Designing of a Mat-heuristic Algorithm for Solving Bi-level Optimization Problems

Hossein Shams Shemirani
Ph. D Candidate of Industrial Engineering
Department of Industrial Engineering
Faculty of Engineering, Shahed University, Tehran, Iran
shams@gut.ac.ir
+989125131558

*Rashed Sahraeian
Professor of Industrial Engineering
Department of Industrial Engineering
Faculty of Engineering, Shahed University, Tehran, Iran
Corresponding author: Sahraeian@shahed.ac.ir
+989122145489

Mahdi Bashiri
Professor of Industrial Engineering
Department of Industrial Engineering,
Faculty of Engineering, Shahed University, Tehran, Iran
Bashiri@shahed.ac.ir
+989123150355
Designing of a Mat-heuristic Algorithm for Solving Bi-level Optimization Problems

Abstract:

In this paper, a new algorithm for solving bi-level optimization problems is presented. This algorithm can obtain the optimal or near-optimal solution for any bi-level optimization problem. The decision variables of the first and second level models can be both integers and continuous. In this method, by solving a certain number of the bi-objective programming model and then solving the corresponding second-level model, a bi-level feasible solution that is either optimal or near-optimal is identified. To evaluate the efficiency of the algorithm, the value of the objective function, as well as its computation time in different instances, are compared with exact methods as well as evolution-based methods. The numerical results confirm the high efficiency of the proposed algorithm.

Keywords:

Bi-level optimization, Mat-heuristic algorithm, Meta-heuristic methods, Evolution-based methods, Bi-level programming.
1. Introduction

The bi-level programming is a problem in which there are two decision-makers. The first
decision-maker is called a leader and the second decision-maker is called a follower. Some of
the variables are identified by the leader and then the others are identified by the follower. The
objective function and the leader's constraints may also include the decision variables of the
follower and vice versa. In the bi-level programming process, the leader first determines the
values of its decision variables. Then the follower determines the values of its decision
variables by solving a mathematical model considering the leader's decisions. The process of
implementing the bi-level optimization problems is as follows.

Step 1: The leader considers values for first-level decision variables.

Step 2: The follower receives the values of the leader's variables and inserts them as
parameters into the second level model.

Step 3: The follower solves the second level mathematical model and determines the optimal
values for the second level decision variables.

Step 4: The leader receives the values of the follower's variables.

Step 5: The leader calculates the value of the first level objective function.

Step 6: The leader will check whether the first level constraints are in place?

Step 7: The leader wants to determine the values of the first level decision variables in such a
way that, due to the follower's reaction, the first level constraints are satisfied and the
first level objective function is optimized.

The bi-level optimization problems are expressed in general terms using mathematical and
vector symbols in the following form:

\[ \max_{X} f_{i}(X,Y) \]  \hspace{1cm} (1)

\[ \text{Subject to} \]

\[ g_{i}(X,Y) \leq 0 \]  \hspace{1cm} (2)
The following definitions are used in the above equations:

\( X \): Vector of decision variables for the leader

\( Y \): Vector of decision variables for the follower

\( f_1(X,Y) \): Leader objective function

\( g_1(X,Y) \leq 0 \): Leader constraints set

\( f_2(X,Y) \): Follower objective function

\( g_2(X,Y) \leq 0 \): Follower constraints set

Bi-level programming problems, in the simplest form, those mathematical models for the leader and the follower are linear and their decision variables are continuous, can be converted into a MIP problem. In this case, by using the Karush–Kuhn–Tucker (KKT) conditions, we can transform the follower model into a series of constraints. Then, by placing them into the constraints of the leader, a nonlinear model is created. This non-linearity is due to the constraints of the complementary slackness. By using binary variables, these terms can be linearized to create a (Mixed Integer Programming) MIP model. But this model can only be solved in small sizes, and as the number of variables and constraints of followers increases, the number of binary variables will increase, too. Computational experiences have indicated plenty of computational time will be needed.

In the state that the possible values for the leader’s decision variables are discrete and the number of possible states for them is not high, we can identify the optimal solution using the complete enumeration method. If the number of possible solutions is high for the leader, then the computational time of this method will be very long and time-consuming. How to utilize
the complete enumeration method and KKT and MIP method is described in the section of the general formulation.

