Ann Oper Res
DOI 10.1007/s10479-016-2118-3

S.I.: PARDALOS60

A simple greedy heuristic for linear assignment interdiction

Vladimir Stozhkov[1] · **Vladimir Boginski**[1,4] · **Oleg A. Prokopyev**[2] · **Eduardo L. Pasiliao**[3]

Abstract We consider a bilevel extension of the classical linear assignment problem motivated by network interdiction applications. Specifically, given a bipartite graph with two different (namely, the leader's and the follower's) edge costs, the follower solves a linear assignment problem maximizing his/her own profit, whereas the leader is allowed to affect the follower's decisions by eliminating some of the vertices from the graph. The leader's objective is to minimize the total cost given by the cost of the interdiction actions plus the cost of the assignments made by the follower. The considered problem is strongly *NP*-hard. First, we formulate this problem as a linear mixed integer program (MIP), which can be solved by commercial MIP solvers. More importantly, we also describe a greedy-based construction heuristic, which provides (under some mild conditions) an optimal solution for the case, where the leader's and the follower's edge costs are equal to one. Finally, we present the results of our computational experiments comparing the proposed heuristic against an MIP solver.

✉ Oleg A. Prokopyev
prokopyev@engr.pitt.edu

Vladimir Stozhkov
vstozhkov@ufl.edu

Vladimir Boginski
vb@ufl.edu

Eduardo L. Pasiliao
pasiliao@eglin.af.mil

1 Department of Industrial and Systems Engineering, University of Florida, 303 Weil Hall, Gainesville, FL 32611, USA

2 Department of Industrial Engineering, University of Pittsburgh, 1048 Benedum Hall, Pittburgh, PA 15261, USA

3 Munitions Directorate, Air Force Research Laboratory, 101 W. Eglin Blvd., Eglin AFB, FL 32542, USA

4 Department of Industrial Engineering and Management Systems, University of Central Florida, 12800 Pegasus Dr., Orlando, FL 32816, USA

Published online: 27 February 2016

Keywords Bilevel programming · Assignment interdiction · Linear assignment

1 Introduction

Bilevel optimization problems (Bard 1998; Colson et al. 2007; Dempe 2002; Migdalas et al. 1998) involve two independent *decision makers* (DMs) referred to as the *leader* (upper-level DM) and the *follower* (lower-level DM). The decision of the leader, who acts first by setting values of his/her variables, affects the follower's (lower-level) optimization problem. The follower's constraint set and objective are functions of the leader's and the follower's variables simultaneously. Furthermore, the leader's objective in the upper-level optimization problem is also functionally dependant on the follower's decision variables. Thus, while making his/her decision, the leader should take into account the follower's response, i.e., an optimal solution of the follower's optimization problem, which is formally referred to as the *lower-level reaction set* (Colson et al. 2007).

Naturally, bilevel optimization problems are often applied to represent adversarial relationships arising in military and homeland security applications, thus, resulting in different types of attacker-defender, or defender-attacker models (Brown et al. 2006). In particular, many classes of *network interdiction problems* can be recasted as bilevel programs (Shen et al. 2012; Wood 1993; Zenklusen 2010), where the leader and the follower correspond to the *interdictor* and the *adversary* (the latter is often referred to as the *evader*), respectively.

In this paper we consider the *Linear Assignment Interdiction Problem* (further referred to as **LAIP**), which can be viewed as a bilevel extension of the classical linear assignment problem (LAP) (Nemhauser and Wolsey 1988). Formally, let $G = (V \cup U, E)$ be a balanced weighted bipartite graph, i.e., $|V| = |U| = n$, where costs (weights) for each edge $(i, j) \in E$ are possibly different for the leader (given by c_{ij}) and the follower (d_{ij}). The follower solves a linear assignment problem maximizing his/her own profit based on the corresponding edge costs. However, the leader has the option to interdict the follower's decisions by deleting some of the vertices of the graph. The leader's goal is to minimize the total cost given by the cost of the interdiction actions plus the cost of the follower's optimal assignments computed with respect to the leader's costs. In the bilevel optimization framework this problem can be stated as follows:

$$\textbf{LAIP:} \quad \min_{\mathbf{y},\mathbf{z}} \sum_{i\in V} a_i y_i + \sum_{j\in U} b_j z_j + \sum_{(i,j)\in E} c_{ij} x_{ij} \tag{1a}$$

$$\text{s.t.} \quad y_i \in \{0,1\},\ z_j \in \{0,1\} \quad \forall\, i \in V,\ j \in U, \tag{1b}$$

$$\mathbf{x} \in \operatorname*{argmax}_{\mathbf{x}} \sum_{(i,j)\in E} d_{ij} x_{ij} \tag{1c}$$

$$\text{s.t.} \sum_{j\in U:\,(i,j)\in E} x_{ij} \le 1 - y_i \quad \forall\, i \in V, \tag{1d}$$

$$\sum_{i\in V:\,(i,j)\in E} x_{ij} \le 1 - z_j \quad \forall\, j \in U, \tag{1e}$$

$$x_{ij} \in \{0,1\} \quad \forall\, (i,j) \in E, \tag{1f}$$

where $\mathbf{a} \in \mathbb{R}^n_+$ and $\mathbf{b} \in \mathbb{R}^n_+$ are the costs incurred by the leader for removing vertices in V and U, respectively. Binary variables $\mathbf{y}$ and $\mathbf{z}$ represent the leader's interdiction decisions,

while variables $\mathbf{x}$ are used to model the follower's assignments. Clearly, constraints (1d)–(1e) ensure that the follower can only assign vertices that are not deleted by the leader.

In the remainder of the paper we make the following assumption, which is also common in the related assignment optimization literature (Burkard et al. 2009):

A1. The leader's and the follower's edge costs are nonnegative, i.e., $\mathbf{c} \in \mathbb{R}_+^{n\times n}$ and $\mathbf{d} \in \mathbb{R}_+^{n\times n}$.

The bilevel optimization and network interdiction literature is vast, however, to the authors' best knowledge, the following three papers are most closely related to the current work. The first one by Zenklusen (2010) considers a rather similar setting, where the leader can remove graph vertices, thus, affecting the decisions of the follower, who solves the maximum weighted matching problem in the considered graph (which is not necessarily bipartite). The overall leader's objective is to minimize the maximum weight matching obtained by the follower, which implies that all edges have the same costs for both DMs. Furthermore, in Zenklusen (2010) the vertex interdiction costs do not appear in the leader's objective function, but rather in the leader's budgetary constraint, which is not present in our model. This problem is referred to as *matching interdiction*. In Zenklusen (2010), the author establishes computational complexity results for several special classes of the problem. In general, the problem turns out to be *NP*-hard, which also motivates development of approximation methods and pseudo-polynomial algorithms (in particular, dynamic programming based) for graphs with bounded treewidth (Zenklusen 2010).

