(\widehat{x}, d^\top \widehat{y}, \widehat{\zeta}\right) \\ &= g^{\mathsf{P}}\left(\widehat{x}, \phi(\widehat{x}), \widehat{\zeta}\right) \\ &\geqslant \eta\left(\widehat{x}, \phi(\widehat{x})\right), \end{aligned}$$

and thus $(\widehat{x}, \widehat{y})$ satisfies (20d). We conclude that $(\widehat{x}, \widehat{y})$ is feasible to model (20).
We now show that for any solution $(\widehat{x}, \widehat{y})$ feasible to model (20), there exist vectors ξ and ζ such that $(\widehat{x}, \widehat{y}, \xi, \zeta)$ is feasible to model (23). Let $\widehat{w}$ be the flow vector associated with the unique path in the diagram encoding $(\widehat{x}, \widehat{y})$. By construction, we have that $\widehat{w}$ is feasible to model (21) when $\widehat{z} \geqslant d^\top \widehat{y}$ and

$$\sum_{a \in \mathcal{A}} l_a^{\mathsf{P}} \widehat{w}_a = c_F^\top \widehat{y}.$$

Furthermore, constraint (23e) and Proposition 9 ensure that $\widehat{w}$ is an optimal solution to model (21) with arc capacities induced by $(\widehat{x}, d^\top \widehat{y})$, *i.e.*,

$$\sum_{a \in \mathcal{A}} l_a^{\mathsf{P}} \widehat{w}_a = \eta\left(\widehat{x}, d^\top \widehat{y}\right).$$

Now, let ξ' be an optimal solution to model (7) and ζ' be an optimal solution to model (22). By definition, ξ' satisfies (8d) and, by strong duality, satisfies

$$\phi(\widehat{x}) = g(\widehat{x}, \xi').$$

Furthermore, ζ' satisfies (23f) and, by strong duality, satisfies

$$\eta\left(\widehat{x}, d^\top \widehat{y}\right) = g^{\mathsf{P}}\left(\widehat{x}, d^\top \widehat{y}, \zeta'\right).$$

Since $d^\top \widehat{y} = \phi(\widehat{x})$, we conclude that $(\widehat{x}, \widehat{y}, \xi', \zeta')$ is feasible to model (23). As the objective functions for models (20) and (23) are the same, we conclude that both models are equivalent. □

We remark that constraint (23e) can be linearized in a similar fashion to constraint (23c), *i.e.*, using Propositions 3 and 4. Furthermore, we can also obtain a valid bound on the optimal value of model (23) by compiling a restriction of $\mathcal{D}$.

Proposition 11 *The optimal value of*

$$
\begin{aligned}
&\min\ c_L^\top x + c_F^\top y && \text{(24a)}\\
&\text{s.t. } (x, y) \in H && \text{(24b)}\\
&\quad d^\top y \leqslant g^{R}(x, \xi) && \text{(24c)}\\
&\quad \xi = (\pi, \lambda, \beta) \in P\left(\mathcal{D}^{R}\right) && \text{(24d)}\\
&\quad c_F^\top y \geqslant g^{PR}\left(x, d^\top y, \zeta\right) && \text{(24e)}\\
&\quad \zeta = \left(\pi^{P}, \lambda^{P}, \beta^{P}, \gamma^{P}\right) \in P^{P}\left(\mathcal{D}^{R}\right), && \text{(24f)}
\end{aligned}
$$

where

$$
\begin{aligned}
g^{PR}(x, z, \zeta) = \pi_r &+ \sum_{a \in \mathcal{A}^{R0}} \left(1 - x_{i(a)}\right) \lambda_a + \sum_{a \in \mathcal{A}^{R1}} x_{i(a)} \beta_a \\
&+ \sum_{u \in \mathcal{N}_{n_F+1}} \mathbb{1}\left(\bar{l}_u \leqslant z\right) \sum_{a \in \delta^+(u)} \gamma_a,
\end{aligned}
$$

provides a valid lower bound on the optimal value of model (23)*.*

Proof As in the proof of Proposition 5, we show that any feasible solution to model (23) maps to a feasible solution to model (24).
Let $(\widehat{x}, \widehat{y}, \widehat{\xi}, \widehat{\zeta})$ be a feasible solution to model (23). Consider vectors $\xi' = (\pi', \lambda', \beta')$ and $\zeta' = (\pi^{\mathsf{P}'}, \lambda^{\mathsf{P}'}, \beta^{\mathsf{P}'}, \gamma^{\mathsf{P}'})$ such that

$$
\begin{aligned}
&\lambda'_a = \widehat{\lambda}_a,\ \lambda^{\mathsf{P}'}_a = \widehat{\lambda}^{\mathsf{P}}_a && \forall a \in \mathcal{A}^{\mathsf{R0}}\\
&\beta'_a = \widehat{\beta}_a,\ \beta^{\mathsf{P}'}_a = \widehat{\beta}^{\mathsf{P}}_a && \forall a \in \mathcal{A}^{\mathsf{R1}}\\
&\pi'_u = \widehat{\pi}_u,\ \pi^{\mathsf{P}'}_u = \widehat{\pi}^{\mathsf{P}}_u && \forall a \in \mathcal{N}^{\mathsf{R}}\\
&\gamma^{\mathsf{P}'} = \widehat{\gamma}^{\mathsf{P}}_a && \forall a \in \mathcal{A}_{n_F+1}\,.
\end{aligned}
$$

By the proof of Proposition 5, ξ' satisfies (24d) and $(\widehat{x}, \widehat{y}, \xi')$ satisfies (24c). By construction, ζ' satisfies (24f). Moreover, since $\mathcal{A}^{\mathsf{R0}} \subseteq \mathcal{A}^0, \mathcal{A}^{\mathsf{R1}} \subseteq \mathcal{A}^1, \mathcal{N}^{\mathsf{R}}_{n_F+1} \subseteq \mathcal{N}_{n_F+1}$, and $\widehat{\lambda}^{\mathsf{P}}, \widehat{\beta}^{\mathsf{P}}, \widehat{\gamma}^{\mathsf{P}} \geqslant 0$, we have

$$
\begin{aligned}
&\sum_{a \in \mathcal{A}^{\mathsf{R0}}} \left(1 - \widehat{x}_{i(a)}\right) \lambda^{\mathsf{P}'}_a \leqslant \sum_{a \in \mathcal{A}^0} \left(1 - \widehat{x}_{i(a)}\right) \widehat{\lambda}^{\mathsf{P}}_a,\\
&\sum_{a \in \mathcal{A}^{\mathsf{R1}}} \widehat{x}_{i(a)} \beta^{\mathsf{P}'}_a \leqslant \sum_{a \in \mathcal{A}^1} \widehat{x}_{i(a)} \widehat{\beta}^{\mathsf{P}}_a,\\
&\sum_{u \in \mathcal{N}^{\mathsf{R}}_{n_F+1}} \mathbb{1}\left(\bar{l}_u \leqslant d^\top \widehat{y}\right) \sum_{a \in \delta^+(u)} \gamma^{\mathsf{P}'}_a \leqslant \sum_{u \in \mathcal{N}_{n_F+1}} \mathbb{1}\left(\bar{l}_u \leqslant d^\top \widehat{y}\right) \sum_{a \in \delta^+(u)} \widehat{\gamma}^{\mathsf{P}}_a.
\end{aligned}
$$

Therefore, $(\widehat{x}, \widehat{y}, \zeta')$ satisfies (24e). Since $(\widehat{x}, \widehat{y}) \in H$, we conclude that $(\widehat{x}, \widehat{y}, \xi', \zeta')$ is a feasible solution to model (24), implying that model (24) is a relaxation of model (23), as the objective function of models (24) and (23) are the same. □

We conclude this section by remarking that several works from the literature consider special cases of bilevel problems without linking constraints, i.e., $A = B = 0$, or in which the follower variables do not affect the leader's constraints, i.e., $B = 0$. For these cases, model (20) provides an exact reformulation of the pessimistic problem instead of a relaxation, which by Proposition 10 implies that our DD-based model (23) also provides an exact single-level reformulation of the pessimistic problem that can be readily linearized as described above.

7 Case Study: Bilevel Independent Set with Knapsack Constraints

We will see in the experimental section that our generic approach can be effective for some problem classes. However, if we use the standard linear encoding of problem (1), the decision diagrams do not have the ability to exploit problem-specific structures, which can often drastically improve their representational power. For that reason, we next apply our generic method to a case study that possesses a specific combinatorial structure allowing us to further streamline our method. Specifically, we introduce the Bilevel Independent Set Problem with Knapsack Constraints (BISP-KC).

