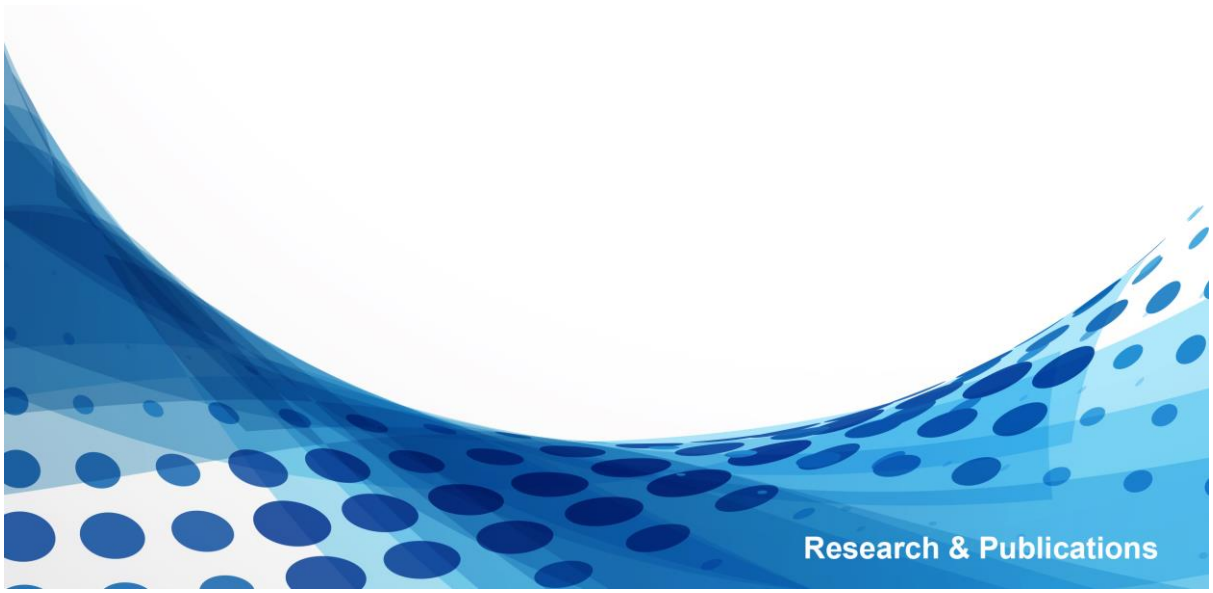
INDIAN INSTITUTE OF MANAGEMENT AHMEDABAD

IIMA
Working Paper

Bilevel Optimization: Applications, Models and Solution Approaches

Sachin Jayaswal
Ankur Sinha



Research &
Publications
W. P. No. 2022-05-02

Bilevel Optimization: Applications, Models and Solution Approaches

Sachin Jayaswal
Ankur Sinha

May 2022

INDIAN INSTITUTE OF MANAGEMENT AHMEDABAD

Bilevel Optimization: Applications, Models and Solution Approaches

Sachin Jayaswal, Ankur Sinha

Production & Quantitative Methods, Indian Institute of Management Ahmedabad, India 380015

Abstract

Bilevel optimization is a difficult class of optimization problems, which contain an inner optimization problem as a constraint to an outer optimization problem. Such optimization problems are commonly referred to as Stackelberg games in the area of game theory, where a hierarchical interaction between a leader and a follower is modeled. This chapter presents several examples of bilevel optimization problems arising in various contexts, e.g., the product line selection problem and the shortest path interdiction problem. Depending on the context of the problem, the leader and the follower may have the same objective function but with conflicting objectives (max-min in the shortest path interdiction), or may have different objective functions (as in the product line selection problem). Under this hierarchical setting, the leader tries to optimize its own decision by taking into account the rational response of the follower. A bilevel optimization problem is NP-hard even in the simplest case in which the problems of the leader and the follower are both simple linear programs. This chapter discuses classical solution approaches that are based on the reformulation of the bilevel problem into a single level. It also discusses several alternate single-level reformulations for the application problems considered in this chapter.

Keywords: `Bilevel optimization, shortest path interdiction, product line design`

1. Introduction

The interest in bilevel programming has been rising over the past few decades, primarily because of new applications that are being identified in various domains. For instance, one of the applications of bilevel programming that is receiving attention is in the area of machine learning, where bilevel optimization provides a way to optimize hyperparameters in the problem. The hyperparameters can appear in various forms, like, network architecture, regularization parameters, and learning rate, among others. Some of the studies that have attempted a bilevel approach to handling hyperparameters are Bennett et al. (2008); Sinha, Khandait and Mohanty (2020). The hyperparameter optimization problem is also important for researchers in the other domain, for example, in the area of evolutionary computation as well, the methods contain hyperparameters, a good choice for which

impact algorithm performance (Sinha, Malo, Xu and Deb, 2014). Another example is from the area of transportation, in the context of a toll-setting problem, that is inherently a bilevel program. Consider a large network of highways operated by the government or a private entity that desires to optimize the tolls for maximizing its revenues. A high amount of the toll would deter users from using the network, thereby reducing revenues due to low volume, while a low amount of the toll would increase the volume but may not be optimal for maximizing revenues. Under such a situation, the government cannot write its own optimization problem by ignoring the highway users, rather a two-level optimization problem that is hierarchical in nature has to be written. In such a problem the upper level is the government that has the objective to maximize revenues under a set of constraints, with an additional constraint being represented by the highway users' optimization problem. The highway users' optimization problem consists of an objective function that minimizes the generalized cost for highway-users, which may include travel distance, travel time and toll amount. Solving the highway-users optimization problem allows the government to figure out the response of the users for any value of toll that is set, which in turn allows the government to compute its revenues. In this problem, the upper level is the government that is often referred to as the leader, and the lower level is the set of highway users that is referred to as the follower. Such scenarios with a leader-follower interaction within an optimization problem are studied in the area of game theory and economics as Stackelberg games (Stackelberg, 1952). For a further reading on leader-follower games in the transportation policy literature, the readers may look at Migdalas (1995); Brotcorne et al. (2001); Sinha et al. (2015).

Applications of bilevel optimization exist in almost all the domains, for example in engineering, bilevel problems are common in structural optimization (Christiansen et al., 2001; Sobieszczanski-Sobieski et al., 2000), where the design variables in an integrated system are decomposed into upper level (system level) design variables and lower level (sub-system level) design variables. Similarly, in chemical engineering the engineers have to decide the optimal state variables and quantity of reactants, and for them the lower level optimization problem appears as an equilibrium condition requiring entropy functional minimization. Bilevel problems are common in the area of defense as well, where they appear as attacker-defender, or defender-attacker problems (Brown et al., 2005; Wein, 2009; Ramamoorthy et al., 2017, 2018; Nigudkar et al., 2021; Bhatt et al., 2021). In the area of business, these problems are common in designing optimal tax policies (Labbé et al., 1998; Sinha et al., 2013), investigation of strategic behavior in deregulated markets (Hu and Ralph, 2007), agri-business management (Whittaker et al., 2017; Bostian et al., 2015), and supply chain and location problems (Küçükaydin et al., 2011; Sun et al., 2008). The readers may refer to review papers and books for additional information about the bilevel methods and applications (Sinha et al., 2018; Dempe, 2002;

Bard, 1998).