Because there has not been an exact method that can identify the optimal solution for a variety of bi-level optimization problems in an acceptable time, designing a method that can identify optimal or near-optimal solutions in a short time is crucial. In this paper, using fuzzy normalization, goal programming, and concepts of bi-level optimization, we introduce an algorithm that can find optimal or near-optimal solutions in a variety of bi-level optimization problems. Problems in which models of leader and follower can be solved using methods of (Linear programming) LP, (Non-linear programming) NLP, (Mixed integer programming) MIP, and (Mixed integer non-linear programming) MINLP in a relatively short time. Because of the mechanism of this method, we named it GPBLO (Goal Programming for Bi-level Optimization).

The rest of this paper is organized as follows. In section 2, an overview of the published papers on the various applications of bi-level programming is discussed, as well as the various methods raised for solving them. In section 3, the general formulation of the bi-level optimization problems and the method of using the KKT conditions as well as the complete enumeration for the applicable cases are raised. In section 4, the proposed solution method, which is a mat-heuristic algorithm, is introduced. In section 5, computational experiments are analyzed on 20 sample problems in different sizes that can be solved using the KKT method, also 20 sample problems that cannot be solved using KKT but can be solved using the complete enumeration method. These problems are also solved using the proposed method, and the results are compared in terms of the value of the objective function and the time of calculations.

In the final section, conclusions and future research directions are provided.

2. Related literature
Many researchers have studied the application of bi-level programming to solve problems related to industries, services, transportation, military operations, and so on. The following can be mentioned:

1. Facility location problems [1-3]
2. Supply chain management [4-6]
3. Missile defense design [7]
4. Environmental pollution control policies [8]
5. Energy sources planning [9]

Solving the bi-level optimization problems, even in its simplest form, is strongly NP-hard according to known methods so far. Even the study of whether a given solution is optimal is also NP-hard [14]. For this reason, many researchers have tried to provide suitable procedures for such problems in an acceptable time using evolutionary algorithms. In [14] classical and evolutionary approaches proposed to solve bi-level programming problems by 2018 are under review and also some uses for bi-level programming problems are raised. Several studies have been done to solve linear and nonlinear bi-level optimization problems using evolutionary methods such as genetic algorithm (GA) and particle swarm optimization (PSO) [15-21]. The use of evolutionary methods for solving single-level and bi-level programming problems may offer optimal or near-optimal solutions, but using these methods to solve bi-level problems requires, unlike the single-level problems, very large computations and a very long time. Several studies have also been carried out to solve linear and nonlinear bi-level optimization problems using methods such as branch and bound, branch and cut, Benders decomposition,
and KKT conditions [22-26]. These methods are also commonly developed for specific models and cannot be used for solving a variety of bi-level problems, or they require a long time. Considering the wide range of applications of bi-level programming and also considering that an algorithm is not able to achieve an optimal or a near-optimal solution in an acceptable time, the identification of heuristic and mat-heuristic methods that can bring us closer to this request is crucial. In this paper, we present a fast and flexible mat-heuristic method that can obtain optimal or near-optimal solutions for various types of bi-level optimization problems in a relatively short time.

3. The general formulation

All bi-level optimization problems are expressed using vector symbols by Equations (1-4). If the mathematical relations in the problem are linear, the general formulation of the problem in the form of a matrix can be stated as follows:

\[
\begin{align*}
& \text{Max } f_1 = C_1X + D_1Y \\
& A_1X + B_1Y \leq G_1 \\
& X \geq 0 \\
& \text{Max } f_2 = C_2X + D_2Y \\
& A_2X + B_2Y \leq G_2 \\
& Y \geq 0
\end{align*}
\]

Also, the bi-level linear programming problems in general and in algebraic form can be demonstrated as follows:
Max: \( f_1 = \sum_{j_1} C_1(j_1) X(j_1) + \sum_{j_2} D_1(j_2) Y(j_2) \) \hspace{1cm} (11)