In the BISP-KC, we are given a graph $G = (V, E)$ with vertex set V and edge set E. The leader solves a knapsack problem while the follower solves a maximum-reward independent set problem with an additional knapsack-type constraint depending on the leader's decisions. This problem is motivated by social network applications, where a follower entity tries to maximize its influence on the network by selecting compatible "influencers", and a regulator entity, acting as a leader, aims to maximize social welfare by altering or interdicting the compatibility of the "influencers", subject to a budget. In this case, the follower's knapsack constraint generalizes the interdiction effect of the leader's decisions.

The mathematical formulation has the form of the generic model (1), where both the leader and the follower have a binary variable associated with each node in V, *i.e.*, $n_L = n_F = |V|$. The leader's problem knapsack constraint has "weight" coefficients (w_L, w_F) and "capacity" scalar b_L, whereas for the follower's problem knapsack constraint these are (ω_L, ω_F) and b_F.

$$\min \; c_L^\top x + c_F^\top y \tag{25a}$$

$$\text{s.t.} \; w_L^\top x + w_F^\top y \leqslant b_L \tag{25b}$$

$$x \in \{0,1\}^{n_L} \tag{25c}$$

$$y \in \operatorname{argmin}_{\bar{y} \in \mathbb{Y}(x)} \left\{ d^\top \bar{y} \right\}, \tag{25d}$$

where

$$\mathbb{Y}(x) := \left\{ y \in \{0,1\}^{n_F} : \begin{array}{l} \omega_F^\top y \leqslant b_F - \omega_L^\top x \\ y_i + y_j \leqslant 1 \quad \forall (i,j) \in E \end{array} \right\}.$$

We assume $c_L, w_L, \omega_L \in \mathbb{R}^{n_L}$, $c_F, w_F, \omega_F \in \mathbb{R}^{n_F}$, and $b_L, b_F \in \mathbb{R}$. To maximize the follower's objective function, we keep the argmin function for consistency and encode the reward coefficients d as negative numbers.
To compile the decision diagram, we could apply Algorithm 1 directly on the linear representation of model (25). However, a more efficient representation for the follower's portion uses the compilation proposed in [7], in which each node of the decision diagram represents the set of vertices S that can still be added to the independent set. In addition, we represent the remaining capacity k of the follower's problem at that node. Given a node u with state $p^u = (S, k)$ in layer Q_ℓ representing decision $y_{i(\ell)}$, function $\mathsf{notInfeasible}$ for $\nu = 1$ returns false if $i(\ell) \notin S$ or if $\omega_{F_{i(\ell)}} + k > b_F$. For $\nu = 0$, it always returns true. The compilation of the leader portion of the decision diagram follows the generic implementation of the algorithm.

8 Experimental Results

We next present a computational evaluation of our approach. We first consider generic bilevel integer programming problems from the BOBILib benchmark instance set [38]. We then present results for the BISP-KC as an example for which the decision diagram approach can leverage the problem structure. We compare our exact iterative compilation method (Section 4.3) to the branch-and-cut approaches from [19] and [36] which represent the state-of-the-art for solving integer bilevel programming problems. We will refer to our iterative method as `IDDR`, and to the branch-and-cut-approaches as `FLMS` and `MibS`, respectively. Both `FLMS` and `MibS` are compiled using their latest version (`MibS`'s is 1.2.1 and `FLMS` has a unique version) and executed using their default settings.
For the construction of the restricted diagrams, we use Algorithm 1 with a maximum width W of 25 for the bilevel MIPLIB instances, and 500, 1000, and 2000 for the BISP-KC instances (these numbers were determined to strike a reasonable balance between bound quality and solving time). We employ the result from [19, Thm. 2] to pre-process the follower's y values and reduce the size of model (17). All decision diagrams apply the layer contraction operations for the leader and the follower as described in Section 5. The iterative compilation method (Section 4.3) uses the no-good encoding of the leader solution and optimal follower solutions at each iteration.
Our algorithms are implemented in Python 3.10 and all computations are executed on a 12–core Intel(R) Xeon(R) Gold 6248R CPU @ 3.00GHz running Linux Ubuntu. All optimization problems are solved using CPLEX 22 or Gurobi 10 (when indicated) with a 1-hour time limit.

8.1 Generic Integer Bilevel Programming: BOBILib Benchmark

We first study generic integer bilevel programming instances from the BOBILib benchmark set [38]. We consider instances that only contain binary

Table 1: Results for the BOBILib instances. For each instance, we report the best known solution (`BKS`), and the total solving time in seconds, upper bound (UB), and optimality gap (%) for the branch-and-cut methods `FLMS` [19], `MibS` [36], and our method `IDDR`. Bold entries indicate new/improved bounds, or newly closed open instances (0% gap).

Instance	`BKS`	`FLMS`			`MibS`			`IDDR`		
		Time (s)	UB	Gap (%)	Time (s)	UB	Gap (%)	Time (s)	UB	Gap (%)
stein27-0.1	18	0	18	0.00	1	18	0.00	1	18	0.00
stein27-0.5	19	0	19	0.00	0	19	0.00	0	19	0.00
stein27-0.9	24	0	24	0.00	0	24	0.00	1	24	0.00
stein45-0.1	30	7	30	0.00	16	30	0.00	11	30	0.00
stein45-0.5	32	0	32	0.00	0	32	0.00	11	32	0.00
stein45-0.9	40	0	40	0.00	2	40	0.00	14	40	0.00
enigma-0.1	0	1	0	0.00	1	0	0.00	0	0	0.00
enigma-0.5	0	8	0	0.00	1	0	0.00	1	0	0.00
enigma-0.9	0	593	0	0.00	1	0	0.00	3	0	0.00
lseu-0.1	1,120	0	1,120	0.00	2	1,120	0.00	1	1,120	0.00
lseu-0.5	2,263	3,600	2,313	7.63	3,600	2,263	12.15	3,600	2,400	46.71
lseu-0.9	5,838	95	5,838	0.00	3,600	5,838	19.01	5	5,838	0.00
p0033-0.1	3,089	0	3,089	0.00	0	3,089	0.00	0	3,089	0.00
p0033-0.5	3,095	0	3,095	0.00	0	3,095	0.00	0	3,095	0.00
p0033-0.9	4,679	0	4,679	0.00	1	4,679	0.00	0	4,679	0.00
p0201-0.1	12,465	3,600	12,465	20.90	3,600	12,615	55.11	3	**12,225**	**0.00**
p0201-0.5	13,635	2,195	13,635	0.00	3,600	13,985	70.07	3,600	13,810	26.43
p0201-0.9	15,025	5	15,025	0.00	3,600	15,025	78.06	10	15,025	0.00
p0282-0.1	260,781	3	260,781	0.00	3,600	261,188	1.05	2	260,781	0.00
p0282-0.5	272,659	3,600	272,659	0.93	3,600	272,914	5.56	34	272,659	0.00
p0282-0.9	614,837	3,600	614,837	28.85	3,600	638,033	139.84	3	614,837	**0.00**
p0548-0.1	11,069	3,600	11,086	16.85	3,600	-	-	3,600	**10,968**	14.55
p0548-0.5	-	3,600	-	-	3,600	-	-	3,600	**22,177**	52.77
p0548-0.9	58,835	3,600	**48,942**	60.25	3,600	-	-	3,600	**48,942**	64.33
p2756-0.1	13,912	3,600	14,550	77.07	3,600	-	-	3,600	14,431	77.70
p2756-0.5	-	3,600	**23,547**	83.05	3,600	-	-	3,600	26,663	85.64
p2756-0.9	33,847	3,600	**33,362**	86.12	3,600	-	-	3,600	38,131	88.36
l152lav-0.1	4,722	7	4,722	0.00	5	4,722	0.00	33	4,722	0.00
l152lav-0.5	4,816	3,600	4,817	1.27	3,600	4,907	2.81	3,600	4,915	3.21
l152lav-0.9	5,066	3,600	5,090	6.37	3,600	5,355	12.18	3,600	5,540	14.19
mod010-0.1	6,554	5	6,554	0.00	1	6,554	0.00	155	6,554	0.00
mod010-0.5	6,684	3,600	**6,678**	1.73	3,600	7,024	6.96	3,600	7,024	6.73
mod010-0.9	7,411	3,600	7,620	13.92	3,600	8,609	30.36	3,600	8,847	25.91
air03-0.1	381,812	3,600	**378,890**	8.96	3,600	386,502	6.62	3,600	403,334	15.26
air03-0.5	504,230	3,600	513,226	31.04	3,600	621,620	70.82	3,600	671,964	48.99
air03-0.9	763,632	3,600	**762,132**	53.18	3,600	883,408	142.33	3,600	885,774	61.07
air04-0.1	56,399	3,600	56,415	0.35	3,600	-	-	3,600	57,754	2.80
air04-0.5	59,721	3,600	60,725	7.45	3,600	-	-	3,600	64,960	13.57
air04-0.9	-	3,600	-	-	3,600	-	-	3,600	**91,270**	38.48
air05-0.1	26,577	1,005	26,577	0.00	3,600	-	-	3,600	29,286	9.85
air05-0.5	33,757	3,600	**32,528**	18.70	3,600	-	-	3,600	50,752	47.97
air05-0.9	40,046	3,600	-	-	3,600	-	-	3,600	60,259	56.09
fast0507-0.1	12,484	1	12,484	0.00	3,600	-	-	422	12,484	0.00
fast0507-0.5	61,439	1	61,439	0.00	3,600	-	-	449	61,439	0.00
fast0507-0.9	109,916	1	109,916	0.00	3,600	-	-	417	109,916	0.00
cap6000-0.1	-1,967,015	802	-1,967,015	0.00	3,600	-1,734,110	29.26	237	-1,967,015	0.00
cap6000-0.5	-	3,600	-	-	3,600	-1,169,820	52.28	3,600	**-1,170,489**	106.92
cap6000-0.9	-	3,600	-	-	3,600	-100,780	95.89	3,600	**-108,130**	2134.40
mitre-0.1	122,190	3,600	-	-	3,600	122,310	6.21	3,600	122,205	5.72
mitre-0.5	147,030	3,600	-	-	3,600	147,045	27.69	3,600	147,030	21.67
mitre-0.9	169,425	3,600	169,470	31.95	3,600	169,470	47.15	3,600	169,470	32.04
nw04-0.1	17,066	1,023	17,066	0.00	3,600	-	-	3,600	17,796	4.45
nw04-0.5	24,828	3,600	26,396	37.08	3,600	49,288	199.10	3,600	39,734	57.04
nw04-0.9	50,540	3,600	**44,026**	59.80	3,600	-	-	3,600	64,666	72.12
seymour-0.1	475	3,600	476	1.33	3,600	-	-	3,600	-	-
seymour-0.5	807	1	807	0.00	3,600	-	-	229	807	0.00
seymour-0.9	1,251	0	1,251	0.00	3,600	-	-	240	1,251	0.00
harp2-0.1	-72,444,149	3,600	-72,111,431	2.45	3,600	-	-	3,600	-67,299,607	9.75
harp2-0.5	-48,776,284	3,600	-	-	3,600	-	-	3,600	-43,700,126	69.08
harp2-0.9	-1,323,015	3,600	-154,337	47,836.26	3,600	-	-	3,600	3,013,774	2538.80