General bilevel problems are NP-hard (Vicente et al., 1994), and the linear bilevel problems with continuous variables are no simple, which are often reduced to a mixed integer linear program (MILP) with a BigM. One of the most straightforward ways for handling bilevel programs has been the Karush-Kuhn-Tucker (KKT) approach, where the lower level optimization problem is replaced by the set of its KKT constraints to reduce the bilevel program to a single level problem that can be handled by standard solvers. However, the KKT approach can be used only in special cases, where the lower level bilevel problem adheres to certain regularity conditions, and additionally even when the regularity conditions at the lower level hold, the reduced single-level formulation may not be an easy problem to handle. Linear bilevel programs (Wen and Hsu, 1991; Ben-Ayed, 1993; Bard and Falk, 1982; Fortuny-Amat and McCarl, 1981; Tuy et al., 1993) and quadratic bilevel problems (Bard and Moore, 1990; Edmunds and Bard, 1991; Al-Khayyal et al., 1992) have been widely solved using this approach. Other aproaches to handle bilevel problems include gradient descent (Savard and Gauvin, 1994; Vicente et al., 1994), trust-region (Liu et al., 1998; Marcotte et al., 2001; Colson et al., 2005), and penalty (Aiyoshi and Shimizu, 1981, 1984; Ishizuka and Aiyoshi, 1992; White and Anandalingam, 1993) methods. Given the difficulty in handling bilevel programs, researchers and practitioners have often resorted to solving bilevel problems using computationally intensive approaches, like, nested search. In such an approach the lower level optimization problem is solved corresponding to a large number of upper level decision vectors and a bilevel level solution is *searched*. Evolutionary algorithms have been the most widely used approach for implementing nested search (Li et al., 2006; Sinha, Malo, Frantsev and Deb, 2014; Angelo et al., 2013; Islam et al., 2017). Recently, the algorithm development on bilevel optimization is focused on exploiting two important mappings in bilevel optimization; namely, the reaction set mapping and the lower level optimal value function mapping (Sinha and Shaikh, 2021; Sinha, Lu, Deb and Malo, 2020; Sinha et al., 2017; Angelo et al., 2014). The details of these two mappings have been discussed later in this chapter.

The focus of the chapter is on two classes of bilevel optimization problems, i.e., the product line selection problem and the shortest path interdiction problem. However, we begin the chapter by discussing about the basics of bilevel optimization that will be useful in solving the two classes of optimization problems. The structure of the chapter is as follows. In Section 2, a general bilevel problem is discussed along with its various formulations that can be written with the help of different mappings. This is followed by Sections 3 and 4, where the product line selection problem and the shortest path interdiction problem have been discussed along with the solution methodology, respectively. Finally, we conclude in Section 5. The chapter also includes the AMPL codes to handle the two classes of

optimization problems. The code and the ideas in this chapter can be adapted to solve various classes of bilevel optimization problems.

2. An overview of bilevel optimization

A general bilevel program with the upper-level objective function $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ and lower-level objective function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is given as follows:

$$\text{“}\min_{x,y}\text{”} \quad F(x,y) \tag{1}$$

$$\text{subject to} \tag{2}$$

$$y \in \operatorname*{argmin}_y \{f(x,y) : g_i(x,y) \leq 0, i = 1,\ldots,I, h_j(x,y) = 0, j = 1,\ldots,J\} \tag{3}$$

$$G_k(x,y) \leq 0, \quad k = 1,\ldots,K \tag{4}$$

$$H_l(x,y) = 0, \quad l = 1,\ldots,L \tag{5}$$

where $g_i : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ represents the set of lower level inequality constraints, $h_j : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ represents the set of lower level equality constraints, $G_k : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ represents the set of upper level inequality constraints, and $H_l : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ represents the set of upper level equality constraints. The upper and lower level variables, x and y, in the above definition may be continuous, integers, or mixed-integers.

The reason for using double quotes (“$\min_{x,y}$”) in the above definition is to highlight that the formulation is ill-defined. In the context of multiple lower level optimal solutions corresponding to a given upper level decision vector, the above formulation does not clearly define that which solution from the lower level optimal set should be considered by the upper level. Let us first define the lower level optimality set before we discuss how the ambiguity is handled. Let $\Psi : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ be a set-valued mapping that defines the solution set of the lower level decision maker for every upper level decision vector:

$$\Psi(x) = \operatorname*{argmin}_y \{f(x,y) : g(x,y) \leq 0, h(x,y) = 0\}$$

This ambiguity on the choice of the solution to be chosen from $\Psi(x)$ for any given x can be handled by specifying the position assumed by the upper level decision maker. Below we discuss the two common positions that the upper level decision maker assumes in the presence of multiple lower level optimal solutions for one or more upper level vectors.

2.1. Optimistic Position

If the upper level is optimistic that the lower level will cooperate and choose that solution from its optimal set that is most favourable for the upper level, it is referred to as the optimistic formulation. In an optimistic position, the choice function of the follower is given as follows:

$$\psi_{\mathrm{o}}(x) = \underset{y}{\operatorname{argmin}}\{F(x,y) : y \in \Psi(x)\}$$

that leads to the following bilevel formulation.

$$\begin{aligned} \min_{x,y} \quad & F(x,y) \\ \text{subject to} \quad & \\ & y = \psi_{\mathrm{o}}(x) \\ & G(x,y) \leq 0 \\ & H(x,y) \leq 0 \end{aligned}$$

Optimistic position being more tractable as compared to the pessimistic position is more commonly studied as compared to the pessimistic formulation.

2.2. Pessimistic Position

If the upper level is pessimistic that the lower level may choose that solution from its optimal set that is least favourable for the upper level, it is referred to as the pessimistic formulations. In a pessimistic position, the choice function of the follower is given as follows:

$$\psi_{\mathrm{p}}(x) = \underset{y}{\operatorname{argmax}}\{F(x,y) : y \in \Psi(x)\}$$

that leads to the following bilevel formulation.

$$\begin{aligned} \min_{x,y} \quad & F(x,y) \\ \text{subject to} \quad & \\ & y = \psi_{\mathrm{p}}(x) \\ & G(x,y) \leq 0 \\ & H(x,y) \leq 0 \end{aligned}$$

Unless, specifically mentioned in the context of bilevel programs, an optimistic position is commonly assumed. There are intermediate formulations possible as well based on the choice function of the lower level decision maker in case of multiple optimal solutions. If the lower level optimization problem is known to have only single optimal solution for all possible upper level decision vectors, these alternative formulations do not arise.

2.3. Single-level Reformulations

In this section, we discuss various single level formulations of a bilevel program. In all these discussions, we assume an optimistic position for the upper level decision maker.

2.3.1. Using set-valued mapping

Using $\Psi(x)$ that defines the solution set of the lower level decision maker for every upper level decision vector, the bilevel optimization problem can be expressed as a constrained optimization problem as follows:

$$
\begin{aligned}
&\text{“}\min_{x,y}\text{”} \quad F(x,y) \\
&\text{subject to} \\
&\quad y \in \Psi(x) \\
&\quad G_k(x,y) \leq 0, \quad k = 1,\ldots,K \\
&\quad H_l(x,y) = 0, \quad l = 1,\ldots,L
\end{aligned}
$$

where Ψ is a parameterized range-constraint for the lower-level decision vector y. The Ψ mapping is unknown *a priori* and some algorithms rely on estimation of this mapping to solve the bilevel optimization problem.

2.3.2. Using Lower Level Karush-Kuhn-Tucker Conditions

When the lower level optimization problem adheres to certain regularity conditions and is a convex optimization problem, it can be replaced by its Karush-Kuhn-Tucker (KKT) conditions in the bilevel program. The KKT conditions help in reducing the bilevel optimization problem to a single-level constrained optimization problem. The problem in (1-5) can be written in the following form, when the convexity and regularity conditions for the lower level problem are met:

$$
\min_{x,y,\lambda,\mu} \quad F(x,y) \tag{6}
$$

subject to (7)

$$\nabla_y L(x, y, \lambda, \mu) = 0 \tag{8}$$

$$g_i(x, y) \leq 0, i = 1, \ldots, I \tag{9}$$

$$h_j(x, y) = 0, j = 1, \ldots, J \tag{10}$$

$$G_k(x, y) \leq 0, k = 1, \ldots, K \tag{11}$$

$$H_l(x, y) = 0, l = 1, \ldots, L \tag{12}$$

$$\lambda_i g_j(x, y) = 0, i = 1, \ldots, I \tag{13}$$

$$\lambda_i \geq 0, i = 1, \ldots, I \tag{14}$$

where

$$L(x, y, \lambda) = f(x, y) + \sum_{i=1}^{I} \lambda_i g_i(x, y) + \sum_{j=1}^{J} \mu_j h_j(x, y) \tag{15}$$

The KKT conditions appear as Lagrangian and complementarity constraints. Therefore, the above formulation, may not necessarily be simple to handle. The Lagrangian constraints often lead to non-convexities and the complementarity conditions are inherently combinatorial. The combination of the two may render the single-level optimization problem into a highly non-linear mixed integer program. It is common to linearize the complementary slackness conditions (13) with the help of combinatorial variables (u_i) and a large number (M) as follows:

$$\lambda_i \leq M u_i \qquad i = 1, \ldots, I$$

$$g_i(x, y) \geq -M(1 - u_i) \qquad i = 1, \ldots, I$$

The above two constraint sets replace (13), as along with (9) and (14) they ensure at least one of the two product terms (λ_i or $g_i(x, y)$) to be zero. The complementarity conditions can also be handled as Special Ordered Sets (SOS) (Beale and Tomlin, 1970) in the modern solvers.