The other two papers consider the *bilevel assignment problem* (Beheshti et al. 2015; Gassner and Klinz 2009), where each DM has its own objective function (linear sum or linear bottleneck as in standard versions of the linear assignment problem) and controls a distinct set of edges in a given bipartite graph. The leader acts first by choosing some of his/her edges and, subsequently, the follower completes the assignment process. The edges selected by the leader and the follower are required to form a perfect matching. In Gassner and Klinz (2009), the authors establish that the problem is *NP*-hard in various settings, while the work in Beheshti et al. (2015) is mostly concerned with the development of an exact branch-and-bound based solution approach. The key difference of Beheshti et al. (2015), Gassner and Klinz (2009) from our work is that the leader does not remove graph vertices but rather performs some of the assignments initially by himself/herself, thus, indirectly enforcing the follower to select assignments that are preferable to the leader.

The remainder of the paper is organized as follows. In Sect. 2 we demonstrate that the considered bilevel optimization problem is strongly *NP*-hard. In Sect. 3 by following a standard approach from the bilevel optimization literature we formulate it as a linear mixed integer program (linear MIP), which implies that **LAIP** can be solved by commercial MIP solvers. Section 4 proposes a simple greedy-based construction heuristic approach, which is also shown to provide (under some mild conditions) an optimal solution in the case of unit edge costs for both the leader and the follower. In Sect. 5 we present the results of our computational experiments comparing the proposed heuristic against a commercial MIP solver. Finally, we conclude the discussion by highlighting possible directions of future research in Sect. 6.

2 Computational complexity issues

A number of classes of bilevel optimization problems are known to be *NP*-hard (Beheshti et al. 2015; Deng 1998). Furthermore, as mentioned in the previous section, our model is similar to bilevel problems discussed in Gassner and Klinz (2009), Zenklusen (2010), namely, the

matching interdiction and the bilevel assignment problems, which are also *NP*-hard. Thus, it is relatively unsurprising that **LAIP** is *NP*-hard as well.

Before we proceed with a formal proof of the result, it is important to note that the lower-level reaction set of a bilevel program (for instance, problem (1c)–(1f) for **LAIP**) is not necessarily a singleton and the follower has an option of choosing one of his/her optimal solutions. Consequently, if the follower selects a solution, which is the most or the least favorable for the leader, then the corresponding cases of the bilevel program are referred to as *optimistic* or *pessimistic*, respectively (Colson et al. 2007). In our discussion next we consider both of the above.

Following the standard approach (Garey and Johnson 1979), we define a recognition (or decision) version of the **LAIP** problem as follows:

LAIP-D
INSTANCE: A balanced bipartite graph $G = (V \cup U, E)$, a constant $K \in \mathbb{R}_+$, the leader's and the follower's edge "costs" $c_{ij} \in \mathbb{R}_+$ and $d_{ij} \in \mathbb{R}_+$ for each edge $(i, j) \in E$, respectively, and deletion "costs" $a_i \in \mathbb{R}_+$ and $b_j \in \mathbb{R}_+$ for each vertex $i \in V$ and $j \in U$, respectively.
QUESTION: Are there subsets of vertices $I_D \subseteq V$ and $J_D \subseteq V$ such that

$$\sum_{i \in I_D} a_i + \sum_{j \in J_D} b_j + \sum_{(i,j) \in E} c_{ij} x_{ij} \leq K \tag{2}$$

where $\mathbf{x}$ satisfies (1c)–(1f), $y_i = 1$ iff $i \in I_D$, $z_j = 1$ iff $j \in J_D$ and (1b) holds?

Similar to bilevel problems, we assume that there exist both optimistic and pessimistic cases of **LAIP-D**. To prove the theoretical computational complexity of **LAIP-D**, we construct a reduction from the *minimum vertex cover* problem known to be *NP*-hard (Garey and Johnson 1979). Recall that given graph $G = (V, E)$, a subset of vertices $V' \subseteq V$ is called a *vertex cover* of G, if for any $(i, j) \in E$, we have that either $i \in V'$, or $j \in V'$, or both $i, j \in V'$. Then the decision version of the minimum vertex cover problem is defined as follows: Given $G = (V, E)$ and $K \in \mathbb{Z}_+$, does there exists a vertex cover of G of size at most K?

Note that a reduction from the minimum vertex cover problem is also used in Gassner and Klinz (2009) and, due to similarities in the problem settings (recall our discussion in Sect. 1) in the proof below we exploit the construction ideas similar to those used in Gassner and Klinz (2009).

Proposition 1 *Problem* **LAIP-D** *is strongly NP-hard for both pessimistic and optimistic cases.*

Proof Given an instance of the minimum vertex cover problem, we construct an instance of **LAIP-D** using the following approach. We define balanced bipartite graph $\widetilde{G} = (U \cup W, \widetilde{E})$ with vertex sets $U = \{u_1, \ldots, u_{n+1}\}$ and $W = \{w_1, \ldots, w_{n+1}\}$, where $|U| = |W| = |V| + 1 = n + 1$. Simply speaking, U and W are copies of V with additional vertices u_{n+1} and w_{n+1}, respectively. Edge set $\widetilde{E}$ is given by

$$\widetilde{E} = E_0 \cup E_1 \cup E_2 \cup E_G \cup E_{\bar{G}},$$

where

- E_0 contains edges (u_i, w_i) with costs $c_{ii} = 0$ and $d_{ii} = 4$ for all $i \in \{1, 2, \ldots, n\}$;
- E_1 contains edges (u_i, w_{n+1}) and (u_{n+1}, w_j) with costs $c_{i,(n+1)} = c_{(n+1),j} = C$ and $d_{i,(n+1)} = d_{(n+1),j} = 2$ for all $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, n\}$;
- E_2 contains only one edge (u_{n+1}, w_{n+1}) with its costs $c_{(n+1),(n+1)} = 0$ and $d_{(n+1),(n+1)} = 1$;

- E_G is a subset of edges of the form (u_i, w_j) and (u_j, w_i) such that $(i, j) \in E$, i.e., edge (i, j) exists in the original graph G, with costs $c_{ij} = C$ and $d_{ij} = 5$ for all $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, n\}$;
- $E_{\bar{G}}$ is a subset of edges of the form (u_i, w_j) and (u_j, w_i) such that $(i, j) \notin E$, i.e., edge (i, j) does not exist in the original graph G, with costs $c_{ij} = C$ and $d_{ij} = 3$ for all $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, n\}$;

and C is a sufficiently large constant parameter such that $C > n$. Note that in our construction of edge costs we use c_{ij} and d_{ij} instead of $c_{u_i w_j}$ and $d_{u_i w_j}$, respectively, for notational simplicity. Finally, define vertex deletion costs as:

$$a_i = b_j = \frac{1}{2} \quad \forall i,\ j \in \{1, \ldots, n\}, \quad \text{and} \quad a_{n+1} = b_{n+1} = C,$$

where, similar to the notation above, we have a_i and b_j instead of a_{u_i} and b_{w_j}, respectively. Next, we show that there exists a vertex cover of size at most K in G if and only if (2) holds for $\widetilde{G}$.