variables; for these, constraints are all present in the follower's problem (*i.e.*, $m_L = 0$), and d is set to $-c_F$. The BOBILib instances were derived from MIPLIB 3.0 instances, and consider a different percentage of follower variables ($n_F = \delta \cdot (n_F + n_L)$ for $\delta \in \{10\%, 50\%, 90\%\}$). Detailed information about the instances can be found in Table A.1 of Appendix A.. We compare our exact iterative method (`IDDR`) with the state-of-the-art branch-and-cut methods by Fischetti, Ljubić, Monaci, and Sinnl [19], denoted `FLMS`, and by Tahernejad, Ralphs, and DeNegre [36], denoted `MibS`. Because the results in [19] consider CPLEX 12 as the default MIP solver, we first update their code and execute `FLMS` using CPLEX 22. Table B.2 in Appendix B. presents the comparison of running `FLMS` with CPLEX 12 and CPLEX 22 on the benchmark set. This update allows `FLMS` to find an improved upper bound for eight instances, including instance p2756-0.5, for which no known upper bound has been reported in BOBILib yet.

Table 1 compares the methods `IDDR`, `FLMS`, and `MibS` on the BOBILib benchmark instances, all running CPLEX 22 as the default MIP solver. For each instance, we indicate the best-known solution (`BKS`), and we report for each solution method the total solving time in seconds, upper bound (UB), and optimality gap (%). The optimality gap is computed as $(\text{UB} - \text{LB}) \cdot |\text{UB}|^{-1}$, where LB corresponds to the lower bound. The reported CPU times include all preprocessing and decision diagram compilation runtimes. A dash (-) indicates a missing value due to a time-out. Bold entries indicate new or improved bounds.

Overall, no method always dominates the others on this benchmark. However, `FLMS` and `IDDR` both solve 27 instances optimally, while `MibS` solves 15 instances optimally. We also observe that `IDDR` is able to report bounds for 59 (out of 60) instances of the testbed, whereas `FLMS` and `MibS` report 52 and 37, respectively, within the 1-hour time limit. In addition, we note that `IDDR` finds new or improved upper bounds for six instances, and closes two of these instances (p0201-0.1 and p0282-0.9) for the first time. We conclude that `IDDR` provides competitive results on this generic benchmark set, even if the decision diagram compilation does not leverage specific problem structures.

8.2 Bilevel Independent Set Problem with Knapsack Constraints

We next consider the BISP-KP, which we introduced in Section 7. The purpose of this experiment is to evaluate the performance of our method when the decision diagram compilation can take advantage of problem-specific structures. We consider randomly generated instances with varying problem parameters:

– Independent set problem: We randomly generate Erdös-Rényi graphs of different sizes and varying edge density, similar to Bergman et al. [7], with $|V| \in \{50, 100, 200, 300\}$ and edge density (the probability of an edge appearing in the graph) $p \in \{0.25, 0.5, 0.75\}$.

Table 2: Computational evaluation on the BISP-KC. The decision diagram approach `IDDR` with varying maximum width W is compared to branch-and-cut methods `FLMS` [19] and `MibS` [36]. Method `IDDR (`$W = 0$`)` starts with an empty diagram. For each instance class and method, we report the number of instances solved, the average solving time (of the solved instances), and the average optimality gap (of the unsolved instances). All methods apply a 1-hour time limit.

$\lvert V\rvert$	p	`FLMS`			`MibS`			`IDDR` ($W = 0$)		
		#Opt	Time (s)	Gap (%)	#Opt	Time (s)	Gap (%)	#Opt	Time (s)	Gap (%)
	0.25	5/30	1145	49.45	0/30	-	23.65	30/30	3	0
50	0.5	4/30	410	22.26	3/30	64	15.42	30/30	2	0
	0.75	21/30	113	7.45	18/30	502	5.52	30/30	1	0
	0.25	0/30	-	26.1	0/30	-	18.56	30/30	24	0
100	0.5	0/30	-	18.43	0/30	-	12.77	30/30	8	0
	0.75	5/30	2078	9.29	1/30	1355	7.33	30/30	7	0
	0.25	0/30	-	-	0/30	-	20.04	10/30	2971	12.73
200	0.5	0/30	-	-	0/30	-	9.67	30/30	321	0
	0.75	0/30	-	9.46	0/30	-	5.52	30/30	73	0
	0.25	0/30	-	-	0/30	-	-	0/30	-	13.92
300	0.5	0/30	-	-	0/30	-	9.92	14/30	2426	4.23
	0.75	0/30	-	3.35	0/30	-	7.58	30/30	642	0

$\lvert V\rvert$	p	`IDDR` ($W = 500$)			`IDDR` ($W = 1000$)			`IDDR` ($W = 2000$)		
		#Opt	Time (s)	Gap (%)	#Opt	Time (s)	Gap (%)	#Opt	Time (s)	Gap (%)
	0.25	30/30	3	0	30/30	3	0	30/30	3	0
50	0.5	30/30	1	0	30/30	1	0	30/30	1	0
	0.75	30/30	1	0	30/30	1	0	30/30	1	0
	0.25	30/30	34	0	30/30	32	0	30/30	31	0
100	0.5	30/30	6	0	30/30	6	0	30/30	7	0
	0.75	30/30	4	0	30/30	4	0	30/30	4	0
	0.25	9/30	2021	6.49	9/30	1687	6.55	9/30	1667	5.99
200	0.5	30/30	183	0	30/30	147	0	30/30	136	0
	0.75	30/30	43	0	30/30	43	0	30/30	43	0
	0.25	0/30	-	13.1	0/30	-	12.04	0/30	-	11.75
300	0.5	18/30	1724	1.98	23/30	1833	3.6	27/30	1465	1.74
	0.75	30/30	177	0	30/30	166	0	30/30	166	0

– Knapsack constraints: The weight vectors $w_L, w_F, \omega_L, \omega_F$ are drawn from $U[-10, 10]$, while the right-hand sides b_L, b_F are defined as $\alpha \cdot (n_F + n_L)$ using a scaling parameter $\alpha \in \{-0.1, 0, 0.1\}$.
– Objective functions: The leader coefficients c_L and c_F are drawn from $U[-50, 50]$, and the follower coefficients d are drawn from $U[-50, -25]$.