2.3.3. Using Lower Level Optimal Value Function

The lower level optimal value function (φ mapping) is commonly used to write an alternate formulation for the general bilevel optimization problem. The optimal value function $\varphi : \mathbb{R}^n \to \mathbb{R}$ is given as follows:

$$\varphi(x) = \min_{y \in Y} \{f(x, y) : y \in \Omega(x)\}$$

This mapping provides the optimal lower level function value for any given upper level decision vector. Using the φ-mapping, the bilevel problem can be written as follows (Ye and Zhu, 2010):

$$\begin{aligned}
&\min_{x,y} \quad F(x,y) \\
&\text{subject to} \\
&\quad f(x,y) \leq \varphi(x) \\
&\quad g_i(x,y) \leq 0, \quad i = 1,\ldots,I \\
&\quad h_j(x,y) = 0, \quad j = 1,\ldots,J \\
&\quad G_k(x,y) \leq 0, \quad k = 1,\ldots,K \\
&\quad H_l(x,y) = 0, \quad l = 1,\ldots,L
\end{aligned}$$

Since the value function is almost never known *a priori*, most of the algorithms that rely on using this mapping estimate it during the iterations of the algorithm. The estimation of this mapping, which is actually a function, is often easier than the $\Psi(x)$ mapping that for any given x returns a set of vectors.

2.3.4. Using Duality-based Approach

For a convex lower level program that satisfies strong duality (i.e., Slater's constraint qualification), the bilevel program can be reformulated into a single level program using the duality of the lower level optimization problem. In this section, we demonstrate this approach on a general linear bilevel program, but the ideas can be extended to other classes of bilevel problems that satisfy convexity and strong duality conditions. Consider a general bilevel linear program with continuous variables:

$$\min_{x,y} \; c_x^\top x + c_y^\top y \tag{16}$$

$$\text{subject to} \tag{17}$$

$$A_x x + A_y y \geq a \tag{18}$$

$$y \in \operatorname*{argmin}_y \left\{ d^\top y : B_x x + B_y y \geq b \right\} \tag{19}$$

with $c_x \in \mathbb{R}^n, c_y, d \in \mathbb{R}^m, A_x \in \mathbb{R}^{K\times n}, A_y \in \mathbb{R}^{K\times m}$, and $a \in \mathbb{R}^K$, $B_x \in \mathbb{R}^{I\times n}, B_y \in \mathbb{R}^{I\times m}$, and $b \in \mathbb{R}^I$. We have omitted the linear term corresponding to the upper level variable in the lower level objective function without any loss of generality. First, let us write the KKT-based single level reduction of the

above problem.

$$\min_{x,y,\lambda} c_x^\top x + c_y^\top y \tag{20}$$

$$\text{s.t. } A_x x + A_y y \geq a \tag{21}$$

$$B_x x + B_y y \geq b \tag{22}$$

$$B_y^\top \lambda = d \tag{23}$$

$$\lambda_i \left(B_x + B_y y - b\right)_i = 0 \quad \forall\, i = 1, \ldots, I \tag{24}$$

$$\lambda \geq 0 \tag{25}$$

The bilinear non-linearity in the complementary slackness conditions can be handled as discussed in Section 2.3.2. Next, let us write the single-level reformulation of the linear bilevel program in (16-19) using the duality-based approach. The dual of the lower level linear program is given as follows:

$$\max_{\lambda} (b - B_x x)^\top \lambda \quad \text{s.t.} \quad B_y^\top \lambda = d, \lambda \geq 0$$

Using the principle of weak duality, the following is valid for every feasible primal-dual pair (y, λ).

$$d^\top y \geq (b - B_x x)^\top \lambda$$

Using strong duality, any feasible pair that satisfies the following condition is an optimal solution to the linear program.

$$d^\top y \leq (b - B_x x)^\top \lambda$$

Using the above, the linear bilevel program in (16-19), can be reformulated as a single level problem as follows:

$$\min_{x,y,\lambda} c_x^\top x + c_y^\top y \tag{26}$$

$$\text{s.t. } A_x x + A_y y \geq a, B_x x + B_y y \geq b \tag{27}$$

$$B_y^\top \lambda = d, \lambda \geq 0 \tag{28}$$

$$d^\top y \leq (b - B_x x)^\top \lambda \tag{29}$$

The difference between (20-25) and (26-29) is that the set of complementary conditions in (24) has been replaced with the strong duality constraint (29). The constraint set (29) in the duality-

based single level reformulation is a product of variables x and λ, and is therefore bilinear in nature. Such bilinear constraints are not easy to linearize, if both x and λ are continuous, when compared to the complementary slackness conditions. One often resorts to McCormick envelopes to handle such bilinear constraints.

Linear bilevel programs where upper level variables are mixed integers can also be handled using the duality based approach in a manner similar to what we have discussed above. In fact, when x is combinatorial, linearization of the bilinear constraint can be achieved easily. The duality-based method is often a preferred approach for a number of classes of bilevel problems, for instance, the minimax optimization problems.

3. Product Line Selection

New products are commonly introduced in several configurations so as to appeal to different market segments (McBride and Zufryden, 1988). For example, a refrigerator may be introduced in various capacities (95 liters, 420 liters, 570 liters, etc.), number of doors (single or double) and colors (white, grey, red, etc.). Capacity, number of doors, and color are three of the several attributes based on which a refrigerator can be differentiated for the different market segments. Thus, each configuration of refrigerator can be viewed as a bundle of attributes (e.g, capacity) fixed at some levels (e.g., 570 liters). Using just these three attributes, and just two possible levels for each, gives rise to 8 possible configurations. Developing each of these configurations entails some fixed cost. Moreover, developing too many variants can cannibalize each other's sales with the result that some of them may fail to recoup through sales the investments in their development. Firms, therefore, face the problem of selecting a subset from the set of alternative product configurations. This problem is commonly referred to as the product line selection (PLS) problem.

Let us understand the problem using an illustrative example. Assume that a firm can introduce a product in 10 different configurations (bundles of attributes), the development costs for which are given in Table 1. The table also provides profit per unit sale of each of these configurations. The firm has to decide a subset of these configurations to develop so as to maximize its profit from their sales. The sales of each configuration will depend on the customers' preferences among the set of all the configurations available to them. For this, firms need to predict which one among a set of available configurations will a given customer buy. Conjoint analysis makes this possible by estimating the potential customers' "value system", i.e., part-worth utility corresponding to every level of each of the attributes of a product

(Green and Krieger, 1985; Dolan, 2001).[1] Further, different customer segments can be identified by performing clustering analysis on the customers' value systems. Suppose the conjoint analysis and the subsequent cluster analysis produce the utility matrix as shown in Table 1 for the 5 customer segments identified. Given this utility matrix, the firm can predict which product configuration will be purchased by a given customer segment. There are two alternate rules to translate the utility matrix into the customers' choice predictions: (i) first (deterministic) choice rule; (ii) probabilistic choice rule (Dolan, 2001). In the first choice rule, a customer is assumed to always buy the product configuration that gives her the highest utility among all the configurations available, provided its utility is no less than her reservation utility[2]. For example, if all the 10 product configurations in Table 1 are developed by the firm, then given the utility matrix in Table 1, customer segment 1 will choose product 8, while customer segment 2 will choose product 3. If we assume the customers' buying decisions are governed by the first choice rule, and the size of each customer segment is as given in the last row of Table 1, then which of these 10 product configurations should the firm develop to maximize its total profit?[3] This is the question the firm's problem of PLS tries to answer. PLS can also be studied using the probabilistic choice rule wherein the probability of buying a particular configuration is proportional to its utility. In the rest of the chapter, the discussion is restricted to the first choice rule.