Assume that there exists a vertex cover of size ℓ in G, $\ell \leq K \leq n$. Without loss of generality, let vertices $\{1, \ldots, \ell\} \subseteq V$ be contained in this cover. Then define $I_D = \{u_1, \ldots, u_\ell\}$ and $J_D = \{w_1, \ldots, w_\ell\}$, i.e., the interdictor removes vertices in U and W that are copies of the vertex cover in G. Note that $V \backslash \{1, \ldots, \ell\}$ forms an independent set in G (recall that an independent set is a set of vertices such that the induced subgraph has no edges). Thus, the follower cannot use edges in E_G for assignments. Then one can easily check that the follower's solution contains edges (u_i, w_i) for $i \in \{\ell + 1, \ldots, n\}$ (because $d_{ii} = 4$) and (u_{n+1}, w_{n+1}). Note that the leader's costs for all these edges are zero, which implies that the objective function value of the leader is given by:

$$\sum_{i \in I_D} \frac{1}{2} + \sum_{j \in J_D} \frac{1}{2} + \sum_{(i,j) \in E} 0 \cdot x_{ij} = \frac{\ell}{2} + \frac{\ell}{2} \leq K,$$

and (2) holds.

Conversely, assume that there exists a feasible solution of **LAIP-D** such that (2) is satisfied. Recall that $K \leq n$ and $C > n$. Thus, vertices u_{n+1} and w_{n+1} are not deleted, i.e., $u_{n+1} \notin I_D$ and $w_{n+1} \notin J_D$; furthermore, the follower's assignment does not contain edges from the set $E_1 \cup E_G \cup E_{\overline{G}}$.

First, we show that $|I_D| = |J_D|$. Assume that this is not the case. Without loss of generality, let $|I_D| > |J_D|$. Thus, the follower cannot match all vertices in $W \backslash \{J_D \cup w_{n+1}\}$ to vertices in $U \backslash \{I_D \cup u_{n+1}\}$. Moreover, $d_{(n+1),(n+1)} = 1 < 2 = d_{(n+1),j}$ for all $j \in \{1, \ldots, n\}$, thus, there exists a vertex from $W \backslash \{J_D \cup w_{n+1}\}$, which is matched by the follower to $u_{n+1} \in \{U \backslash I_D\}$. However, $c_{(n+1),j} = C > K$ and, consequently, using the contradiction argument we have $|I_D| = |J_D|$.

Second, we show that $U \backslash I_D$ and $W \backslash J_D$ contain vertices with the same indices, i.e., if $u_i \in U \backslash I_D$, then $w_i \in W \backslash J_D$ for all $i \in \{1, \ldots, n\}$. Notice that if it is not the case, then at least one edge *not* from $E_0 \cup E_2$ should be used by the follower. However, as mentioned above, such edges have their leader's costs equal to $C > K$, which is not possible by (2).

Third, recall that edges from E_G are not used by the follower as constraint (2) holds in the considered solution of **LAIP-D**. Given the discussion above, it implies that vertices in G that correspond (via indices) to vertices in $U \backslash I_D$ (and $W \backslash J_D$) form an independent set. Therefore, vertices in G that corresponds to I_D (and J_D) form a vertex cover of size at most K in G.

Finally, we note that the proof is valid for both the optimistic and pessimistic cases of **LAIP-D**. □

3 Basic linear MIP models

In the remainder of the paper we make the following additional assumptions:

A2. Optimistic case of **LAIP** is considered.

A3. G is a complete bipartite graph, i.e., $(i, j) \in E$ for all $i \in V$ and $j \in U$.

Assumption **A2** is typical in the bilevel (hierarchical) optimization literature (Colson et al. 2007). Assumption **A3** is introduced for simplicity; however, it is not too restrictive and also appears in the related assignment literature (Burkard et al. 2009).

Next, observe that the follower's optimization problem in **LAIP** may become an unbalanced LAP due to the leader's interdiction actions. However, there exists an equivalent reformulation of **LAIP**, where the follower's problem is balanced. (By "equivalency" in this section we imply that the corresponding models provide the same optimal objective function value.) Specifically, consider the following bilevel problem:

$$\min_{\mathbf{x},\mathbf{y},\mathbf{z},\widetilde{\mathbf{c}},\widehat{\mathbf{d}}} \sum_{i\in V} a_i y_i + \sum_{j\in U} b_j z_j + \sum_{(i,j)\in E} \widetilde{c}_{ij} \tag{3a}$$

$$\text{s.t. } \widetilde{c}_{ij} \le c_{ij}(1-y_i),\ \widetilde{c}_{ij} \le c_{ij}(1-z_j),\ \widetilde{c}_{ij} \le c_{ij}x_{ij} \quad \forall (i,j)\in E, \tag{3b}$$

$$\widetilde{c}_{ij} \ge c_{ij}(x_{ij}-y_i-z_j) \quad \forall (i,j)\in E, \tag{3c}$$

$$\widehat{d}_{ij} \le d_{ij}(1-y_i),\quad \widehat{d}_{ij} \le d_{ij}(1-z_j) \quad \forall (i,j)\in E, \tag{3d}$$

$$\widehat{d}_{ij} \ge d_{ij}(1-y_i-z_j) \quad \forall (i,j)\in E, \tag{3e}$$

$$\widetilde{c}_{ij} \ge 0,\ \widehat{d}_{ij} \ge 0 \quad \forall (i,j)\in E, \tag{3f}$$

$$y_i \in \{0,1\},\ z_j \in \{0,1\} \quad i\in V,\ j\in U, \tag{3g}$$

$$\mathbf{x} \in \operatorname*{argmax}_{\mathbf{x}} \sum_{(i,j)\in E} \widehat{d}_{ij}x_{ij} \tag{3h}$$

$$\text{s.t. } \sum_{j\in U} x_{ij} = 1 \quad \forall i\in V, \tag{3i}$$

$$\sum_{i\in V} x_{ij} = 1 \quad \forall j\in U, \tag{3j}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j)\in E, \tag{3k}$$

where if $i \in V$ or $j \in U$ are removed by the leader then $\widehat{d}_{ij} = \widetilde{c}_{ij} = 0$ by constraints (3b)–(3f); furthermore, $\widetilde{c}_{ij} = c_{ij}$ if and only if $y_i = z_j = 0$ (vertices i and j are not interdicted by the leader) and $x_{ij} = 1$ (the follower selects the corresponding edge in the assignment). Simply speaking, whenever a vertex is removed by the interdictor it is replaced by a "dummy" vertex with the assignment costs equal to 0 for both the follower and the leader. Also, note that constraints (3b) are omitted in the remainder of the paper as the objective in (3a) involves minimization of $\widetilde{c}_{ij}$, $(i, j) \in E$, with a strictly positive coefficient.