Subject to
\[
\sum_{j_1} A_1(i_1, j_1) X(j_1) + \sum_{j_2} B_1(i_1, j_2) Y(j_2) \leq G_1(i_1) \quad \forall i_1
\]
\hspace{1cm} (12)
\[X(j_1) \geq 0 \quad \forall j_1 \] \hspace{1cm} (13)

\[\max_r f_2 = \sum_{j_1} C_2(j_1) X(j_1) + \sum_{j_2} D_2(j_2) Y(j_2) \] \hspace{1cm} (14)

Subject to
\[
\sum_{j_1} A_2(i_2, j_1) X(j_1) + \sum_{j_2} B_2(i_2, j_2) Y(j_2) \leq G_2(i_2) \quad \forall i_2
\]
\hspace{1cm} (15)
\[Y(j_2) \geq 0 \quad \forall j_2 \] \hspace{1cm} (16)

3.1. Using the KKT

In linear bi-level programming problems, the second level mathematical model is a convex programming model. Because decision variables are continuous and all terms in the objective function, as well as constraints, are all linear. So, using the KKT conditions, it can be converted to the series of following constraints.

\[D_2(j_2) = \sum_{i_2} \left[ \lambda(i_2) B_2(i_2, j_2) \right] - \theta(j_2) \quad \forall j_2 \] \hspace{1cm} (17)

\[
\sum_{j_1} A_2(i_2, j_1) X(j_1) + \sum_{j_2} B_2(i_2, j_2) Y(j_2) + S(i_2) = G_2(i_2) \quad \forall i_2
\] \hspace{1cm} (18)

\[\lambda(i_2) S(i_2) = 0 \quad \forall i_2 \] \hspace{1cm} (19)

\[\theta(j_2) Y(j_2) = 0 \quad \forall j_2 \] \hspace{1cm} (20)

\[Y(j_2) \geq 0 \quad \forall j_2 \] \hspace{1cm} (21)

\[\lambda(i_2) \geq 0 \quad \forall i_2 \] \hspace{1cm} (22)
Replacing these constraints instead of the second level model turns the problem into a single-level model. Of course, relations (19) and (20) are non-linear and the problem is the type of NLP. However, we can linearize them using the following method by adding binary variables \( P(i_2) \) and \( Q(j_2) \) and using a large enough numerical coefficient called \( M_{big} \).

\[
\lambda(i_2) \leq M_{big} P(i_2) \quad \forall i_2 \tag{25}
\]

\[
S(i_2) \leq M_{big} [1 - P(i_2)] \quad \forall i_2 \tag{26}
\]

\[
\theta(j_2) \leq M_{big} Q(j_2) \quad \forall j_2 \tag{27}
\]

\[
Y(j_2) \leq M_{big} [1 - Q(j_2)] \quad \forall j_2 \tag{28}
\]

Therefore, any bi-level linear programming problem can be converted into a MIP problem as follows:

\[
\max f_1 = \sum_{j_1} C_{l}(j_1) X(j_1) + \sum_{j_2} D_{l}(j_2) Y(j_2) \tag{29}
\]

Subject to

\[
\sum_{j} A_{l}(i_1, j_1) X(j_1) + \sum_{j_2} B_{l}(i_1, j_2) Y(j_2) \leq G_{l}(i_1) \quad \forall i_1 \tag{30}
\]

\[
X(j_1) \geq 0 \quad \forall j_1 \tag{31}
\]

\[
D_{2}(j_2) = \sum_{i_1} [\lambda(i_2) B_{2}(i_2, j_2)] - \theta(j_2) \quad \forall j_2 \tag{32}
\]

\[
\sum_{j} A_{l}(i_2, j_1) X(j_1) + \sum_{j_2} B_{l}(i_2, j_2) Y(j_2) + S(i_2) = G_{l}(i_2) \quad \forall i_2 \tag{33}
\]

\[
\lambda(i_2) \leq M_{big} P(i_2) \quad \forall i_2 \tag{34}
\]
\[ S(i_2) \leq M_{\text{big}} \left[ 1 - P(i_2) \right] \quad \forall i_2 \quad (35) \]

\[ \theta(j_2) \leq M_{\text{big}} Q(j_2) \quad \forall j_2 \quad (36) \]