Table 1: Product line selection parameters

Firms' parameters			Customer segments & their utilities				
Configurations	Fixed Cost	Unit Profit	1	2	3	4	5
1	15,000	50	6	9	8	4	10
2	12,000	60	4	3	9	5	4
3	9,000	40	-2	10	7	1	9
4	7,000	55	4	5	7	8	9
5	6,000	35	-4	9	4	2	5
6	5,000	45	6	9	9	6	10
7	6,000	55	7	9	3	7	1
8	10,000	60	9	2	4	7	3
9	8,000	55	8	8	10	-1	-4
10	9,500	45	6	5	10	6	10
Customer segment sizes			6,000	8,500	9,500	7,000	9,000
Customer segment reservation utilities			4	4	4	4	4

Clearly, PLS is a Stackelberg game between the firm (leader), who decides the subset of the given

[1] Conjoint analysis asks the potential customers to make judgements about a small subset of the firm's envisaged product configurations, based on which it recovers the customers' value systems using some mathematical analysis. These value systems are, in turn, used to estimate the customers' utility for any product configuration that can be composed from the basic attribute levels used to represent their value systems (Green and Krieger, 1985; Dolan, 2001).

[2] A customer's reservation utility may represent her utility from some other competing product/brand that she will buy in the absence of any product configuration offered by this firm.

[3] Here, we assume that each customer will buy only one unit of the product configuration in the planning horizon for which the firm is developing its PLS. This is not an unrealistic assumption for consumer durable goods like refrigerators.

product configurations to develop to maximize its profit, and the customers (followers), who decide which among the developed configurations to purchase to maximize their utility. As such, PLS can be mathematically stated as a bilevel optimization problem. To state it formally, the following notation is first introduced.

Indices and Parameters:

P	:	Set of product configurations
p, q	:	Indices for product configurations; $p, q \in P$
S	:	Set of customer segments
s	:	Index for customer segment; $s \in S$
f_p	:	Fixed cost of developing product configuration p
π_p	:	Unit profit from product configuration p
k_s	:	Size of market segment s
u_{sp}	:	Utility of customer segment s from product configuration p
u_{s0}	:	Threshold utility of customer segment s

Decision Variables:

x_p	:	1 if product configuration p is developed, 0 otherwise
y_{sp}	:	1 if customer segment s buys product configuration p, 0 otherwise

Using the above notation, the bilevel optimization model of PLS can be stated as follows:

$$\max_{x} \sum_{s \in S} k_s \sum_{p \in P} \pi_p y_{sp} - \sum_{p \in P} f_p y_p \tag{30}$$

$$s.t.\ x_p \in \{0,1\} \qquad \forall p \in P \tag{31}$$

$$\max_{y} \sum_{s \in S} \sum_{p \in P} u_{sp} y_{sp} \tag{32}$$

$$s.t.\ y_{sp} \leq x_p \qquad \forall s \in S, p \in P \tag{33}$$

$$\sum_{p} y_{sp} \leq 1 \qquad \forall s \in S \tag{34}$$

$$y_{sp} \leq 0 \qquad \forall s \in S, p \in P : u_{sp} < u_{s0} \tag{35}$$

$$y_{sp} \in \{0,1\} \qquad \forall s \in S, p \in P \tag{36}$$

where (30) represents the firm's (leader's) objective of profit maximization, which clearly depends on the customers' buying decision (given by the y_{sp} variables), besides its own product development decision (given by the x_p variables). (32)-(36) is the inner optimization problem, which represents the

combined utility maximization problem for all the customer segments. (32) represents the customers' utility maximization objective. (33) ensures that customers can buy only the product configurations that are developed by the firm, while (34) ensures that each customer segment buys at most one configuration. (35) ensures that a customer does not buy a product configuration if the utility derived from it is less than her reservation utility. In other words, she considers only those configurations that give her a utility at least as large as her reservation utility. Note that (35) should ideally be an equality constraint, but may also be written as an inequality constraint.

The customers' problem exhibits integrality property[4], due to which the binary constraints in (36) can be replaced by its continuous relaxation ($0 \leq y_{sp} \leq 1$). This transforms the customers' problem into a pure linear program, which allows us to reformulate (30)-(36) into a single level program using the KKT conditions, as described in Section 2.3.2, or using the duality-based approach, as described in Section 2.3.4. In the following, we demonstrate the application of the duality-based approach to reduce (30)-(36) to a single level. For this, first note that the upper bounds on y_{sp} are redundant in presence of (34). Hence, (36) can be simply replaced by the non-negative constraints of the form ($y_{sp} \geq 0$). Associating the dual variables α_{sp}, β_s and γ_{sp} with the constraint sets (33), (34), and (35), respectively, the single level reformulation of (30)-(36) can be written as:

$$\max_{x} \sum_{s \in S} k_s \sum_{p \in P} \pi_p y_{sp} - \sum_{p \in P} f_p y_p \tag{37}$$

$$s.t.\ (31), (33) - (35)$$

$$y_{sp} \geq 0 \qquad \forall s \in S, p \in P \tag{38}$$

$$\alpha_{sp} + \beta_s + \gamma_{sp} \geq u_{sp} \qquad \forall s \in S, p \in P : u_{sp} < u_{s0} \tag{39}$$

$$\alpha_{sp} + \beta_s \geq u_{sp} \qquad \forall s \in S, p \in P : u_{sp} \geq u_{s0} \tag{40}$$

$$\alpha_{sp} \geq 0 \qquad \forall s \in S, p \in P \tag{41}$$

$$\beta_s \geq 0 \qquad \forall s \in S \tag{42}$$

$$\gamma_{sp} \geq 0 \qquad \forall s \in S, p \in P \tag{43}$$

[4]The integrality property can be shown using the following argument. Let us assume that the integrality property does not hold. Specifically, assume that a customer segment s buys a fraction f unit of one product configuration p and a fraction $1 - f$ of another configuration q, i.e., $y_{sp} = f$, $y_{sq} = 1 - f$ (note that constraint (34) requires $y_{sp} + y_{sq} \leq 1$ but the objective function (32) forces this constraint to be binding at optimality). If $u_{sp} > u_{sq}$, then the above solution cannot be optimal (since $1 - f$ should be 0 in the optimal solution). If, on the other hand, $u_{sp} < u_{sq}$, then the above solution cannot be optimal once again (since now f should be 0 in the optimal solution). So, the only case in which the optimal solution can be fractional is when $u_{sp} = u_{sq}$. But, in such a case, $f = 1$, $1 - f = 0$ and $f = 0$, $1 - f = 1$ are also alternate optimal solutions. Since these two are extreme point solutions, either of them will get selected as an optimal solution by any simplex-based linear program.

$$\sum_{s\in S}\sum_{p\in P} u_{sp}y_{sp} \leq \sum_{s\in S}\sum_{p\in P} \alpha_{sp}x_p + \sum_{s\in S}\beta_s \tag{44}$$

Here, (33)-(35), (38) are the constraints of the lower level primal problem, whereas (39)-(43) are the constraints of its dual. (44) is the strong duality constraint for the lower level problem, and (37), (31) represent the upper level problem. Note that (44) is nonlinear, due to the bilinear term $\alpha_{sp}x_p$, which can be replaced by the following set of linear constraints.

$$\sum_{s\in S}\sum_{p\in P} u_{sp}y_{sp} \leq \sum_{s\in S}\sum_{p\in P} z_{sp} + \sum_{s\in S}\beta_s \tag{45}$$

$$z_{sp} \leq \alpha_{sp} \qquad \forall s \in S, p \in P \tag{46}$$

$$z_{sp} \leq M_{sp}x_p \qquad \forall p \in P \tag{47}$$

$$z_{sp} \geq \alpha_{sp} - M_{sp}(1 - x_p) \qquad \forall s \in S, p \in P \tag{48}$$

$$z_{sp} \geq 0 \qquad \forall s \in S, p \in P \tag{49}$$

where M_{sp} appearing in (47) and (48) are large numbers (BigM).

Proposition 1. *The following expression*[5] *provides a valid value of* M_{sp} *appearing in* (47)-(48)*:*

$$M_{sp} \geq \begin{cases} u_{sp}, & \text{if } u_{sp} \geq u_{s0} \\ 0, & \text{if } u_{sp} < u_{s0} \end{cases} \tag{50}$$

When the lower level problem is a linear program, as is the case with PLS, using the duality-based approach replaces the follower's problem by a set of primal constraints, dual constraints and the strong duality condition. For some problems, it is possible to replace the follower's problem by constraints not necessarily obtained from the duality-based approach or the KKT conditions. For example, Ramamoorthy et al. (2017, 2018) reduce the bilevel hub interdiction[6] to a single level by replacing the follower's problem by the so-called "closest assignment constraints". This is also possible for PLS, using which it can be stated as follows (McBride and Zufryden, 1988):

$$(30), (31), (33) - (36)$$

[5]The validity of the expression can be shown as follows. From (46) and (47), it is clear that a valid M_{sp} should satisfy $M_{sp} \geq \alpha_{sp}$. Since α_{sp} is the dual variable corresponding to (33), the largest value it can take is given by the maximum change in the objective function (32) when x_p in the RHS of (33) changes from 0 to 1. Clearly, the maximum change in (32) when x_p changes from 0 to 1 is as given by (50).