We use the three parameters $|V|$, p, and α to define an instance configuration. This allows us to evaluate the performance of the methods when the graph size $|V|$ increases, when the edge density p increases, and when the knapsack capacity increases (via α). With the above ranges, this yields 36 configurations. For each configuration $(|V|, p, \alpha)$ we generate 10 instances, resulting in a total of 360 instances.

We compare the decision diagram approach IDDR with varying width ($W = 0, 500, 1000, 2000$) to the branch-and-cut methods FLMS and MibS. All methods use CPLEX 22 as the default MIP solver. The results are presented in Table 2. (A more detailed comparison, using performance plots, can be found in Figure C.1 in Appendix C..) For each method, we report the total number of instances solved (#Opt), the average solving time in seconds of the solved instances, and the average optimality gap (%) of the unsolved instances.
One can observe that the generic integer bilevel programming solvers FLMS and MibS do not scale beyond $|V| = 50$, whereas IDDR can optimally solve instances up to $|V| = 300$, for larger density. In addition, we find that instances with denser graphs are easier for all methods. The impact of the scaling factor α for the knapsack constraints is less pronounced, so we do not show this separately in the table. Overall, on this benchmark, FLMS solves 35 instances, MibS solves 22 instances, and IDDR solves up to 306 instances. We also observe that IDDR generally performs better when providing a larger initial diagram.
This demonstrates that the decision diagram-based reformulation substantially benefits from leveraging domain-specific structure, relative to the generic approaches. Larger initial diagrams generally strengthen the formulation, and reduce solving time by lowering the number of exact value function computations. An interesting observation is that the baseline approach IDDR with $W = 0$ performs remarkably well. It shows that even without an initial diagram IDDR, it can still be effective on this problem domain.

9 Conclusions

We introduced a new single-level reformulation for integer bilevel programs. It is based on a compact representation of the follower's feasible solution set using decision diagrams, over which we define a network flow model, parameterized by the leader's solution. Our reformulation ensures bilevel optimality by encoding the optimality conditions over the paths of the network, and provides an optimal solution when it is defined over an exact decision diagram. Because exact decision diagrams can grow exponentially large, we introduced a scalable approach that uses restricted decision diagrams of polynomial size, and showed that these provide dual bounds. To obtain an exact method, we proposed to iteratively solve and strengthen the relaxation by adding a new path to the decision diagram or, equivalently, by adding bilevel no-good cuts. We also extended our DD-based relaxations for the pessimistic setting. In our experimental evaluation, we first compared our approach to state-of-the-art solvers on the BOBILib benchmark with integer bilevel programming instances. We showed that our method provides competitive results, optimally solving as many instances as the best existing solver. In addition, we found new or improved bounds for six instances and closed two open instances for the first time. We also introduced a new problem domain, the bilevel independent set problem with knapsack constraint, to study the performance of our approach when the decision diagram compilation can benefit from the combinatorial structure of the follower's problem. Indeed, we showed that our method scaled to much larger instances than the generic integer bilevel programming solvers. We conclude that our decision diagram reformulation can be a scalable and effective approach for solving integer bilevel programming problems.

Acknowledgements Dr. Lozano gratefully acknowledges the support of the *Air Force Office of Scientific Research* under grant FA9550-22-1-0236.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Akers: Binary decision diagrams. IEEE Transactions on computers **100**(6), 509–516 (1978)
2. Bard, J.F.: Coordination of a multidivisional organization through two levels of management. Omega **11**(5), 457–468 (1983)
3. Bard, J.F.: Practical Bilevel Optimization, *Nonconvex Optimization and Its Applications*, vol. 30. Springer US, Boston, MA (1998)
4. Bard, J.F., Moore, J.T.: A branch and bound algorithm for the bilevel programming problem. SIAM Journal on Scientific and Statistical Computing **11**(2), 281–292 (1990)
5. Beck, Y., Ljubić, I., Schmidt, M.: A survey on bilevel optimization under uncertainty. European Journal of Operational Research **311**(2), 401–426 (2023)
6. Bergman, D., Cire, A.A., van Hoeve, W.J.: Lagrangian bounds from decision diagrams. Constraints **20**, 346–361 (2015)
7. Bergman, D., Ciré, A.A., van Hoeve, W.J., Hooker, J.N.: Optimization Bounds from Binary Decision Diagrams. INFORMS Journal on Computing **26**(2), 253–268 (2014)
8. Bergman, D., Cire, A.A., Van Hoeve, W.J., Hooker, J.: Decision Diagrams for Optimization. Artificial Intelligence: Foundations, Theory, and Algorithms. Springer International Publishing (2016)
9. Bracken, J., McGill, J.T.: Mathematical Programs with Optimization Problems in the Constraints. Operations Research **21**(1), 37–44 (1973)
10. Bryant, R.E.: Graph-based algorithms for boolean function manipulation. IEEE Transactions on Computers **C-35**, 677–691 (1986)
11. Bui, Q.M., Carvalho, M., Neto, J.: Solving combinatorial pricing problems using embedded dynamic programming models. arXiv preprint arXiv:2403.12923 (2024)
12. Castro, M.P., Cire, A.A., Beck, J.C.: A combinatorial cut-and-lift procedure with an application to 0–1 second-order conic programming. Mathematical Programming **196**(1), 115–171 (2022)
13. Castro, M.P., Ciré, A.A., Beck, J.C.: Decision Diagrams for Discrete Optimization: A Survey of Recent Advances. INFORMS Journal on Computing **34**(4), 2271–2295 (2022)
14. Cire, A.A., Van Hoeve, W.J.: Multivalued decision diagrams for sequencing problems. Operations Research **61**(6), 1411–1428 (2013)
15. Colson, B., Marcotte, P., Savard, G.: An overview of bilevel optimization. Annals of Operations Research **153**(1), 235–256 (2007)
16. Dempe, S., Kalashnikov, V., Rios-Mercado, R.Z.: Discrete bilevel programming: Application to a natural gas cash-out problem. European Journal of Operational Research **166**(2), 469–488 (2005)
17. DeNegre, S.T., Ralphs, T.K.: A Branch-and-cut Algorithm for Integer Bilevel Linear Programs. In: J.W. Chinneck, B. Kristjansson, M.J. Saltzman (eds.) Operations research and cyber-infrastructure, pp. 65–78. Springer (2009)