\[ Y(j_2) \leq M_{\text{big}} \left[ 1 - Q(j_2) \right] \quad \forall j_2 \quad (37) \]

\[ Y(j_2) \geq 0 \quad \forall j_2 \quad (38) \]

\[ \lambda(i_2) \geq 0 \quad \forall i_2 \quad (39) \]

\[ \theta(j_2) \geq 0 \quad \forall j_2 \quad (40) \]

\[ S(i_2) \geq 0 \quad \forall i_2 \quad (41) \]

\[ P(i_2) \in \{0,1\} \quad \forall i_2 \quad (42) \]

\[ Q(j_2) \in \{0,1\} \quad \forall j_2 \quad (43) \]

The above problem can be solved using MIP solvers such as CPLEX. Of course, computational experiments with this model indicated that if the number of binary variables in the model increases, the time required for computing will be increased very much. The number of binary variables required to linearize a resulting single-level model is equal to the number of limitations plus the number of variables in the second level model. Therefore, this method can only be used for those bi-level models that their second level model is linear programming, with less number of variables and number of constraints in the second level model. The steps to apply this procedure for solving bi-level linear programming (BLLP) are as follows.

**Step 1:** Convert the second level model to a series of constraints using the KKT conditions.

**Step 2:** Linearize nonlinear terms arising from KKT conditions.

**Step 3:** Solve the resulting single-level model using MIP solvers such as CPLEX.

### 3.2. Using the complete enumeration
If the decision variables of the first level model are discrete and the number of different states for them is not too high, we can utilize the complete enumeration (CE) method to obtain the optimal solution to the problem. In this method, for each of the possible solutions for the vector of first-level decision variables, the second level mathematical model is solved and the value of the leader's objective function is calculated. Then, among all the solutions, the solution is the optimal solution for which the value of the leader's objective function is the best. In this method, if the number of possible solutions for the leader is high, the calculation time will be very long. If the first level model contains only binary variables and the number of these variables is $n$, then the number of possible solutions is $2^n$. If there are only 30 binary variables, more than 1 milliard possible solutions should be considered. For each of these solutions, the second level model should be solved, which will be very time-consuming. Table 1 demonstrates the number of possible solutions for the first level model in terms of the number of binary variables.

The steps for solving bi-level integer programming (BLIP) by complete enumeration (CE) are as follows.

**Step 1:** Consider negative infinity for the initial candidate value of the first level objective function.

**Step 2:** Identify a feasible solution for the first level decision variables.

**Step 3:** Set the values for the first level variables as parameters in the second level model.

**Step 4:** Solve the second level model.

**Step 5:** If the optimal solution for the second level model is obtained, calculate the value of the first level objective function.

**Step 6:** If the value obtained for the first level objective function is better than the candidate’s value, it should be replaced with the previous one. Then save the values of the first and second level variables.

**Step 7:** Repeat steps 1 through 6 for all feasible solutions for the first level model.

**Step 8:** The candidate’s values are identified as the optimal solution to the bi-level problem.
The complete enumeration method can only be used for models whose number of binary variables in the first level model is approximately not more than 20 variables.

4. GPBLO algorithm

Because the computational time for bi-level optimization problems is very long, even for cases that KKT conditions can be used or the complete enumeration method is applied, a faster way to solve such problems is needed. Therefore, we propose a method that can identify the optimal or near-optimal solution for bi-level problems in a relatively short time. The result of this method can be used as a candidate solution for applying exact methods such as branch and bound or branch and cut to reduce their computational time, significantly.

To solve a variety of bi-level models, we present an algorithm that can identify either the optimal solution or the near-optimal solution by solving a limited number of single-level mathematical models. Because in this method we utilize solving several goal programming problems to find the appropriate solution for the bi-level optimization which is called the GPBLO. In this method, first, the leader and follower objective functions are normalized by identifying the best and worst possible logical values for them according to fuzzy normalization. Then by considering the importance coefficient of $w$ for the leader's objective function and the importance coefficient $(1-w)$ for the follower's objective function, they are aggregated to a single form.

$$f = (w) \frac{f_1 - f_{1,\text{min}}}{f_{1,\text{max}} - f_{1,\text{min}}} + (1-w) \frac{f_2 - f_{2,\text{min}}}{f_{2,\text{max}} - f_{2,\text{min}}}$$

To identify the best possible value for $f1$, we release the bi-level model from the objective function of the follower. Then, by solving a resulting single-level model, we obtain an upper bound for $f1$, which we call $f_{1,\text{max}}$. With the values obtained for variables $X$ and $Y$, we calculate the value of $f2$. This value is the lower bound value for $f2$, which we call $f_{2,\text{min}}$. 