[6]An example of an interdiction problem is discussed in Section 4.

$$u_{sp} \geq u_{sq}x_q - M_{spq}(1 - y_{sp}) \qquad \forall s \in S, p \in P, q \in P : p \neq q \tag{51}$$

where (51) models the customers' first choice rule, and M_{spq} in used in the constraint is a large number (BigM). The above single level integer program can be solved using any standard off-the-shelf integer program solver. The tightness of its linear program (LP) relaxation, and hence its computational efficiency, depends on the choice of M_{spq}: the lower the value of M_{spq}, the better it is.

Proposition 2. *A valid value of M_{spq} appearing in* (51) *is given by the expression*[7]*: $M_{spq} \geq |u_{sq}| + |u_{sp}|$.*

Below, we provide alternate ways to model the customers' first choice rule", where one of the following constraint sets that do not require BigM can be used instead of (51).

$$y_{sp} \geq y_{sq} - (1 - x_p) \qquad \forall s \in S, p \in P, q \in P : u_{sp} > u_{sq} \tag{44.1}$$

$$\sum_{q:u_{sq}<u_{sp}} y_{sq} \leq 1 - x_p \qquad \forall s \in S, p \in P \tag{44.2}$$

$$\sum_{q:u_{sq}\geq u_{sp}} y_{sq} \geq x_p \qquad \forall s \in S, p \in P \tag{44.3}$$

$$\sum_{q} u_{sq}y_{sq} \geq u_{sp}x_p \qquad \forall s \in S, p \in P \tag{44.4}$$

(44.1) and (44.2) ensure, although in slightly different ways, that if product configuration p is developed by the firm, then any customer segment cannot buy any product configuration that gives it strictly less utility than configuration p. (44.3) and (44.4) ensure that if product configuration p is developed by the firm, then any customer segment should only buy a configuration with utility at least as high as it gets from configuration p.

Table 2: Optimal product line selection

	Customer segments & their configuration choices				
Configurations developed	1	2	3	4	5
2	0	0	1	0	1
7	0	1	0	0	0
8	1	0	0	1	0

The optimal solution to the problem is summarized in Table 2 and its AMPL code is provided in the appendix. It is worth noting that in the optimal solution, the customer segment 4 chooses product

[7]The validity of the expression can be shown as follows. For $y_{sp} = 1$, (51) is always valid irrespective of the value of M_{spq}. For $y_{sp} = 0$, (51) reduces to $u_{sp} \geq u_{sq}x_q - M_{spq}$, which is valid for any $M_{spq} \geq |u_{sp}| + |u_{sp}|$.

configuration 8, and not 7, even though both provide it the same utility (see Table 1). Further, note that product configuration 8 is more profitable to the firm than configuration 7: it gives the firm a profit of 60 per unit, compared to only 55 for configuration 7. Hence, the above optimal solution represents an optimistic solution to the problem, and all the alternate formulations of the problem discussed above are optimistic formulations, as discussed in Section 2. On the other hand, if the firm is pessimistic that the customer segment 4 may select product configuration 7, instead of 8, then the above formulations need to be modified to take into account the firm's pessimistic position.

4. Shortest Path Interdiction

The shortest path interdiction problem entails destroying (interdicting) a subset of arcs, using a given set of resources/budget, in a given network so as to make the shortest path between a source-destination pair as long as possible. The problem belongs to the general class of network interdiction problems, which has military applications (see references in Israeli and Wood, 2002). Let us understand the problem using a simple 5-node directed network given in Figure 1, in which node s is the source and node t is the destination. Each arc is labelled with a number, and the triplet against it represents its normal length (or time), the delay on it caused by its interdiction, and the amount of resource required to interdict it, respectively. Assume the interdictor has a limited budget sufficient to interdict a maximum of two arcs. With this budget, the possible subset of arcs that can interdicted and the resulting lengths of the different paths between the origin-destination pair are given in Table 3. Clearly, there are five alternate shortest paths, each with a length of 16, that are the longest possible. The five shortest paths and the corresponding subsets of arcs to interdict are summarized in Table 4.

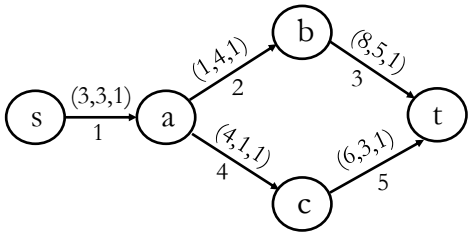


Figure 1: A 5-node illustrative network

The problem in the above illustrative example was small enough to be solved by complete enumeration. However, for a network of a reasonable size, the number of solutions will be too huge to enumerate, thus requiring a more practical approach to solve the problem. Therefore, we next present a mathematical model of the problem, which turns out to be a bilevel program. Subsequently, we who how to reduce the bilevel program to a single level, which can be solved using an off-the-shelf

Table 3: Alternate O-D paths in the 5-node illustrative network

Interdicted	Possible Paths		Shortest Path	
Arcs	Path	Path Length	Path	Path Length
{1,2}	1-2-3	(3+3)+(1+4)+(8+0) = 19	1-4-5	16
	1-4-5	(3+3)+(4+0)+(6+0) = 16		
{1,3}	1-2-3	(3+3)+(1+0)+(8+5) = 20	1-4-5	16
	1-4-5	(3+3)+(4+0)+(6+0) =16		
{1,4}	1-2-3	(3+3)+(1+0)+(8+0) = 15	1-2-3	15
	1-4-5	(3+3)+(4+1)+(6+0) = 17		
{1,5}	1-2-3	(3+3)+(1+0)+(8+0) = 15	1-2-3	15
	1-4-5	(3+3)+(4+0)+(6+3) = 19		
{2,3}	1-2-3	(3+0)+(1+4)+(8+5) =21	1-4-5	13
	1-4-5	(3+0)+(4+0)+(6+0) = 13		
{2,4}	1-2-3	(3+0)+(1+4)+(8+0) = 16	1-4-5	14
	1-4-5	(3+0)+(4+1)+(6+0) = 14		
{2,5}	1-2-3	(3+0)+(1+4)+(8+0) = 16	1-2-3	16
	1-4-5	(3+0)+(4+0)+(6+3) = 16	1-4-5	16
{3,4}	1-2-3	(3+0)+(1+0)+(8+5) = 17	1-4-5	14
	1-4-5	(3+0)+(4+1)+(6+0) = 14		
{3,5}	1-2-3	(3+0)+(1+0)+(8+5) = 17	1-4-5	16
	1-4-5	(3+0)+(4+0)+(6+3) = 16		
{4,5}	1-2-3	(3+0)+(1+0)+(8+0) = 12	1-2-3	12
	1-4-5	(3+0)+(4+1)+(6+3) = 17		

Table 4: Alternate optimal solutions of the 5-node illustrative network

Optimal Solution No.	Arcs Interdicted (Leader's Strategy)	Shortest Path (Follower's Best Response)
1	{1,2}	1-4-5
2	{1,3}	1-4-5
3	{2,5}	1-2-3
		1-4-5
4	{3,5}	1-4-5

mixed-integer programming solver. For the mathematical model, consider the following notation:

Indices and Parameters:

N : Set of nodes in the network

i, j : Indices for nodes; $i, j \in N$

A : Set of arcs in the network

k : Index for arcs; $k \in A$

$I(i)$: Subset of arcs directed into node i

$O(i)$: Subset of arcs directed out of node i

c_k : Normal length of arc k

d_k : Delay on arc k caused by its interdiction

r_k : Resource required to interdict arc k

B : Budget (interdiction resource) available

Decision Variables:

x_k : 1 if arc k is interdicted, 0 otherwise

y_k : 1 if arc k lies on a shortest path

Using the above notation, the mathematical model for the shortest path interdiction problem can be stated as:

$$\max_{x} Z \tag{52}$$

$$s.t. \sum_{k \in A} r_k x_k \leq B \tag{53}$$

$$x_k \in \{0, 1\} \qquad \forall k \in A \tag{54}$$

$$Z = \min_{y} \sum_{k \in A} (c_k + d_k x_k) y_k \tag{55}$$

$$s.t. \sum_{k \in O(i)} y_k - \sum_{k \in I(i)} y_k = \begin{cases} 1, \text{ if } i = s \\ -1, \text{ if } i = t \\ 0, \text{ if } i \in N \setminus \{s, t\} \end{cases} \qquad \forall i \in N \qquad (56)$$

$$y_k \in \{0, 1\} \qquad \forall k \in A \qquad (57)$$

(52)-(57) is a bilevel integer program. (55)-(57) represents the inner (follower's) optimization problem. The follower's problem (55)-(57) is a shortest path problem, in which the objective function (55) captures the normal length (or time) of arc k, and the added delay if arc k is interdicted. Constraint set (56) represents the flow balance constraints typically used in a shortest-path problem. The leader's objective in (52) is to maximize the length of the follower's shortest path, using his limited set of resources given by the budget constraint (53).

Note that the follower's problem (55)-(57), which is a shortest path problem, is known to exhibit integrality property (Bazaraa et al., 2008). This means that the binary constraints in (57) can be replaced by their continuous relaxations ($0 \leq y_k \leq 1$). This makes the follower's optimization problem a continuous linear program, which can be replaced by its dual using the strong law of duality. Further, the upper bounds on y_k are redundant due to (56). Hence, (57) can be simply replaced by the non-negativity constraints of the form ($y_k \geq 0$). Let us illustrate this for the 5 node, 5 arc network in Figure 1. For $x_k = 0 \, \forall k \in A$, the follower's shortest path problem can be written as:

$$\begin{aligned} Z = \min \; & 3y_1 + y_2 + 8y_3 + 4y_4 + 6y_5 && \text{(Dual Variable)} \\ \text{s.t. } & y_1 = 1 && (\pi_s) \\ & -y_1 + y_2 + y_4 = 0 && (\pi_a) \\ & -y_2 + y_3 = 0 && (\pi_b) \\ & -y_4 + y_5 = 0 && (\pi_c) \\ & -y_3 - y_5 = -1 && (\pi_t) \\ & y_k \geq 0 && \forall k \in \{1, 2, 3, 4, 5\} \end{aligned}$$

The dual of the above model can be written as:

$$\begin{aligned} Z = \max \; & \pi_s - \pi_t \\ \text{s.t. } & \pi_s - \pi_a \leq 3 \end{aligned}$$

$$\pi_a - \pi_b \leq 1$$

$$\pi_b - \pi_t \leq 8$$

$$\pi_a - \pi_c \leq 4$$

$$\pi_c - \pi_t \leq 6$$

$$\pi_i \text{ free} \qquad \forall i \in \{s, a, b, c, t\}$$

The above dual problem has multiple optimal solutions: for instance, consider $Z = 5$, then ($\pi_s = 0$, $\pi_t = 5$) and ($\pi_s = 1$, $\pi_5 = 6$) represent two possible optimal solutions. In general, if $\{\pi_t^*\}_{i \in N}$ is an optimal solution, then so is $\{\pi_t^* + n\}_{i \in N}$, $\forall n > 0$. Therefore, we can arbitrarily set $\pi_s = 0$. In that case, for any node i that lies on the shortest path, the value of π_i in the optimal solution gives the negative of the length of the shortest path from the origin s to node i. If we, however, replace π_i by -π_i' in the above dual, then π_i' in the optimal solution has a much nicer interpretation: it represents the length of the shortest path from the origin s to node i if node i lies on the shortest path. Using the variable π_i', the dual of the follower's problem (55)-(57) can, in general, be written as:

$$\max_{\pi_i'} \pi_t' \tag{58}$$

$$s.t.\ \pi_j' - \pi_i' \leq c_k + d_k x_k \qquad \forall k = (i, j) \in A \tag{59}$$

$$\pi_s' = 0 \tag{60}$$

$$\pi_i' \text{ free} \qquad \forall i \in N \tag{61}$$

Using strong duality, the follower's problem (55)-(57) can be replaced by the above dual problem. This simple trick allows us to reduce the bilevel program (52)-(57) to the following single level mixed-integer program.

$$\max_{x, \pi_i'} \pi_t' \tag{62}$$

$$s.t.\ (53), (54), (59) - (61)$$

Solving the above single level reformulation of the shortest path interdiction problem for the 5-node network in Figure 1 results in the following optimal solution: $x_3 = x_5 = 1$, $y_1 = y_4 = y_5 = 1$ (the values of the y variables are recovered using the primal-dual relationship between the y and π variables). This, in the parlance of game theory, implies that if the leader (interdictor) is a rational person and knows that the follower is also rational, then the interdictor will interdict arcs 3 and 5. The

follower, if rational, will choose 1-4-5 as the shortest path in the interdicted network. This solution is an equilibrium solution under a hierarchical setting, commonly referred to as the Stackelberg equilibrium.

The resulting single level program for the 5-node illustrative example can be solved by any standard off-the-shelf mixed-integer programming solver. Its AMPL code and the output are provided in the appendix. However, the model becomes difficult to solve for practical-size problems, and the interested readers can refer to Israeli and Wood (2002) for a discussion on some advanced methods (Benders decomposition and Supvervalid inequalities) to solve it more efficiently. Interdiction problems have also been studied in the context of multicommodity network flows (Lim and Smith, 2007), facility locations (Nigudkar et al., 2021), and hub locations (Ramamoorthy et al., 2017, 2018; Bhatt et al., 2021). A natural extension of these interdiction problems arises when one of the agents (leader or follower) has imperfect information related to some aspect of the problem. Such an extension in which the follower has imperfect information about the arc lengths, in the context of shortest interdiction, is studied by Bayrak and Bailey (2008).

5. Conclusions

This chapter provided a discussion on the basics of bilevel optimization and the various ways to reformulate a bilevel optimization problem into a single level optimization problem. It considered two classes of bilevel problems, namely, the product line selection problem and the shortest path interdiction problem, and showed how to solve them exactly. In the context of the product line selection problem, it also showed how the bilevel structure can be by-passed in special cases using constraints similar to the so-called "closest assignment constraints". Using such constraints, it provided some new single level reformulations for the product line selection problem not reported earlier in the literature. For the ease of implementation of the discussed ideas in the context of the larger instances of the product line selection and the shortest path interdiction problems, and also other application problems, it provided the AMPL codes in the appendix.

6. Acknowledgements

The authors acknowledge the support provided by the Research & Publications Cell and Brij Disa Centre for Data Science and Artificial Intelligence, Indian Institute of Management Ahmedabad.

References

Aiyoshi, E. and Shimizu, K. (1981), 'Hierarchical decentralized systems and its new solution by a barrier method', *IEEE Transactions on Systems, Man, and Cybernetics* **6**, 444–449.

Aiyoshi, E. and Shimizu, K. (1984), 'A solution method for the static constrained Stackelberg problem via penalty method', *IEEE Transactions on Automatic Control* **29**, 1111–1114.

Al-Khayyal, F., Horst, R. and Pardalos, P. (1992), 'Global optimization of concave functions subject to quadratic constraints: an application in nonlinear bilevel programming', *Annals of Operations Research* **34**, 125–147.

Angelo, J., Krempser, E. and Barbosa, H. (2013), Differential evolution for bilevel programming, *in* 'Proceedings of the 2013 Congress on Evolutionary Computation (CEC-2013)', IEEE Press.

Angelo, J. S., Krempser, E. and Barbosa, H. J. C. (2014), Differential evolution assisted by a surrogate model for bilevel programming problems, *in* 'Evolutionary Computation (CEC), 2014 IEEE Congress on', IEEE, pp. 1784–1791.

Bard, J. (1998), *Practical Bilevel Optimization: Algorithms and Applications*, The Netherlands: Kluwer.

Bard, J. and Falk, J. (1982), 'An explicit solution to the multi-level programming problem', *Computers and Operations Research* **9**, 77–100.