18. Fischetti, M., Ljubic, I., Monaci, M., Sinnl, M.: Intersection cuts for bilevel optimization. In: Q. Louveaux, M. Skutella (eds.) Integer Programming and Combinatorial Optimization (IPCO), Proceedings, *Lecture Notes in Computer Science*, vol. 9682, pp. 77–88. Springer (2016)
19. Fischetti, M., Ljubić, I., Monaci, M., Sinnl, M.: A New General-Purpose Algorithm for Mixed-Integer Bilevel Linear Programs. Operations Research **65**(6), 1615–1637 (2017)
20. Fortuny-Amat, J., McCarl, B.: A representation and economic interpretation of a two-level programming problem. Journal of the Operational Research Society **32**(9), 783–792 (1981)
21. van Hoeve, W.J.: Graph coloring with decision diagrams. Mathematical Programming **192**(1), 631–674 (2022)
22. van Hoeve, W.J.: An Introduction to Decision Diagrams for Optimization. In: INFORMS TutORials in Operations Research, pp. 117–145. INFORMS (2024)
23. Hooker, J.N.: Job Sequencing Bounds from Decision Diagrams. In: J.C. Beck (ed.) Principles and Practice of Constraint Programming (CP), Proceedings, *Lecture Notes in Computer Science*, vol. 10416, pp. 565–578. Springer (2017)
24. Karahalios, A., van Hoeve, W.: Column Elimination for Capacitated Vehicle Routing Problems. In: A.A. Ciré (ed.) Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR), Proceedings, *Lecture Notes in Computer Science*, vol. 13884, pp. 35–51. Springer (2023)
25. Kleinert, T., Labbé, M., Ljubić, I., Schmidt, M.: A Survey on Mixed-Integer Programming Techniques in Bilevel Optimization. EURO Journal on Computational Optimization **9**, 100007 (2021)
26. Leal, M., Ponce, D., Puerto, J.: Portfolio problems with two levels decision-makers: Optimal portfolio selection with pricing decisions on transaction costs. European Journal of Operational Research **284**(2), 712–727 (2020)
27. LeBlanc, L.J., Boyce, D.E.: A bilevel programming algorithm for exact solution of the network design problem with user-optimal flows. Transportation Research Part B: Methodological **20**(3), 259–265 (1986)
28. Lee, C.Y.: Representation of switching circuits by binary-decision programs. The Bell System Technical Journal **38**(4), 985–999 (1959)
29. Lozano, L., Bergman, D., Cire, A.A.: Constrained Shortest-Path Reformulations for Discrete Bilevel and Robust Optimization. arXiv preprint arXiv:2206.12962 (2022)
30. Lozano, L., Bergman, D., Cire, A.A.: Network relaxations for discrete bilevel optimization under linear interactions. arXiv preprint arXiv:2407.18193 (2024)
31. Lozano, L., Smith, J.C.: A Value-Function-Based Exact Approach for the Bilevel Mixed-Integer Programming Problem. Operations Research **65**(3), 768–786 (2017)
32. Lozano, L., Smith, J.C.: A binary decision diagram based algorithm for solving a class of binary two-stage stochastic programs. Mathematical Programming **191**(1), 381–404 (2022)
33. Marcotte, P.: Network design problem with congestion effects: A case of bilevel programming. Mathematical Programming **34**(2), 142–162 (1986)
34. Moore, J.T.M., Bard, J.F.: The Mixed Integer Linear Bilevel Programming Problem. Operations Research **38**(5), 911–921 (1990)
35. Smith, J.C., Song, Y.: A survey of network interdiction models and algorithms. European Journal of Operational Research **283**(3), 797–811 (2020)
36. Tahernejad, S., Ralphs, T.K., DeNegre, S.T.: A branch-and-cut algorithm for mixed integer bilevel linear optimization problems and its implementation. Mathematical Programming Computation **12**(4), 529–568 (2020)
37. Tang, Z., van Hoeve, W.J.: Dual bounds from decision diagram-based route relaxations: An application to truck-drone routing. Transportation Science **58**(1), 257–278 (2024)
38. Thürauf, J., Kleinert, T., Ljubić, I., Ralphs, T., Schmidt, M.: BOBILib: Bilevel Optimization (Benchmark) Instance Library. Optimization Online Eprint (2024). URL `https://optimization-online.org/?p=27063`
39. Tjandraatmadja, C., van Hoeve, W.J.: Target Cuts from Relaxed Decision Diagrams. INFORMS Journal on Computing **31**(2), 285–301 (2019)
40. Tjandraatmadja, C., van Hoeve, W.J.: Incorporating bounds from decision diagrams into integer programming. Mathematical Programming Computation **13**(2), 225–256 (2021)

41. Wegener, I.: Branching programs and binary decision diagrams: theory and applications. SIAM (2000)
42. Wiesemann, W., Tsoukalas, A., Kleniati, P.M., Rustem, B.: Pessimistic bilevel optimization. SIAM Journal on Optimization **23**(1), 353–380 (2013)
43. Zare, M.H., Borrero, J.S., Zeng, B., Prokopyev, O.A.: A note on linearized reformulations for a class of bilevel linear integer problems. Annals of Operations Research **272**, 99–117 (2019)

Appendix

A. BOBILib Instances Information

Table A.1 presents the following information for the BOBILib instances [38]: name, number of leader variables, number of follower variables, number of follower constraints, and the instance status as reported in the official website accessed on 01/16/2025. Bold lines indicate previously open instances closed for the first time by our approach.

Table A.1: Information about BOBILib instances [38].

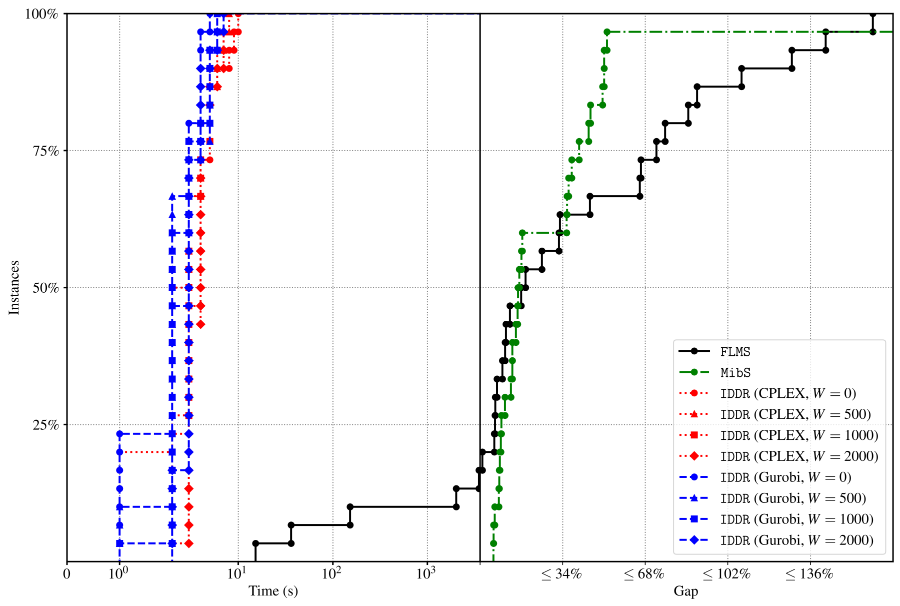
Instance	n_L	n_F	m_F	Status	Instance	n_L	n_F	m_F	Status
stein27-0.1	25	2	118	Closed	mod010-0.1	2,390	265	291	Open
stein27-0.5	14	13	118	Closed	mod010-0.5	1,328	1,327	291	Open
stein27-0.9	3	24	118	Closed	mod010-0.9	266	2,389	291	Closed
stein45-0.1	41	4	331	Closed	air03-0.1	9,682	1,075	248	Open
stein45-0.5	23	22	331	Closed	air03-0.5	5,379	5,378	248	Open
stein45-0.9	5	40	331	Closed	air03-0.9	1,076	9,681	248	Open
enigma-0.1	90	10	42	Closed	air04-0.1	8,014	890	1646	Open
enigma-0.5	50	50	42	Closed	air04-0.5	4,452	4,452	1646	Open
enigma-0.9	10	90	42	Closed	air04-0.9	891	8,013	1646	Open
lseu-0.1	81	8	28	Closed	air05-0.1	6,476	719	852	Closed
lseu-0.5	45	44	28	Closed	air05-0.5	3,598	3,597	852	Open
lseu-0.9	9	80	28	Open	air05-0.9	720	6,475	852	Open
p0033-0.1	30	3	16	Closed	fast0507-0.1	56,709	6,300	507	Closed
p0033-0.5	17	16	16	Closed	fast0507-0.5	31,505	31,504	507	Closed
p0033-0.9	4	29	16	Closed	fast0507-0.9	6,301	56,708	507	Closed
p0201-0.1	181	20	133	**Open**	cap6000-0.1	5,400	600	2299	Closed
p0201-0.5	101	100	133	Closed	cap6000-0.5	3,000	3,000	2299	Open
p0201-0.9	21	180	133	Closed	cap6000-0.9	600	5,400	2299	Open
p0282-0.1	254	28	241	Closed	mitre-0.1	9,652	1,072	2437	Open
p0282-0.5	141	141	241	Closed	mitre-0.5	5,362	5,362	2437	Open
p0282-0.9	29	253	241	**Open**	mitre-0.9	1,073	9,651	2437	Open
p0548-0.1	494	54	176	Open	nw04-0.1	78,734	8,748	72	Closed
p0548-0.5	274	274	176	Open	nw04-0.5	43,741	43,741	72	Open
p0548-0.9	55	493	176	Open	nw04-0.9	8,749	78,733	72	Open
p2756-0.1	2,481	275	755	Open	seymour-0.1	1,235	137	4944	Open
p2756-0.5	1,378	1,378	755	Open	seymour-0.5	686	686	4944	Closed
p2756-0.9	276	2,480	755	Open	seymour-0.9	138	1,234	4944	Closed
l152lav-0.1	1,791	198	193	Closed	harp2-0.1	2,694	299	185	Open
l152lav-0.5	995	994	193	Open	harp2-0.5	1,497	1,496	185	Open
l152lav-0.9	199	1,790	193	Open	harp2-0.9	300	2,693	185	Open