It is well-known that the constraint matrix in the follower's problem (3h)–(3k) is totally unimodular. Thus, integrality restrictions in (3k) can be relaxed and the problem reduces to a simple linear program (LP), which can be rewritten in terms of its dual LP as follows:

$$\begin{aligned} &\min \sum_{i\in V} \lambda_i + \sum_{j\in U} \mu_j \\ &\text{s.t.} \quad \lambda_i + \mu_j \ge \widehat{d}_{ij} \quad \forall (i,j)\in E. \end{aligned} \tag{4}$$

Therefore, due to the strong duality property of LPs, the follower's optimization problem (3h)–(3k) can be replaced by a feasibility problem given by:

$$\sum_{(i,j)\in E} \widehat{d}_{ij}x_{ij} = \sum_{i\in V}\lambda_i + \sum_{j\in U}\mu_j, \tag{5a}$$

$$\sum_{j\in U} x_{ij} = 1 \quad \forall\, i \in V, \tag{5b}$$

$$\sum_{i\in V} x_{ij} = 1 \quad \forall\, j \in U, \tag{5c}$$

$$\lambda_i + \mu_j \geq \widehat{d}_{ij} \quad \forall\, (i,j) \in E, \tag{5d}$$

$$x_{ij} \in \{0,1\} \quad \forall\, (i,j) \in E, \tag{5e}$$

which implies that the original bilevel problem can be reduced to an equivalent single-level optimization problem. Observe that *(i)* the equivalency follows by assumption **A2** as there may be multiple optimal solutions of the follower's LAP, and *(ii)* we keep integrality restrictions for $\mathbf{x}$ due to its presence in (3c) of the leader's problem and nonlinearity of (5a).

Subsequently, nonlinear terms $\widehat{d}_{ij}x_{ij}$ can be linearized (recall that x_{ij} is a binary vector) by defining $\widetilde{d}_{ij} = \widehat{d}_{ij}x_{ij}$ and introducing a set of additional linear constraints. This observation results in the following linear mixed 0–1 programming reformulation of the original problem:

$$\min \sum_{i\in V} a_i y_i + \sum_{j\in U} b_j z_j + \sum_{(i,j)\in E} \widetilde{c}_{ij} \tag{6a}$$

$$\text{s.t.} \quad \widetilde{c}_{ij} \geq c_{ij}(x_{ij} - y_i - z_j) \quad \forall\, (i,j) \in E, \tag{6b}$$

$$\widetilde{d}_{ij} \geq d_{ij}(x_{ij} - y_i - z_j) \quad \forall\, (i,j) \in E, \tag{6c}$$

$$\widetilde{d}_{ij} \leq d_{ij}(1 - y_i), \quad \widetilde{d}_{ij} \leq d_{ij}(1 - z_j) \quad \forall\, (i,j) \in E, \tag{6d}$$

$$\widetilde{d}_{ij} \leq d_{ij}x_{ij} \quad \forall\, (i,j) \in E, \tag{6e}$$

$$\sum_{(i,j)\in E} \widetilde{d}_{ij} = \sum_{i\in V}\lambda_i + \sum_{j\in U}\mu_j, \tag{6f}$$

$$\lambda_i + \mu_j \geq d_{ij}(1 - y_i - z_j) \quad \forall\, (i,j) \in E, \tag{6g}$$

$$\lambda_i + \mu_j \geq 0 \quad \forall\, (i,j) \in E, \tag{6h}$$

$$\sum_{j\in U} x_{ij} = 1 \quad \forall\, i \in V, \tag{6i}$$

$$\sum_{i\in V} x_{ij} = 1 \quad \forall\, j \in U, \tag{6j}$$

$$\widetilde{c}_{ij} \geq 0,\ \widetilde{d}_{ij} \geq 0,\ x_{ij} \in \{0,1\} \quad \forall\, (i,j) \in E, \tag{6k}$$

$$y_i \in \{0,1\},\ z_j \in \{0,1\} \quad i \in V,\ j \in U. \tag{6l}$$

Based on the above discussion we conclude:

Proposition 2 *Models* (1) *and* (6) *are equivalent reformulations of* **LAIP**.

Proof The fact that (1) is equivalent to **LAIP** is due to the following simple observation. Specifically, constraints (3d)–(3e) along with nonnegativity restrictions in (3f) ensure that $\widehat{d}_{ij} = 0$ if and only if either $y_i = 1$, or $z_j = 1$, or $y_i = z_j = 1$, i.e., either vertex $i \in V$, or vertex $j \in U$, or both are removed by the interdictor. Otherwise, $\widehat{d}_{ij} = d_{ij}$.

To show that (6) is equivalent to **LAIP** we note that constraints (6c)–(6e) along with nonnegativity restrictions in (6k) enforce $\widetilde{d}_{ij} = \widehat{d}_{ij} x_{ij}$ for all $(i, j) \in E$. Furthermore, it is clear that constraints (5d) can be equivalently represented as (6g)–(6h). The remainder of the proof follows by the construction of (6) using (5), the strong duality property of LPs and assumption **A2**. □

As mentioned above, for LAPs the integrality restriction can be relaxed, which implies that the original problem can be solved as an LP. Clearly, one would expect that binary restrictions for $\mathbf{x}$ in (6) could also be relaxed (i.e., replacing $x_{ij} \in \{0, 1\}$ by simply $x_{ij} \geq 0$ for all $(i, j) \in E$), thus, decreasing the number of binary variables in the model.