To identify the upper bound value for $f_2$, we release the bi-level model from the leader's objective function. Then, by solving a single-level model resulted, we obtain an upper bound for $f_2$, called $f^{2\max}_2$. With the values obtained for variables $X$ and $Y$, we calculate the value of $f_1$. This is the lower bound value for $f_1$, which we call $f^{1\min}_1$. The coefficient $w$ is a parameter whose values Changes from 0 to 1. The increment step of this parameter can be any arbitrary number between 0 and 1. With smaller values for $w$, the better solutions may be detected, but the amount and time of calculations will increase. In the computational experiments indicated in this paper, we consider this step to be 0.1. In this case, we need to solve 11 bi-objective models, and for each of them, the second level model should be solved by fixing the values of the first-level variables. Also, two mathematical models must be solved to identify the best and worst rational values for $f_1$ and $f_2$. So in this case, we need to solve only 24 mathematical models. If the first and second level models are linear and the decision variables are also continuous, the goal programming model will also be linear. Also, the goal programming model's type is the same as the more complicated model's type between leader and follower.

The goal programming model derived from the relaxed problem from the second level objective function and also the replacement of the function $f$ with the objective function of the first level can be stated as follows:

$$ f = (w) \frac{f^{1\min}_1 - f^{1\min}_1}{f^{1\max}_1 - f^{1\min}_1} + (1-w) \frac{f^{2\min}_2 - f^{2\min}_2}{f^{2\max}_2 - f^{2\min}_2} $$  \hspace{1cm} (45)

Subject to

$$ \sum_{j_1} A_{i_1}(i_1, j_1)X(j_1) + \sum_{j_2} B_{i_1}(i_1, j_2)Y(j_2) \leq G(i_1) \hspace{1cm} \forall i_1 $$  \hspace{1cm} (46)

$$ \sum_{j_1} A_{i_2}(i_2, j_1)X(j_1) + \sum_{j_2} B_{i_2}(i_2, j_2)Y(j_2) \leq G(i_2) \hspace{1cm} \forall i_2 $$  \hspace{1cm} (47)

$$ X(j_1) \geq 0 \hspace{1cm} \forall j_1 $$  \hspace{1cm} (48)

$$ Y(j_2) \geq 0 \hspace{1cm} \forall j_2 $$  \hspace{1cm} (49)
The steps to solve bi-level programming (BLP) by the GPBLO are as follows.

**Step 1:** Normalize the first and second level objective functions with a fuzzy normalization method.

**Step 2:** Consider the numerical coefficient \( w \) for the first level normalized objective function and the numerical coefficient \((1-w)\) for the second level normalized objective function.

**Step 3:** Integrate two objectives in a new single objective function for the problem.

**Step 4:** Ignore first and second level objective functions and consider negative infinity for the initial candidate value of the first level objective function.

**Step 5:** Set the value of \( w \) with 0.

**Step 6:** Solve the resulting single-level model.

**Step 7:** Consider the values obtained for the first level variables as parameters for the second level model.

**Step 8:** Solve the second level model.

**Step 9:** If the solution to the second level model is optimal, then calculate the value of the first level objective function.

**Step 10:** If the value obtained for the first level objective function is better than the candidate's value, update the best found, and save the values of the first and second level variables.

**Step 11:** Increase the value of \( w \) by 0.1 and repeat steps 6 through 10 until \( w \) reaches 1.

**Step 12:** The solution given to the candidate's value and variables are the solution for the bi-level problem.

5. **Computational Analysis**

In this section, we first randomly generated 20 sample bi-level problems in different sizes that can be solved using the KKT method. These problems are linear-linear and continuous-continuous. Then we solve them using the CPLEX solver. We also solved these problems using
the GPBLO algorithm and a metaheuristic method based on Particle Swarm Optimization (PSO). Bi-level PSO has been used in many published papers to solve bi-level models. Results of these methods are presented in Table 2. The values of problem parameters are generated using the uniform distribution. These values are presented in Table 3.