B. MIP Solver Comparison for the Algorithm in [19]

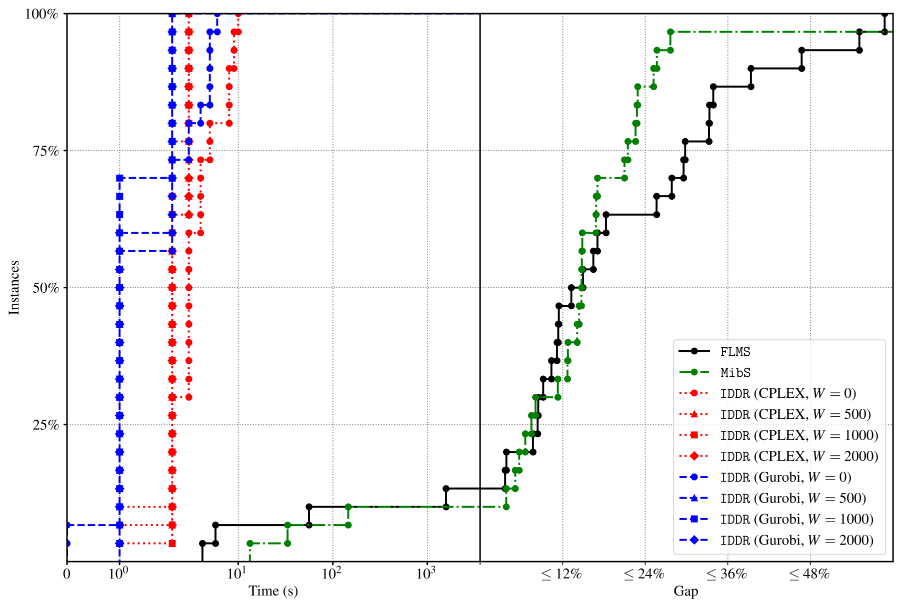
Table B.2: MIP solver comparison on BOBILib instances for `FLMS`.

Instance	CPLEX 12			CPLEX 22		
	Time (s)	UB	Gap	Time (s)	UB	Gap
stein27-0.1	0	18	0.00	0	18	0.00
stein27-0.5	0	19	0.00	0	19	0.00
stein27-0.9	0	24	0.00	0	24	0.00
stein45-0.1	6	30	0.00	7	30	0.00
stein45-0.5	0	32	0.00	0	32	0.00
stein45-0.9	0	40	0.00	0	40	0.00
enigma-0.1	0	0	0.00	1	0	0.00
enigma-0.5	9	0	0.00	8	0	0.00
enigma-0.9	582	0	0.00	593	0	0.00
lseu-0.1	0	1120	0.00	0	1120	0.00
lseu-0.5	3,600	2274	3.32	3,600	2313	7.63
lseu-0.9	73	5838	0.00	95	5838	0.00
p0033-0.1	0	3089	0.00	0	3089	0.00
p0033-0.5	0	3095	0.00	0	3095	0.00
p0033-0.9	0	4679	0.00	0	4679	0.00
p0201-0.1	3,600	12670	24.58	3,600	12465	20.90
p0201-0.5	1,588	13635	0.00	2,195	13635	0.00
p0201-0.9	4	15025	0.00	5	15025	0.00
p0282-0.1	7	260781	0.00	3	260781	0.00
p0282-0.5	3,600	272659	0.70	3,600	272659	0.93
p0282-0.9	3,600	615101	30.54	3,600	614837	28.85
p0548-0.1	3,600	11063	16.12	3,600	11086	16.85
p0548-0.5	3,600	-	-	3,600	-	-
p0548-0.9	3,600	49476	60.97	3,600	48942	60.25
p2756-0.1	3,600	14395	76.26	3,600	14550	77.07
p2756-0.5	3,600	23574	83.07	3,600	23547	83.05
p2756-0.9	3,600	33699	86.23	3,600	33362	86.12
l152lav-0.1	7	4722	0.00	7	4722	0.00
l152lav-0.5	3,600	4851	2.01	3,600	4817	1.27
l152lav-0.9	3,600	5171	7.86	3,600	5090	6.37
mod010-0.1	3	6554	0.00	5	6554	0.00
mod010-0.5	3,600	6689	1.87	3,600	6678	1.73
mod010-0.9	3,600	7604	13.71	3,600	7620	13.92
air03-0.1	3,600	386220	10.80	3,600	378890	8.96
air03-0.5	3,600	524198	33.30	3,600	513226	31.04
air03-0.9	3,600	819652	57.23	3,600	762132	53.18
air04-0.1	3,600	56400	0.20	3,600	56415	0.35
air04-0.5	3,600	61281	8.28	3,600	60725	7.45
air04-0.9	3,600	-	-	3,600	-	-
air05-0.1	848	26577	0.00	1,005	26577	0.00
air05-0.5	3,600	31693	16.61	3,600	32528	18.70
air05-0.9	3,600	42796	38.50	3,600	-	-
fast0507-0.1	2	12484	0.00	1	12484	0.00
fast0507-0.5	1	61439	0.00	1	61439	0.00
fast0507-0.9	1	109916	0.00	1	109916	0.00
cap6000-0.1	433	-1967015	0.00	802	-1967015	0.00
cap6000-0.5	3,600	-	-	3,600	-	-
cap6000-0.9	3,600	-	-	3,600	-	-
mitre-0.1	3,600	-	-	3,600	-	-
mitre-0.5	3,600	-	-	3,600	-	-
mitre-0.9	3,600	-	-	3,600	169470	31.95
nw04-0.1	904	17066	0.00	1,023	17066	0.00
nw04-0.5	3,600	26874	37.90	3,600	26396	37.08
nw04-0.9	3,600	52290	66.11	3,600	44026	59.80
seymour-0.1	3,600	476	1.50	3,600	476	1.33
seymour-0.5	1	807	0.00	1	807	0.00
seymour-0.9	0	1251	0.00	0	1251	0.00
harp2-0.1	3,600	-71920826	2.74	3,600	-72111431	2.45
harp2-0.5	3,600	-	-	3,600	-	-
harp2-0.9	3,600	162804	45544.93	3,600	-154337	47836.26