Proposition 3 *In MIP* (6) *binary restrictions for* x_{ij}, $(i, j) \in E$, *can be replaced by nonnegativity requirements resulting in an equivalent reformulation.*

Proof The proof is rather straightforward and, therefore, omitted. □

Finally, we should note that the reformulation approach used in this section (based on the idea of replacing the follower's problem by its optimality conditions) is standard in the bilevel optimization literature, where the follower's problem is convex, e.g., a linear program. A classical example for linear bilevel LPs can be found in Audet et al. (1997). The same approach is briefly discussed in Beheshti et al. (2015) in the context of the bilevel assignment problem.

4 Greedy-based construction heuristic

4.1 Basic scheme

The main idea of the proposed approach is fairly simple. At each iteration we iteratively delete (based on a greedy-based scheme described below) exactly one vertex from either V or U until some stopping criteria are satisfied. Thus, at the end of iteration k there are either k or $k - 1$ vertices removed from G. In particular, the former case implies that the algorithm proceeds to iteration $k+1$, while the latter one that the algorithm terminates. Next, we describe the algorithm in more detail.

Denote by $I_k \subseteq V$ and $J_k \subseteq U$ the subsets of vertices in V and U, respectively, that are *not* removed by the interdictor at the beginning of iteration k. For $k = 1$ these sets are defined as $I_1 = V$ and $J_1 = U$, respectively, i.e., no vertices are deleted initially. Then we consider the following problem:

$$f_k \equiv \phi(I_k, J_k) = \min \sum_{i \notin I_k} a_i + \sum_{j \notin J_k} b_j + \sum_{(i,j) \in E:\ i \in I_k, j \in J_k} c_{ij} x_{ij} \tag{7a}$$

$$\text{s.t.} \quad \mathbf{x} \in \arg\max \sum_{(i,j) \in E:\ i \in I_k, j \in J_k} d_{ij} x_{ij} \tag{7b}$$

$$\text{s.t.} \quad \sum_{j \in J_k} x_{ij} \leq 1 \quad \forall\, i \in I_k, \tag{7c}$$

$$\sum_{i \in I_k} x_{ij} \leq 1 \quad \forall j \in J_k, \tag{7d}$$

$$x_{ij} \in \{0, 1\} \quad \forall (i, j) \in E, \tag{7e}$$

which provides a feasible solution of **LAIP** at iteration k given that vertices from $V\setminus I_k$ and $U\setminus J_k$ are removed by the interdictor. Note that f_k denotes the leader's objective function value at iteration k.

At each step of the algorithm a vertex can be removed either from V or U. In order to select one of these options, we apply the following criteria (an additional motivation of our approach is provided in Sect. 4.2). Specifically, for each $i \in I_k$ and $j \in J_k$ we define the following parameters:

$$\widetilde{c}^V_{i,k} = \frac{1}{|J_k|}\sum_{j\in J_k} c_{ij} \quad \text{and} \quad \widetilde{c}^U_{j,k} = \frac{1}{|I_k|}\sum_{i\in I_k} c_{ij}, \tag{8}$$

which can be interpreted as "potential cost contributions" of vertices $i \in I_k$ and $j \in J_k$ to the current objective function value. Thus, the interdictor could consider removing vertices whose cost contributions defined by (8) are larger than the corresponding removal costs. Such subsets of vertices are given by:

$$\widetilde{I}_k = \{i \in I_k : a_i \le \widetilde{c}^V_{i,k}\} \quad \text{and} \quad \widetilde{J}_k = \{j \in J_k : b_j \le \widetilde{c}^U_{j,k}\}. \tag{9}$$

If one assumes that vertices either from $\widetilde{I}_k$ or $\widetilde{J}_k$ are deleted, then the resulting objective functions can be estimated heuristically by one of the corresponding terms below:

$$C^V_k = \sum_{i\in\widetilde{I}_k} a_i + \sum_{i\in I_k\setminus\widetilde{I}_k} \widetilde{c}^V_{i,k} \quad \text{and} \quad C^U_k = \sum_{j\in\widetilde{J}_k} b_j + \sum_{j\in J_k\setminus\widetilde{J}_k} \widetilde{c}^U_{j,k}, \tag{10}$$

where (8) is used for cost contributions of non-deleted vertices in subsets $I_k\setminus\widetilde{I}_k$ and $J_k\setminus\widetilde{J}_k$, respectively.

Next, if $C^V_k \le C^U_k$, then we first attempt removing one of the vertices from $I_k \subseteq V$ as follows. First, we define

$$f^V_k = \min_{i\in I_k} \phi(I_k\setminus\{i\}, J_k) \quad \text{and} \quad i_k \in \arg\min_{i\in I_k} \phi(I_k\setminus\{i\}, J_k). \tag{11}$$

Clearly, if $f^V_k \le f_{k-1}$, i.e., removal of i_k improves the objective function value, then we set $I_{k+1} = I_k\setminus\{i_k\}$ and the algorithm proceeds to the next iteration. Otherwise, we attempt removing one of the vertices from $J_k \subseteq U$ in a similar manner using

$$f^U_k = \min_{j\in J_k} \phi(I_k, J_k\setminus\{j\}) \quad \text{and} \quad j_k \in \arg\min_{j\in J_k} \phi(I_k, J_k\setminus\{j\}). \tag{12}$$

The algorithm proceeds to the next iteration if $f^U_k \le f_{k-1}$. However, if removal of either i_k or j_k does not improve the objective function value, i.e., $f^V_k > f_{k-1}$ and $f^U_k > f_{k-1}$, then the algorithm terminates.

Furthermore, if $C^V_k > C^U_k$, then we repeat the above steps in the opposite order, i.e., we first attempt removing a vertex from U, and, if it does not result in the improved objective function value, removal of a vertex from V is also considered. The pseudocode of the method is summarized in Algorithm 1.

Note that each step of the proposed algorithm requires solution of (7). In particular, if the follower's problem (7b)–(7e) contains multiple optimal solutions for given I_k and J_k, then (7) becomes less trivial (recall that the optimistic case is considered; thus, the follower's solution that also minimizes the leader's objective (7a), is preferable). To handle such situations, we apply the following simple approach. First, a balanced version (with equality constraints) of (7b)–(7e) is solved, e.g., using Hungarian algorithm (Burkard et al. 2009). Next, we attempt increasing one of the non-basic variables (of the corresponding linear assignment problem)

Algorithm 1 Greedy-Based Heuristic (**GBH**)

Initialization ▹ Step 1
$I_1 \leftarrow V$, $J_1 \leftarrow U$, $f_0 \leftarrow \phi(I_1, J_1)$ and $k \leftarrow 1$