With increasing problem size, the time of the MIP method is very long. It can be observed that in these 20 instances, the value of the objective function approximately has a 3% gap with the optimal objective value. This is while the computational time, on average, is much lower. Then, 20 bi-level problems were generated in various sizes. These problems are linear-linear and discrete-discrete. We consider the follower model as an integer linear programming type. Due to the non-convexity of this type of model, the condition of KKT is no longer a necessary and sufficient condition for its optimality. Therefore, these problems cannot be solved by the KKT method. But because the first level variables are binary, we can obtain their optimal solutions by the complete enumeration (CE) method. Each iteration will be solved by the CPLEX Solver and then the best solution among them will be identified as the optimal solution. We also solve these problems using the GPBLO method to compare the results. The values of problem parameters are generated similar to Table 3 except for the parameter $G_1(i_1)$ which is generated randomly by a uniform(5,9) distribution. The results are demonstrated in Table 4. It can be observed that in these 20 cases, the value of the objective function is 1% less than the optimal objective value with much lower computational time.

Figure 1 illustrates comparing KKT with GPBLO in terms of computing times (in seconds), and Figure 2 illustrates comparing CE with GPBLO in terms of computing times (in seconds).

In the following, several sample problems designed to test bi-level optimization algorithms are considered. These problems are presented in [18], [21], [27]. We solve these problems using
the GPBLO algorithm as well as by the model obtained from KKT conditions or by the Complete Enumeration (CE) method.

Instance 1 (Zhao et al., 2017 [16])

\[
\max_{x_1,x_2} F_1 = -18x_1 + 10x_2 + 11y_1 + 11y_2 + 23y_3 + 40y_4
\]

s.t.

\[
\max_{y_1,y_2,y_3,y_4} F_2 = -35x_1 - 9x_2 + 20y_1 - 44y_2 + 10y_3 + 7y_4
\]

s.t.

\[
\begin{align*}
47x_1 - 14x_2 - y_1 + 4y_2 + y_3 - 49y_4 & \leq 1.5 \\
-23x_1 + 2x_2 + 45y_1 - 35y_2 + 12y_3 + 41y_4 & \leq 13.5 \\
-9x_1 - 18x_2 + 12y_1 + 13y_2 + 37y_3 - 11y_4 & \leq 5.5 \\
6x_1 - 19x_2 - y_1 - 2y_2 - 49y_3 - 11y_4 & \leq -43.5 \\
-31x_1 - 8x_2 + 2y_1 + 17y_2 + 47y_3 - 25y_4 & \leq 6.3 \\
46x_1 + 3x_2 - 28y_1 + 17y_2 - 36y_3 - 3y_4 & \leq 22.5 \\
-45x_1 + 34x_2 - 44y_1 + 44y_2 + 16y_3 - 2y_4 & \leq 17 \\
29x_1 - 13x_2 + 38y_1 + 19y_2 - 2y_3 + 7y_4 & \leq 39 \\
13x_1 + 10x_2 + 27y_1 - 29y_2 - 49y_3 - 38y_4 & \leq -38 \\
x_1,x_2,y_1,y_2,y_3,y_4 & \geq 0
\end{align*}
\]

Zhao et al. [16] provide three identified solutions to this problem that have been obtained using three different metaheuristic methods. Given that the second-level model in this problem is linear programming, using the KKT conditions, as previously stated, this model can be converted as a single-level model and its global optimal solution can be obtained. This problem was also solved using the GPBLO method. The results are presented in Table 5. As can be seen, the GPBLO solution is the same as the KKT solution and are significantly better than the previously identified solutions.