Bard, J. and Moore, J. (1990), 'A branch and bound algorithm for the bilevel programming problem', *SIAM Journal on Scientific and Statistical Computing* **11**, 281–292.

Bayrak, H. and Bailey, M. D. (2008), 'Shortest path network interdiction with asymmetric information', *Networks: An International Journal* **52**(3), 133–140.

Bazaraa, M. S., Jarvis, J. J. and Sherali, H. D. (2008), *Linear programming and network flows*, John Wiley & Sons.

Beale, E. M. L. and Tomlin, J. A. (1970), 'Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables', *OR* **69**(447-454), 99.

Ben-Ayed, O. (1993), 'Bilevel linear programming', *Computers and Operations Research* **20**, 485–501.

Bennett, K. P., Kunapuli, G., Hu, J. and Pang, J.-S. (2008), Bilevel optimization and machine learning, *in* 'Computational Intelligence: Research Frontiers', Springer, pp. 25–47.

Bhatt, S., Sinha, A. and Jayaswal, S. (2021), 'Hub interdiction problem under congestion: formulations and solution methods', *Working Paper* .

Bostian, M., Whittaker, G., Sinha, A. and Barnhart, B. (2015), Incorporating data envelopment analysis solution methods into bilevel multi-objective optimization, *in* '2015 IEEE Congress on Evolutionary Computation (CEC)', IEEE, pp. 1667–1674.

Brotcorne, L., Labbe, M., Marcotte, P. and Savard, G. (2001), 'A bilevel model for toll optimization on a multicommodity transportation network', *Transportation Science* **35**(4), 345–358.

Brown, G., Carlyle, M., Diehl, D., Kline, J. and Wood, K. (2005), 'A Two-Sided Optimization for Theater Ballistic Missile Defense', *Operations Research* **53**(5), 745–763.

Christiansen, S., Patriksson, M. and Wynter, L. (2001), 'Stochastic bilevel programming in structural optimization', *Structural and multidisciplinary optimization* **21**(5), 361–371.

Colson, B., Marcotte, P. and Savard, G. (2005), 'A trust-region method for nonlinear bilevel programming: algorithm and computational experience', *Computational Optimization and Applications* **30**(3), 211–227.

Dempe, S. (2002), *Foundations of Bilevel Programming*, Kluwer Academic Publishers, Secaucus, NJ, USA.

Dolan, R. J. (2001), *Analyzing consumer preferences*, Harvard Business School Publications.

Edmunds, T. and Bard, J. (1991), 'Algorithms for nonlinear bilevel mathematical programming', *IEEE Transactions on Systems, Man, and Cybernetics* **21**, 83–89.

Fortuny-Amat, J. and McCarl, B. (1981), 'A representation and economic interpretation of a two-level programming problem', *Journal of the Operational Research Society* **32**, 783–792.

Green, P. E. and Krieger, A. M. (1985), 'Models and heuristics for product line selection', *Marketing Science* **4**(1), 1–19.

Hu, X. and Ralph, D. (2007), 'Using EPECs to Model Bilevel Games in Restructured Electricity Markets with Locational Prices', *Operations Research* **55**(5), 809–827.

Ishizuka, Y. and Aiyoshi, E. (1992), 'Double penalty method for bilevel optimization problems', *Annals of Operations Research* **34**, 73–88.

Islam, M. M., Singh, H. K. and Ray, T. (2017), 'A surrogate assisted approach for single-objective bilevel optimization', *IEEE Transactions on Evolutionary Computation* **21**(5), 681–696.

Israeli, E. and Wood, R. K. (2002), 'Shortest-path network interdiction', *Networks: An International Journal* **40**(2), 97–111.

Küçükaydin, H., Aras, N. and Altınel, I. K. (2011), 'Competitive facility location problem with attractiveness adjustment of the follower: A bilevel programming model and its solution', *European Journal of Operational Research* **208**(3), 206–220.

Labbé, M., Marcotte, P. and Savard, G. (1998), 'A Bilevel Model of Taxation and Its Application to Optimal Highway Pricing', *Management Science* **44**(12), 1608–1622.

Li, X., Tian, P. and Min, X. (2006), A hierarchical particle swarm optimization for solving bilevel programming problems, *in* L. Rutkowski, R. Tadeusiewicz, L. A. Zadeh and J. M. Zurada, eds, 'Artificial Intelligence and Soft Computing - ICAISC 2006', Vol. 4029 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 1169–1178.

Lim, C. and Smith, J. C. (2007), 'Algorithms for discrete and continuous multicommodity flow network interdiction problems', *IIE Transactions* **39**(1), 15–26.

Liu, G., Han, J. and Wang, S. (1998), 'A trust region algorithm for bilevel programing problems', *Chinese science bulletin* **43**(10), 820–824.

Marcotte, P., Savard, G. and Zhu, D. L. (2001), 'A trust region algorithm for nonlinear bilevel programming', *Operations research letters* **29**(4), 171–179.

McBride, R. D. and Zufryden, F. S. (1988), 'An integer programming approach to the optimal product line selection problem', *Marketing Science* **7**(2), 126–140.

Migdalas, A. (1995), 'Bilevel programming in traffic planning: Models, methods and challenge', *Journal of Global Optimization* **7**(4), 381–405.

Nigudkar, S., Jayaswal, S., Sinha, A. and Nair, A. (2021), 'Design for failure of food banking networks', *Working Paper* .

Ramamoorthy, P., Jayaswal, S., Sinha, A. and Vidyarthi, N. (2017), Hub-and-spoke network design under the risk of interdiction. (No. WP 2017-05-01), Technical report, Indian Institute of Management Ahmedabad.

Ramamoorthy, P., Jayaswal, S., Sinha, A. and Vidyarthi, N. (2018), 'Multiple allocation hub interdiction and protection problems: Model formulations and solution approaches', *European Journal of Operational Research* **270**(1), 230–245.

Savard, G. and Gauvin, J. (1994), 'The steepest descent direction for the nonlinear bilevel programming problem', *Operations Research Letters* **15**, 275–282.

Sinha, A., Khandait, T. and Mohanty, R. (2020), 'A gradient-based bilevel optimization approach for tuning hyperparameters in machine learning', *CoRR* **abs/2007.11022**.

Sinha, A., Lu, Z., Deb, K. and Malo, P. (2020), 'Bilevel optimization based on iterative approximation of multiple mappings', *Journal of Heuristics* **26**(2), 151–185.

Sinha, A., Malo, P. and Deb, K. (2015), Transportation policy formulation as a multi-objective bilevel optimization problem, *in* '2015 IEEE Congress on Evolutionary Computation (CEC-2015)', IEEE Press.

Sinha, A., Malo, P. and Deb, K. (2017), 'Evolutionary algorithm for bilevel optimization using approximations of the lower level optimal solution mapping', *European Journal of Operational Research* **257**(2), 395–411.

Sinha, A., Malo, P. and Deb, K. (2018), 'A review on bilevel optimization: From classical to evolutionary approaches and applications', *IEEE Transactions on Evolutionary Computation* **22**(2), 276–295.

Sinha, A., Malo, P., Frantsev, A. and Deb, K. (2013), Multi-objective stackelberg game between a regulating authority and a mining company: A case study in environmental economics, *in* '2013 IEEE Congress on Evolutionary Computation (CEC-2013)', IEEE Press.

Sinha, A., Malo, P., Frantsev, A. and Deb, K. (2014), 'Finding optimal strategies in a multi-period multi-leader-follower stackelberg game using an evolutionary algorithm', *Computers & Operations Research* **41**, 374–385.

Sinha, A., Malo, P., Xu, P. and Deb, K. (2014), A bilevel optimization approach to automated parameter tuning, *in* 'Proceedings of the 16th Annual Genetic and Evolutionary Computation Conference (GECCO 2014)', New York: ACM Press.

Sinha, A. and Shaikh, V. (2021), 'Solving bilevel optimization problems using kriging approximations', *IEEE Transactions on Cybernetics* .

Sobieszczanski-Sobieski, J., Agte, J. S. and Sandusky Jr, R. R. (2000), 'Bilevel integrated system synthesis', *AIAA journal* **38**(1), 164–172.

Stackelberg, H. (1952), *The theory of the market economy*, Oxford University Press, New York, Oxford.

Sun, H., Gao, Z. and Wu, J. (2008), 'A bi-level programming model and solution algorithm for the location of logistics distribution centers', *Applied Mathematical Modelling* **32**(4), 610–616.