Main Cycle ▹ Step k
▹ Compute objectives if k-th vertex is removed from either V or U
$f_k^V \leftarrow \min_{i \in I_k} \phi(I_k \backslash \{i\}, J_k)$ and $i_k \leftarrow \arg\min_{i \in I_k} \phi(I_k \backslash \{i\}, J_k)$
$f_k^U \leftarrow \min_{j \in J_k} \phi(I_k, J_k \backslash \{j\})$ and $j_k \leftarrow \arg\min_{j \in J_k} \phi(I_k, J_k \backslash \{j\})$
▹ Check if vertex removal is "profitable"
if $C_k^V \leq C_k^U$ **then** ▹ First, try to remove from V
  if $f_k^V \leq f_{k-1}$ **then** ▹ Check if removal from V is "profitable"
    $I_{k+1} \leftarrow I_k \backslash \{i_k\}$, $J_{k+1} \leftarrow J_k$, $f_k \leftarrow f_k^V$, $k \leftarrow k+1$ ▹ Go to next step
  else if $f_k^U \leq f_{k-1}$ **then** ▹ Check if removal from U is "profitable"
    $I_{k+1} \leftarrow I_k$, $J_{k+1} \leftarrow J_k \backslash \{j_k\}$, $f_k \leftarrow f_k^U$, $k \leftarrow k+1$ ▹ Go to next step
  else return f_{k-1} ▹ Obtained solution
  end if
end if
if $C_k^V > C_k^U$ **then** ▹ First, try to remove from U
  if $f_k^U \leq f_{k-1}$ **then** ▹ Check if removal from U is "profitable"
    $I_{k+1} \leftarrow I_k$, $J_{k+1} \leftarrow J_k \backslash \{j_k\}$, $f_k \leftarrow f_k^U$, $k \leftarrow k+1$ ▹ Go to next step
  else if $f_k^V \leq f_{k-1}$ **then** ▹ Check if removal from V is "profitable"
    $I_{k+1} \leftarrow I_k \backslash \{i_k\}$, $J_{k+1} \leftarrow J_k$, $f_k \leftarrow f_k^V$, $k \leftarrow k+1$ ▹ Go to next step
  else return f_{k-1} ▹ Obtained solution
  end if
end if

with a zero reduced cost (thus, making it basic) in order to possibly obtain an alternative optimal solution of (7b)–(7e). To avoid enumeration of the exponential (in the worst-case scenario) number of optimal solutions for the lower-level problem, we consider at most $\Theta(n)$ basic feasible solutions of the follower's polyhedral set and choose the one that results in the best objective function value for the upper-level problem. The above procedure implies that (7) may be solved sub-optimally at some iterations of **GBH**. Nevertheless, this approach turns out to be adequate to ensure a reasonably good performance of the proposed heuristic in our computational experiments.

4.2 Special case: unit edge costs

In this section we focus on a special case of **LAIP**, where $c_{ij} = c$ and $d_{ij} = d$ for all $(i, j) \in E$. Moreover, without loss of generality it can be assumed that $c = d = 1$. Define

$$I_V = \{i \in V \mid a_i \leq 1\} \quad \text{and} \quad J_U = \{j \in U \mid b_j \leq 1\},$$

i.e., subsets of vertices in V and U, respectively, whose removal costs are at most one.

Proposition 4 *Let $c_{ij} = d_{ij} = 1$ for all $(i, j) \in E$. Then the optimal objective function value of* **LAIP** *is given by:*

$$f^* = \min \left\{ \sum_{i \in I_V} a_i + (n - |I_V|), \sum_{j \in J_U} b_j + (n - |J_U|) \right\}. \tag{13}$$

Proof Denote by I_D and J_D (I_D^* and J_D^*) subsets of V and U, respectively, removed by the leader in a feasible (optimal) solution of **LAIP**. Similarly, denote by $g(I_D, J_D)$ and $g(I_D^*, J_D^*)$ optimal objective function values of the follower's assignment problem corresponding to subsets of deleted vertices (I_D, J_D) and (I_D^*, J_D^*), respectively.

First, we prove (by contradiction) that there exists an optimal solution such that at least one of the subsets I_D^* and J_D^* is empty, i.e., the leader can remove vertices only either from V or U. Specifically, consider an optimal solution of **LAIP** such that $I_D^* \neq \emptyset$ and $J_D^* \neq \emptyset$. Also, without loss of generality, we assume that $|I_D^*| \leq |J_D^*|$. Clearly, as $c_{ij} = d_{ij} = 1$ for all $(i, j) \in E$, then the objective function value of the optimal solution is given by

$$\phi(U \backslash I_D^*, V \backslash J_D^*) = \sum_{i \in I_D^*} a_i + \sum_{j \in J_D^*} b_j + (n - |J_D^*|),$$

where $\phi(\cdot, \cdot)$ is defined by (7).

Recall that by assumption **A3** graph G is bipartite and complete. Thus,

$$g(\emptyset, J_D^*) = g(I_D^*, J_D^*) = n - |J_D^*|, \tag{14}$$

which holds due to the fact that $c_{ij} = d_{ij} = 1$ for all $(i, j) \in E$ and $|I_D^*| \leq |J_D^*|$. The term $g(\emptyset, J_D^*)$ defines an optimal objective function value of the follower's assignment problem corresponding to deleted vertices $I_D = \emptyset$ and $J_D = J_D^*$.

Consider an alternative feasible solution of the leader with $I_D = \emptyset$ and $J_D = J_D^*$. Then the leader's objective function value satisfies:

$$\begin{aligned} \sum_{j \in J_D^*} b_j + g(\emptyset, J_D^*) &= \sum_{j \in J_D^*} b_j + (n - |J_D^*|) \\ &\leq \sum_{i \in I_D^*} a_i + \sum_{j \in J_D^*} b_j + g(I_D^*, J_D^*), \end{aligned} \tag{15}$$

where the inequality follows by (14) and assumption **A1**. Clearly, (15) implies that the leader's solution with $I_D = \emptyset$ and $J_D = J_D^*$ is also optimal.

Next, assume $I_D^* = \emptyset$ (the case of $J_D^* = \emptyset$ can be considered similarly). In order to establish (13) it is sufficient to show that there exists an optimal solution of **LAIP** such that $J_D^* = J_U$. First, we show that $J_D^* \subseteq J_U$. In particular, assume that there exists $w \in J_D^* \backslash J_U$ such that $b_w > 1$. Then the corresponding objective function satisfies:

$$\begin{aligned} \phi(V, U \backslash J_D^*) = \sum_{j \in J_D^*} b_j + g(\emptyset, J_D^*) &= \sum_{j \in J_D^* \backslash \{w\}} b_j + b_w + (n - |J_D^*|) \\ > \sum_{j \in J_D^* \backslash \{w\}} b_j + (n + 1 - |J_D^*|) &= \phi(V, U \backslash \left\{J_D^* \backslash \{w\}\right\}), \end{aligned} \tag{16}$$

which contradicts optimality of J_D^*. The last equality follows by assumption **A3** and the fact that $c_{ij} = d_{ij} = 1$ for all $(i, j) \in E$.