Instance 2 (Zhao et al., 2017 [16])
\[
\begin{align*}
\max_{x_1, x_2} & : F_1 = 5x_1 + 2x_2 + 4y_1 \\
\text{s.t.} & \\
\max_{y_1, y_2} & : F_2 = 3x_2 + 5y_1 - 2y_2 \\
\text{s.t.} & \\
2x_1 + 2x_2 + 2y_1 + 4y_2 & \leq 8 \\
x_1 + x_2 + y_1 & \leq 2 \\
x_2 + x_3 + y_2 & \leq 3 \\
x_1 & \leq 4 \\
x_2 & \leq 4 \\
y_1 & \leq 2 \\
y_2 & \leq 2 \\
x_1, x_2, y_1, y_2 & \geq 0
\end{align*}
\]

For this problem, Zhao et al. [16] also, provide three identified solutions that have been obtained using three different metaheuristic methods. Given that the second-level model in this problem is linear programming, using the KKT conditions, as previously stated, this model can be converted as a single-level model and its global optimal solution can be obtained. This problem was also solved using the GPBLO method. The results are presented in Table 6. As can be seen, the GPBLO solution is the same as the KKT solution and are significantly better than the previously identified solutions.

Instance 3 (Kuo et al., 2011 [27])
max : $F_1 = 110x_1 + 120x_2 - 40y_1 - 50y_2$

s.t.

min : $F_2 = 130x_1 + 145x_2$

s.t.

$x_1 \leq y_1$

$x_2 \leq y_2$

$y_1 \leq 1000$

$y_2 \leq 500$

$y_1 + y_2 \geq 750$

$x_1, x_2, y_1, y_2 \geq 0$

Kuo et al. [27] provide three identified solutions to this problem that have been obtained using three different metaheuristic methods. Given that the second-level model in this problem is linear programming, using the KKT conditions, as previously stated, this model can be converted as a single-level model and its global optimal solution can be obtained. This problem was also solved using the GPBLO method. The results are presented in Table 7. As can be seen, the GPBLO solution is the same as the KKT solution and are better than the previously identified solutions.

Instance 4 (Tahernejad et al., 2020 [25])

max : $F_1 = x + 10y$

s.t.

min : $F_2 = y$

s.t.

$- 25x + 20y \leq 30$

$x + 2y \leq 10$

$2x - y \leq 15$

$2x + 10y \geq 15$

$x, y \in Z^+$

Tahernejad et al., [25] solved this problem using a metaheuristic method. Because in this problem, the second-level model is integer programming, the KKT method cannot be used to
solve it. However, considering that the decision variables of the first-level model are integers, the global optimal solution can be obtained by using the CE method. This problem was also solved using the GPBLO method. The results are presented in Table 8. As can be seen, the results of these three methods are equal to each other.

6. Conclusions and future research

In this paper, we presented a mat-heuristic algorithm called GPBLO for solving bi-level optimization problems. Numerical experiments on small, medium, and large-sized instances whose optimal solution is obtained by solving a single-level model using KKT conditions or by a CE method indicated algorithm efficiency. Several sample problems designed to test bi-level optimization algorithms were considered. We solved these problems using the GPBLO algorithm as well by KKT conditions or CE methods. In some instances of these problems, the solution obtained from the GPBLO algorithm was the same as the global optimal solution and were better than the previously identified solutions. Therefore, this method can be used to identify the solutions that may be optimal or near-optimal. For future research, it is possible to design exact algorithms that can identify the optimal solution in an acceptable time.

Reference


(2019).

24. Dempe, S. and Zemkoho, A.B. “KKT reformulation and necessary conditions for
optimality in nonsmooth bilevel optimization”, *SIAM Journal on Optimization*, 24(4), pp.1639-
1669 (2014).

integer bilevel linear optimization problems and its implementation”, *Mathematical

aggregate production planning considering pricing decisions in competitive environment: A

27. Kuo, R.J. and Han, Y.S. “A hybrid of genetic algorithm and particle swarm optimization
for solving bi-level linear programming problem–A case study on supply chain model”,

**Biographies**

**Hossein Shams Shemirani** received his BSc in industrial engineering from the Sharif
University of Technology, Tehran, Iran. He also received his MSc in industrial engineering
from the Sharif University of Technology. He is currently a Ph.D. student, and a faculty of the
industrial engineering group at the Golpayegan University of Technology, Isfahan, Iran. His
research interests include operations research, maintenance management, and management
information systems.