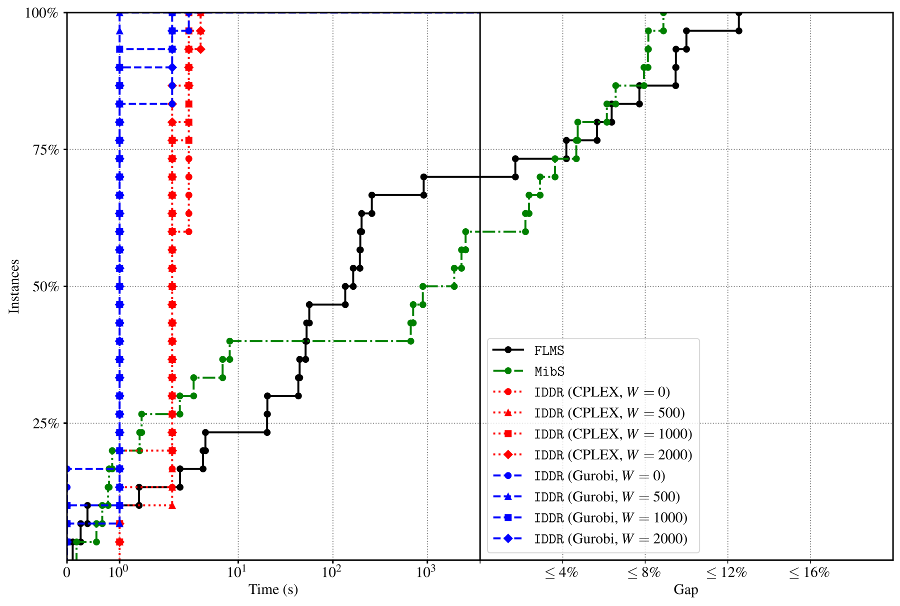
C. Performance plots for BISP-KC



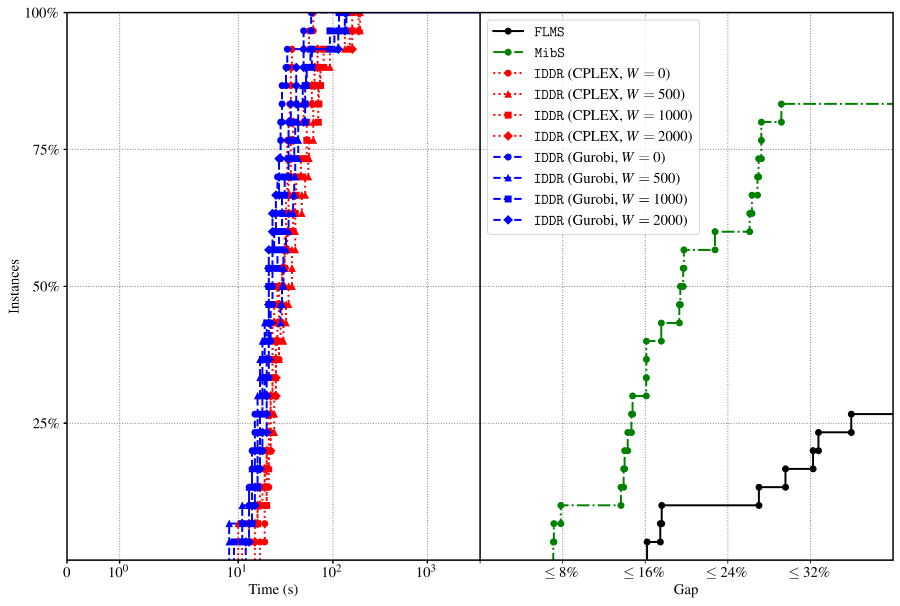
a. $|V| = 50$, $p = 0.25$



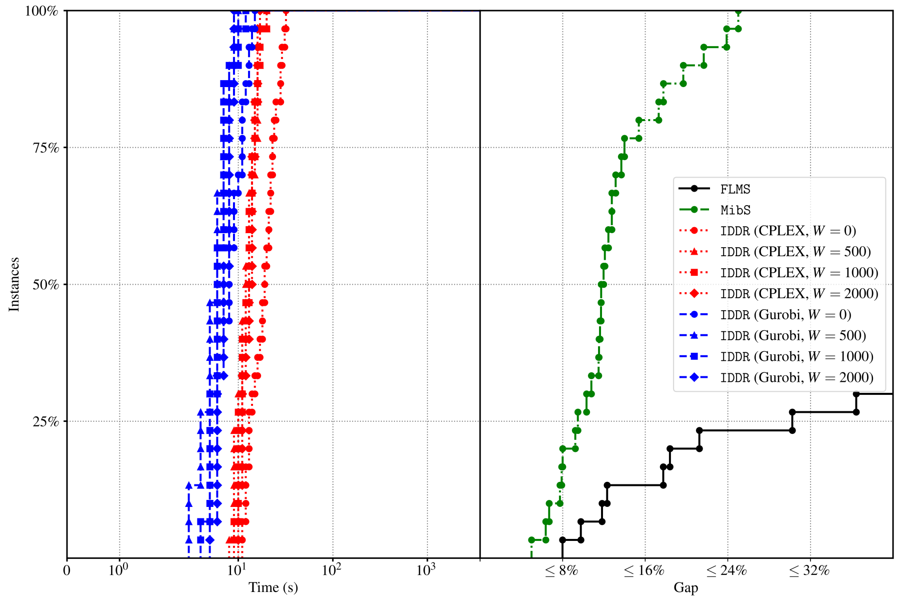
b. $|V| = 50$, $p = 0.50$



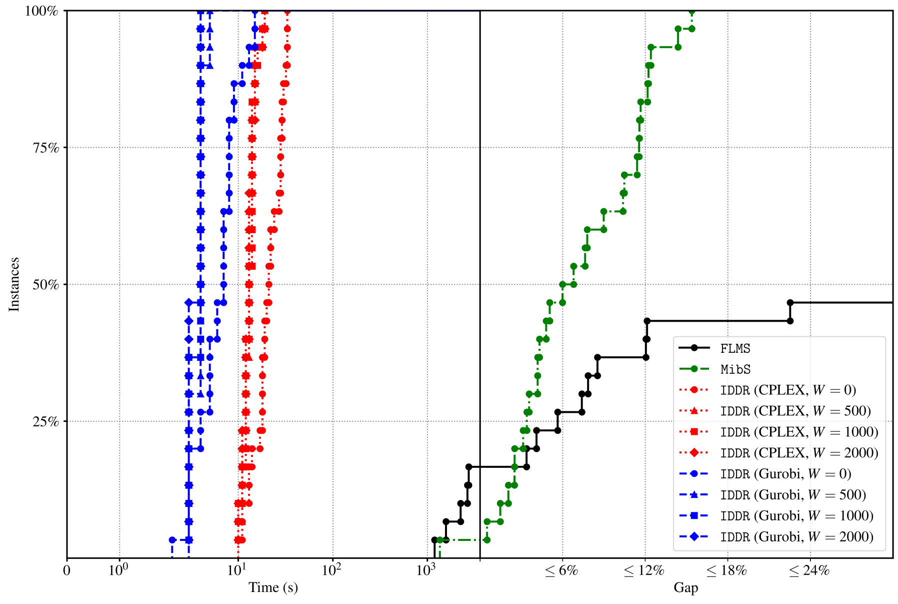
c. $|V| = 50$, $p = 0.75$



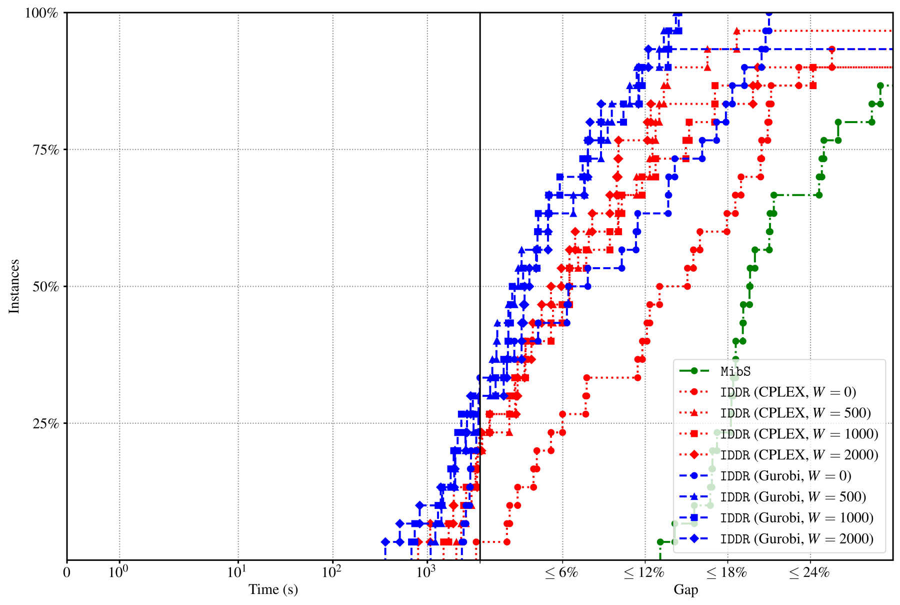
d. $|V| = 100$, $p = 0.25$



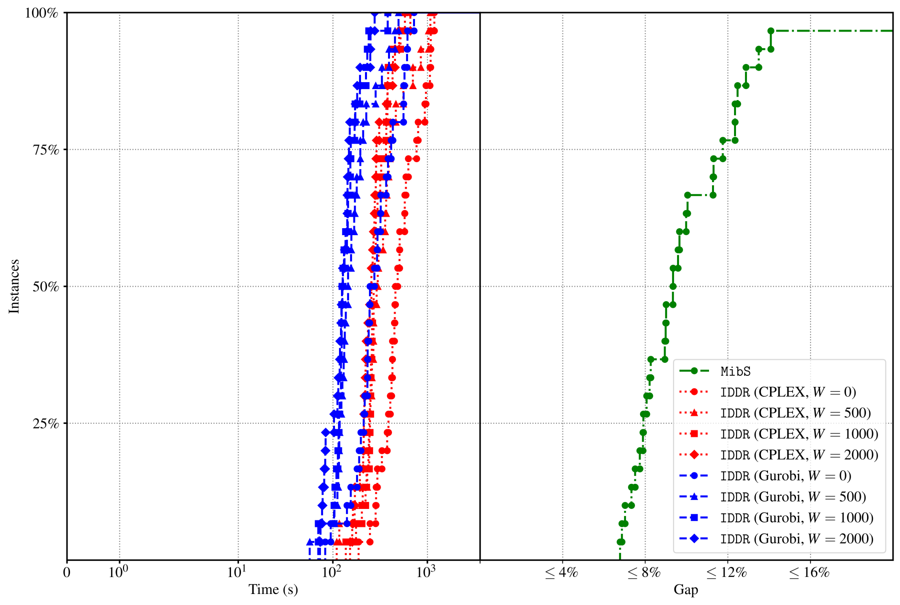
e. $|V| = 100$, $p = 0.50$



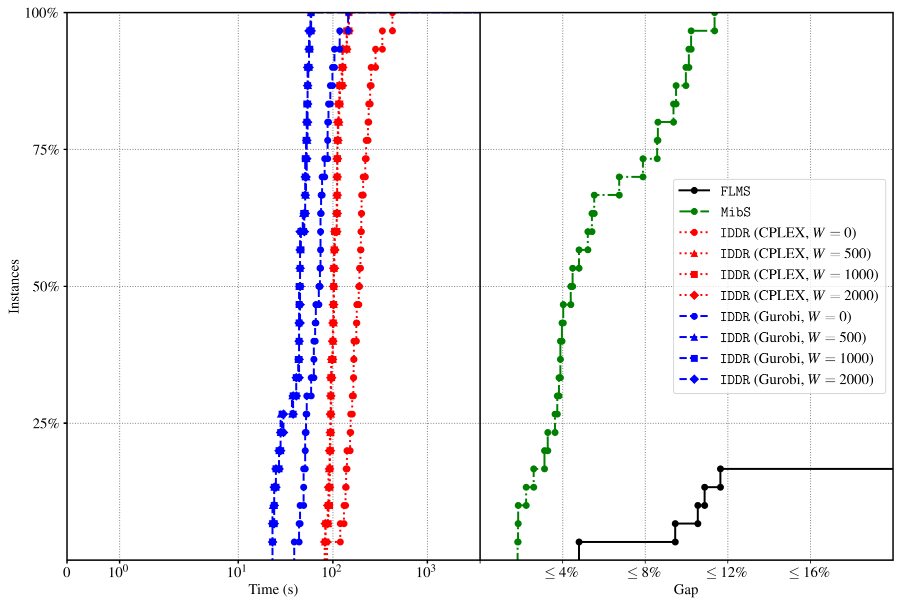