Tuy, H., Migdalas, A. and Värbrand, P. (1993), 'A global optimization approach for the linear two-level program', *Journal of Global Optimization* **3**, 1–23.

Vicente, L., Savard, G. and Júdice, J. (1994), 'Descent approaches for quadratic bilevel programming', *Journal of Optimization Theory and Applications* **81**, 379–399.

Wein, L. (2009), 'Homeland Security: From Mathematical Models to Policy Implementation: The 2008 Philip McCord Morse Lecture', *Operations Research* **57**(4), 801–811.

Wen, U. and Hsu, S. (1991), 'Linear bi-level programming problems - a review', *Journal of the Operational Research Society* **42**, 125–133.

White, D. and Anandalingam, G. (1993), 'A penalty function approach for solving bi-level linear programs', *Journal of Global Optimization* **3**, 397–419.

Whittaker, G., Färe, R., Grosskopf, S., Barnhart, B., Bostian, M., Mueller-Warrant, G. and Griffith, S. (2017), 'Spatial targeting of agri-environmental policy using bilevel evolutionary optimization', *Omega* **66**, 15–27.

Ye, J. J. and Zhu, D. (2010), 'New necessary optimality conditions for bilevel programs by combining the mpec and value function approaches', *SIAM Journal on Optimization* **20**(4), 1885–1905.

Appendix A. AMPL Codes for Product Line Selection

```
#----------DATA File-------------
#Name of Data file: PLS_10_5.dat

param N_Product:= 10;
param N_Segment:= 5;

param:
dev_cost profit:=
1 15000 50
2 12000 60
```

```
3 9000 40
4 7000 55
5 6000  35
6 5000 45
7 6000 55
8 10000 60
9 8000 55
10 9500 45;

param:
seg_size reserv_util:=
1 6000 4
2 8500 4
3 9500 4
4 7000  4
5 9000 4;

param utility(tr):
1 2 3 4 5:=
1 6 9 8 4 10
2 4 3 9 5 4
3 -2 10 7 1 9
4 4 5 7 8 9
5 -4 9 4 2 5
6 6 9 9 6 10
7 7 9 3 7 1
8 9 2 4 7 3
9 8 8 10 -1 -4
10 6 5 10 6 10;
#-----------END OF DATA File---------

#---------MODEL FILE----------------
#Name of Model file: model PLS.mod (Single Level)
param N_Product;
param N_Segment;
set Product:= 1..N_Product;
set Segment:= 1..N_Segment;

param utility {Segment, Product}; #Utility of a product to a given customer segment
param seg_size {Segment} >=0; #Size of each customer segment
param reserv_util{Segment} >=0;  #UReservation tility of a given customer segment
param dev_cost {Product} >=0; #Product Development Cost
param profit {Product} >=0; #Unit profit on each product
param M{s in Segment, p in product, q in product: q!=p}:= abs(utility[s, q])+ abs(utility[s, p]);#Large number

var Develop {Product} binary; #1 if the particular product is selected, 0 otherwise
```

```
var Buy {Segment, Product} binary; #1 if the product is chosen by the customer segment, 0 otherwise

maximize
Profit: sum {s in Segment, p in Product}profit[p]*seg_size[s]*Buy[s, p] - sum {p in Product}dev_cost[p]*Develop[p];
subject to
Buy_only_if_developed{s in Segment, p in Product}: Buy[s, p] <= Develop[p];
Buy_max_one_product{s in Segment}: sum {p in Product}Buy[s, p] <= 1;
Dont_buy_if_below_reserve_util{s in Segment, p in Product: utility[s, p] < reserv_util[s]}: Buy[s, p] <= 0;
Max_Util_Choice{s in Segment, p in Product, q in Product: q != p}: utility[s, p] >= utility[s, q]*Develop[q] - M[s,p,q]*(1-Buy[s,
#---------END OF MODEL FILE----------------

#---------RUN FILE----------------
reset;
model PLS.mod;
data PLS_10_5.dat;
option solver cplex;
option show_stats 1;
option cplex_options 'mipdisplay=4 mipinterval=2';#Extent to which display B&B Search info

solve;

option omit_zero_rows 1;#Do not display variables that have 0 values;
option omit_zero_cols 1;#Do not display variables that have 0 values;

display _total_solve_time >Output_PLS.txt;
display Profit>Output_PLS.txt;
display Develop>Output_PLS.txt;
display Buy>Output_PLS.txt;
expand>Output_PLS.txt;
close Output_PLS.txt;
#---------END OF RUN FILE----------------

#---------OUTPUT FILE----------------
#Output_PLS.txt
_total_solve_time = 0.0

Profit = 2329500.0

Develop [*] :=
 2  1.0
 7  1.0
 8  1.0;

Buy :=
1 8   1.0
2 7   1.0
```

```
3 2   1.0
4 8   1.0
5 2   1.0;
#---------END OF OUTPUT FILE----------------
```

Appendix B. AMPL Codes for Shortest Path Interdiction

```
#----------DATA File-------------
#Name of Data file: SPI_5_5.dat
param N_Nodes:= 5;
param N_Arcs:= 5;
param Budget:= 2;#Total resource available to the Interdictor
param Source_Node:= 1;
param Dest_Node:= 5;
param Node_Arc:
1 2 3 4 5:=
1 -1 1 0 0 0
2 0 -1 1 0 0
3 0 0 -1 0 1
4 0 -1 0 1 0
5 0 0 0 -1 1; #Each Row is an Arc Number
param: Arc_Length Arc_Delay Arc_Resource:=
1 3 3 1
2 1 4 1
3 8 5 1
4 4 1 1
5 6 3 1;
#-----------END OF DATA File---------

#---------MODEL FILE----------------
#Name of Model file: model SP_Interdict.mod
param N_Nodes;
param N_Arcs;
param Budget;#Total resource available to the Interdictor
param Source_Node;
param Dest_Node;

set NODES:= 1..N_Nodes;
set ARCS:= 1..N_Arcs;

param Node_Arc{ARCS, NODES};
param Arc_Length{ARCS};
param Arc_Delay{ARCS};
param Arc_Resource{ARCS}; #amount of resource required to interdict the arc

var Interdict{ARCS} binary; #1 if Arc a is interdicted, 0 otherwise
```

```
var Dual{NODES}; #Dual variable for Node i. It represents the length of the shortest path to node i from source s

maximize Longest_shortest_path: Dual[Dest_Node] - Dual[Source_Node];
subject to
Dual_Constraint{a in ARCS}: sum{n in NODES} Node_Arc[a,n]*Dual[n] -
Arc_Delay[a]*Interdict[a] <= Arc_Length[a];
Dual_node_source: Dual[Source_Node] = 0;
Interdiction_budget: sum{a in ARCS} Arc_Resource[a]*Interdict[a] <= Budget;
#---------END OF MODEL FILE----------------

#---------RUN FILE---------------
reset;
model SP_Interdict.mod;
data SP_5_5.dat;
option solver cplex;
option show_stats 1;
option cplex_options 'mipdisplay=4 mipinterval=2';#Extent to which display B&B Search info

solve;

option omit_zero_rows 1;# Do not display variables that have 0 values
option omit_zero_cols 1;# Do not display variables that have 0 values

display _total_solve_time>SP_Interdict.txt;#Display the total CPU time
display Longest_shortest_path>SP_Interdict.txt;
display Interdict>SP_Interdict.txt;
display Dual>SP_Interdict.txt;
display _conname, _con.dual>SP_Interdict.txt;#Dual variable values (Arcs selected)
close >SP_Interdict.txt;
#---------END OF RUN FILE----------------

#---------OUTPUT FILE---------------
#SP_Interdict.txt
_total_solve_time = 0.1
Longest_shortest_path = 16.0
Interdict [*] :=
3  1.0
5  1.0;

Dual [*] :=
2   3.0
3   3.0
4   7.0
5  16.0;

:         _conname       _con.dual    :=
```

```
1   'Dual_Constraint[1]'     1.0
2   'Dual_Constraint[2]'     0.0
3   'Dual_Constraint[3]'     0.0
4   'Dual_Constraint[4]'     1.0
5   'Dual_Constraint[5]'     1.0
6   Dual_node_source         0.0
7   Interdiction_budget      3.0;
#---------END OF OUTPUT FILE----------------
```