Finally, in a similar manner it can be established that there exists J_D^* such that $J_U \subseteq J_D^*$, which completes the proof. □

The discussion above can be used to provide an intuitive motivation for computing parameters C_k^V and C_k^U given by (10). In particular, consider an instance of **LAIP** with $c_{ij} = d_{ij} = 1$ for all $(i, j) \in E$ such that $a_i = \epsilon$ for all $i \in V$, $b_1 = \frac{\epsilon}{2}$ and $b_j > 1$ for all $j \in U \backslash \{1\}$, where $0 < \epsilon < 1$.

First, we focus on a simple modification of **GBH**, where C_k^V and C_k^U are not exploited and the interdictor simply removes a vertex either from V or U, namely, the one, which provides the best improvement with respect to the objective function value using (11)–(12). It is easy to verify that at the first iteration deletion of node "1" from U with $b_1 = \epsilon/2$ is preferable and results in the objective function $f_1 = \epsilon/2 + (n-1)$. Furthermore, at the very next iteration the algorithm terminates as deletion of the remaining vertices from U is "too expensive," while deletion of any vertex from V does not improve the objective function value.

On the other hand, in the first iteration of **GBH** we have $\widetilde{I}_1 = V$ and $\widetilde{J}_1 = \{1\}$. Thus, $C_1^V = n\epsilon$ and $C_1^U = \epsilon/2 + (n-1)$, which implies that the algorithm removes a vertex from V at the very first iteration. Consequently, **GBH** terminates only after all vertices from V are removed resulting in the objective function value $f_n = n\epsilon$. Clearly, this solution is optimal. Moreover, we observe that

$$\frac{\frac{\epsilon}{2} + (n-1)}{n\epsilon} \xrightarrow[\epsilon \to +0]{} +\infty,$$

which implies that the difference in the performance of these two greedy-based heuristics (namely, **GBH** and its simplified version) can be substantial. Therefore, comparing the values of C_k^V and C_k^U at each iteration k allows the heuristic to remove a vertex from the "proper" partition of G. Finally, we show next that **GBH** obtains an optimal solution for graphs with unit edge costs.

Proposition 5 *Let $c_{ij} = d_{ij} = 1$ for all $(i, j) \in E$. If f_k^V and f_k^U are computed exactly at each iteration k of Algorithm 1, then* **GBH** *obtains an optimal solution of* **LAIP**.

Proof Consider the first iteration of **GBH**. Similar to the discussion above, one can verify that $\widetilde{I}_1 = I_V$ and $\widetilde{J}_1 = J_U$ by (9). Thus,

$$C_1^V = \sum_{i \in I_V} a_i + (n - |I_V|) \quad \text{and} \quad C_1^U = \sum_{j \in J_U} b_j + (n - |J_U|). \tag{17}$$

Assume that $C_1^V \le C_1^U$. Then the algorithm removes a vertex from I_V at the first iteration. Moreover, one can show that for $k \ge 1$ we have $C_{k+1}^V \le C_k^V$, while $C_{k+1}^U = C_k^U$. Therefore, **GBH** terminates only after all vertices from I_V are removed. Clearly, the obtained result is optimal by (17) and Proposition 4. The case of $C_1^V > C_1^U$ can be shown similarly. □

5 Computational experiments

5.1 Setup and test instances

We consider three classes of test instances (referred to as **T1**, **T2** and **T3**), where the input data (namely, parameters a_i, b_j, c_{ij} and d_{ij}) is generated according to a uniform distribution over intervals [0, 10], [0, 50], [0, 100], respectively. We consider $n \in \{10, 11, \ldots, 29, 30\}$. For each class and the problem's size we report average results over 5 randomly generated instances. All computational experiments are conducted using a laptop with an Intel(R) Core(TM)2 Quad 2.83 GHz CPU and 4 GB of RAM. For solving MIP models we use Gurobi Optimizer 5.5, which is one of the state-of-the-art commercial MIP solvers.

Table 1 Estimated approximation factors of **GBH**, i.e., f^{H}/f^{*}, where f^{H} and f^{*} denote the objective function values found by **GBH** and the exact MIP solver (thus, optimal), respectively

n	T1	T2	T3
10	1.074	1.173	1.072
11	1.000	1.142	1.260
12	1.095	1.127	1.070
13	1.140	1.208	1.049
14	1.123	1.134	1.103
15	1.068	1.229	1.132
Total	1.087	1.171	1.108

Table 2 Estimated ratio $f^{\mathrm{MIP}}/f^{\mathrm{H}}$, where f^{MIP} denotes the objective function value of a feasible solution found by the exact MIP solver at the time of **GBH** termination

$n \in$	T1	T2	T3
{11, 12, 13, 14, 15}	1.061	1.035	1.077
{16, 17, 18, 19, 20}	1.090	1.377	1.310
{21, 22, 23, 24, 25}	1.005	1.287	1.302
{26, 27, 28, 29, 30}	0.897	1.173	1.273
Total	0.973	1.229	1.268

Table 3 Estimated ratio $t^{\mathrm{MIP}}/t^{\mathrm{H}}$, where t^{H} denotes the total running time of **GBH** and t^{MIP} denotes the running time necessary for the exact solver to find a solution, which is at least as good as the one found by **GBH**

$n \in$	T1	T2	T3
$\{11, 12, \ldots, 20\}$	29.546	136.860	909.364
$\{21, 22, \ldots, 29, 30\}$	1.521	82.202	151.715
Total	2.152	82.752	155.794

5.2 Results and discussion

In all the tables (namely, Tables 1, 2, 3) we report ratios, which compare the results (including the objective function values and running times) obtained by the proposed greedy-based heuristic **GBH** and the exact MIP solver with model (6). Such ratios are computed using a standard linear regression approach as briefly described below.

Generally speaking, consider a sequence of experimental data points (r_k, s_k), $k = 1, \ldots, K$, where K denotes the total number of performed experiments. If one wants to find approximate coefficient α such that dependence between two samples can be expressed in terms of $\alpha = s_k/r_k$ for each k, then the least squares method can be used to minimize the objective function of the form $F(\mathbf{r}, \mathbf{s}) = \sum_{k=1}^{K} (s_k - \alpha r_k)^2$, thus, providing an estimate of α by $\bar{\alpha}$ as follows:

$$\bar{\alpha} = \frac{\sum_{k=1}^{K} r_k s_k}{\sum_{k=1}^{K} r_k^2}.$$

Such estimation approach is used in all our experiments discussed next. Finally, we note that the row denoted by 'Total' in each of our tables indicates that all data points in the considered classes of test instances are used to estimate the corresponding values in each column.