f. $|V| = 100$, $p = 0.75$



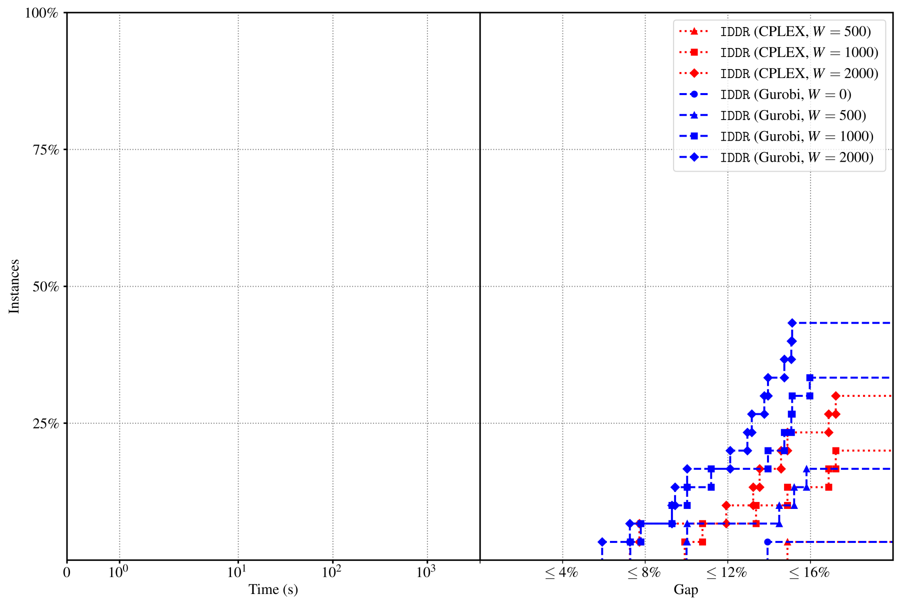
g. $|V| = 200,\ p = 0.25$



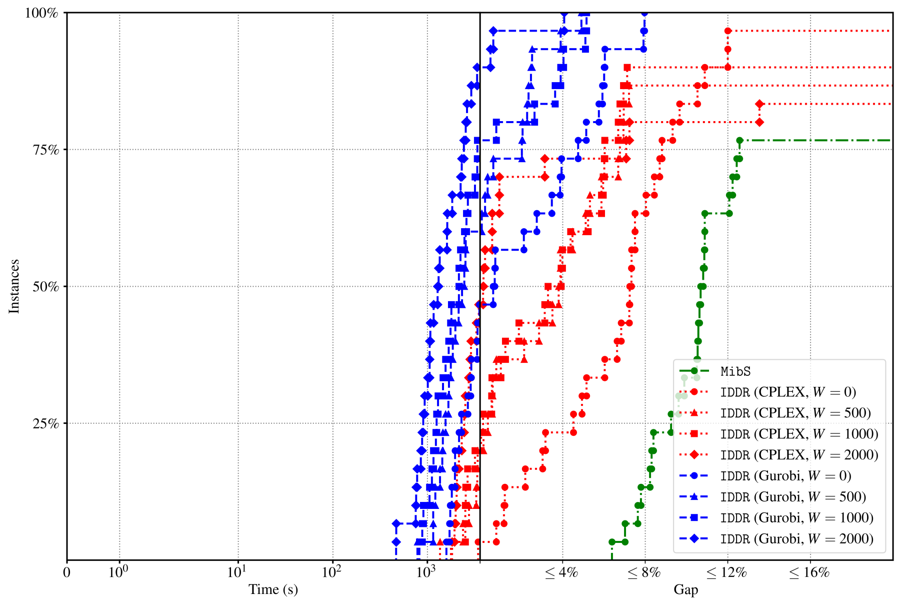
h. $|V| = 200,\ p = 0.50$



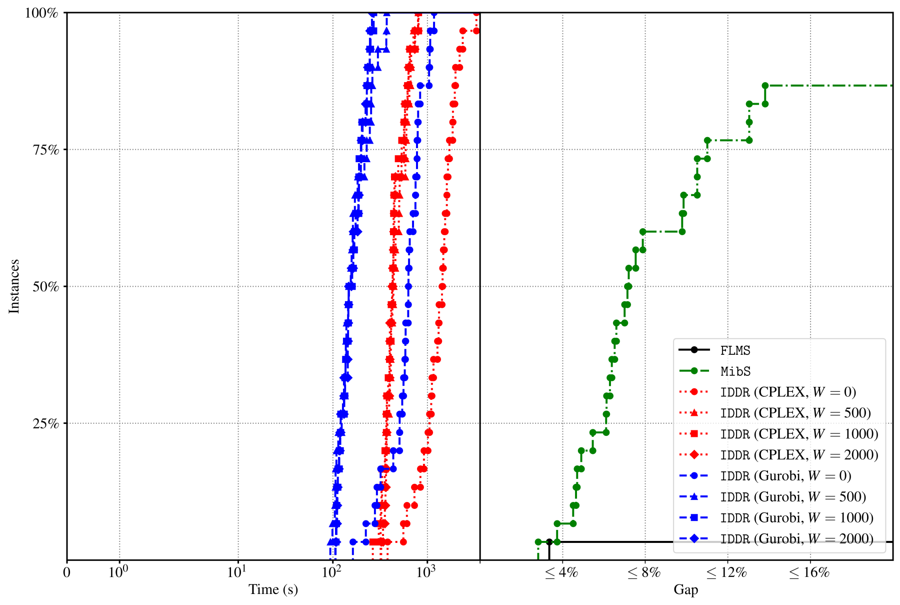
i. $|V| = 200,\ p = 0.75$



j. $|V| = 300,\ p = 0.25$



k. $|V| = 300$, $p = 0.50$



l. $|V| = 300$, $p = 0.75$

Fig. C.1: Performance profiles for BISP-KC. Left x-axis shows running times of solved instances in log scale, and right x-axis shows optimality gaps of unsolved instances.