Experiments in Table 1 In this set of experiments we compare the solutions obtained by **GBH** versus optimal solutions computed using the exact solver with MIP model (6). The experiments are performed for $n \in \{10, \ldots, 15\}$ only, because the required running times of the exact solver for $n > 15$ become prohibitively large (typically, more than a day for most instances). However, from the reported experiments it is rather clear that the experimental approximation factor of **GBH** is sufficiently good, i.e., the obtained errors are on average 8.7 % for the test instance class **T1**, 17.1 % for **T2** and 10.8 % for **T3**.

Experiments in Table 2 When comparing results of any heuristic versus an exact solver, one could argue that their running times differ substantially. Indeed, heuristic methods are much faster than their exact counterparts. However, exact solvers often provide good feasible solutions sufficiently fast. Thus, one natural question arising in this context is as follows: What is the quality of the final heuristic solution in comparison to a temporary feasible solution (if available) obtained by the exact solver at the time of the heuristic termination? We report the corresponding results in Table 2.

One observation is that the results for **GBH** are disappointing for **T1**. We attribute this to the fact that the range of [0, 10] used for generating instances in this class is rather small, which makes the problem easier for the exact solver. Recall that, as discussed in Sect. 4.2, instances with unit edge costs can be solved easily in linear time. Thus, the above observation is rather intuitive. Moreover, commercial MIP solvers employ heuristic methods for computation of initial feasible solutions. Admittedly, such heuristics typically do not exploit the structure of the underlying problem available to us; however, for easier problems they often provide solutions with a reasonably good quality (as in the case of **T1**). Nevertheless, for classes **T2** and **T3** the proposed heuristic performs better than the MIP solver.

Experiments in Table 3 One could argue that the results reported in Table 2 are not particularly impressive (improvements provided by the heuristic are about 10–30 %). However, we can address this concern using the following question: What is the running time necessary for the exact MIP solver to find a solution, which is at least as good as the one found by **GBH**? This also provides a lower bound on the difference in the total running times between the heuristic and the exact solver. The corresponding results are reported in Table 3.

As expected, the obtained results confirm our earlier observations regarding instances in **T1**, namely, the fact that they can be tackled easily by the exact solver. However, the results for classes **T2** and **T3** are very encouraging. Despite relatively small improvement (about 10–30 %) in the objective function values obtained by **GBH** (as indicated in Table 2), the running times necessary for the exact MIP solver to find solutions, which are at least as good as the ones found by the proposed greedy heuristic, are rather large. Thus, we conclude that despite its relative simplicity, the proposed heuristic approach can be used to obtain good quality solutions for **LAIP**.

6 Concluding remarks

In this paper we consider a class of interdiction problems referred to as the linear assignment interdiction problem. We briefly discuss some related computational complexity issues and describe an equivalent linear MIP formulation of the problem. More importantly, we also

propose a simple greedy-based construction heuristic approach. Besides providing (under some mild conditions) an optimal solution in the case of unit edge costs for both the leader and the follower, the proposed heuristic method also compares favorably against a commercial MIP solver in our computational experiments with randomly generated test instances.

In terms of future research, we note that our method can be used as a construction phase within any advanced meta-heuristic approach, e.g., GRASP (Feo and Resende 1995; Resende and Ribeiro 2011). If implemented, this enhancement should substantially improve the quality of the obtained solutions, in particular, for large-scale problems.

Acknowledgments This material is based upon work supported by AFRL Mathematical Modeling and Optimization Institute. The research of the first three authors is also partially supported by grants from AFOSR. In addition, we are grateful to Dr. Behdad Beheshti for his helpful comments. Finally, the authors would like to thank anonymous referees whose constructive comments resulted in the improvements to this paper.

References

Audet, C., Hansen, P., Jaumard, B., & Savard, G. (1997). Links between linear bilevel and mixed 0–1 programming problems. *Journal of Optimization Theory and Applications*, *93*(2), 273–300.

Bard, J. F. (1998). *Practical bilevel optimization*. Dordrecht: Kluwer.

Beheshti, B., Özaltın, O. Y., Zare, M. H., & Prokopyev, O. A. (2015). Exact solution approach for a class of nonlinear bilevel knapsack problems. *Journal of Global Optimization*, *61*(2), 291–310.

Beheshti, B., Prokopyev, O.A., & Pasiliao, E.L. (2015). Exact solution approach for the bilevel assignment problem. *Computational Optimization and Applications*, 2015. Accepted for publication.

Brown, G., Carlyle, M., Salmeron, J., & Wood, K. (2006). Defending critical infrastructure. *Interfaces*, *36*, 530–544.

Burkard, R., Dell'Amico, M., & Martello, S. (2009). *Assignment problems*. Philadelphia: Society for Industrial Mathematics.

Colson, B., Marcotte, P., & Savard, G. (2007). An overview of bilevel optimization. *Annals of Operations Research*, *153*(1), 235–256.

Dempe, S. (2002). *Foundations of bilevel programming*. Dordrecht: Kluwer.

Deng, X. (1998). Complexity issues in bilevel linear programming. In A. Migdalas, P. M. Pardalos, & P. Varbrand (Eds.), *Multilevel optimization: Algorithms and applications* (pp. 149–164). Dordrecht: Kluwer.

Feo, T. A., & Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of global optimization*, *6*(2), 109–133.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. New York: W.H. Freeman.

Gassner, E., & Klinz, B. (2009). The computational complexity of bilevel assignment problems. *4OR*, *7*(4), 379–394.

Migdalas, A., Pardalos, P. M., & Värbrand, P. (1998). *Multilevel optimization: Algorithms and applications*. Norwell: Kluwer.

Nemhauser, G. L., & Wolsey, L. A. (1988). *Integer and Combinatorial Optimization*. NewYork, NY: Wiley.

Resende, M. G. C., & Ribeiro, C. C. (2011). Restart strategies for grasp with path-relinking heuristics. *Optimization Letters*, *5*(3), 467–478.

Shen, S., Smith, J. C., & Goli, R. (2012). Exact interdiction models and algorithms for disconnecting networks via node deletions. *Discrete Optimization*, *9*(3), 172–188.

Wood, K. R. (1993). Deterministic network interdiction. *Mathematical and Computer Modelling*, *17*, 1–18.

Zenklusen, R. (2010). Matching interdiction. *Discrete Applied Mathematics*, *158*(15), 1676–1690.