**Rashed Sahraeian** is a Professor of Industrial Engineering at Shahed University. He holds a
B.Sc. in Industrial Engineering from Amirkabir University of Technology, M.Sc., and Ph.D.
from Tarbiat Modares University. His research interests include operations research, supply
chain management, and scheduling. He has published some books and a large number of papers
in high-impact international journals and many papers in research indexed journals and international conferences.

**Mahdi Bashiri** is a Professor of Industrial Engineering at Shahed University. He holds a B.Sc. in Industrial Engineering from Iran University of Science and Technology, M.Sc. and Ph.D. from Tarbiat Modares University. His research interests are facilities design, meta-heuristics, and multi-response optimization. He has published some books and a large number of papers in high-impact international journals and many papers in research indexed journals and international conferences.
Figure and table captions

Figure 1: Comparing KKT with GPBLO in terms of computing times (in seconds).
Figure 2: Comparing CE with GPBLO in terms of computing times (in seconds).

Table 1: The number of possible solutions based on the number of binary variables
Table 2: Comparing the KKT method with the GPBLO
Table 3: The values of problem parameters
Table 4: Comparing the CE method with the GPBLO
Table 5: The best solutions of instance 1 for different methods.
Table 6: The best solutions of instance 2 for different methods.
Table 7: The best solutions of instance 3 for different methods.
Table 8: The best solutions of instance 4 for different methods.
Figure 1: Comparing KKT with GPBLO in terms of computing times (in seconds).

Figure 2: Comparing CE with GPBLO in terms of computing times (in seconds).
Table 1: The number of possible solutions based on the number of binary variables

<table>
<thead>
<tr>
<th>n</th>
<th>Number of Solutions</th>
<th>n</th>
<th>Number of Solutions</th>
<th>n</th>
<th>Number of Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>11</td>
<td>2,048</td>
<td>21</td>
<td>2,097,152</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>12</td>
<td>4,096</td>
<td>22</td>
<td>4,194,304</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>13</td>
<td>8,192</td>
<td>23</td>
<td>8,388,608</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>14</td>
<td>16,384</td>
<td>24</td>
<td>16,777,216</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>15</td>
<td>32,768</td>
<td>25</td>
<td>33,554,432</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>16</td>
<td>65,536</td>
<td>26</td>
<td>67,108,864</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>17</td>
<td>131,072</td>
<td>27</td>
<td>134,217,728</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>18</td>
<td>262,144</td>
<td>28</td>
<td>268,435,456</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>19</td>
<td>524,288</td>
<td>29</td>
<td>536,870,912</td>
</tr>
<tr>
<td>10</td>
<td>1,024</td>
<td>20</td>
<td>1,048,576</td>
<td>30</td>
<td>1,073,741,824</td>
</tr>
</tbody>
</table>

Table 2: Comparing the KKT method with the GPBLO

<table>
<thead>
<tr>
<th>Ins.</th>
<th># X&lt;sup&gt;a&lt;/sup&gt;</th>
<th># Y&lt;sup&gt;b&lt;/sup&gt;</th>
<th># C1&lt;sup&gt;c&lt;/sup&gt;</th>
<th># C2&lt;sup&gt;d&lt;/sup&gt;</th>
<th>KKT</th>
<th>GPBLO</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F1&lt;sup&gt;e&lt;/sup&gt;</td>
<td>Time&lt;sup&gt;f&lt;/sup&gt;</td>
<td>F1&lt;sup&gt;e&lt;/sup&gt;</td>
<td>Time&lt;sup&gt;f&lt;/sup&gt;</td>
<td>Gap</td>
<td>F1&lt;sup&gt;e&lt;/sup&gt;</td>
<td>Time&lt;sup&gt;f&lt;/sup&gt;</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>411.7</td>
<td>0.1</td>
<td>411.7</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>357.9</td>
<td>0.1</td>
<td>357.9</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>439.0</td>
<td>0.2</td>
<td>439.0</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>514.2</td>
<td>0.1</td>
<td>514.2</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>519.0</td>
<td>0.2</td>
<td>518.9</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>20</td>
<td>457.0</td>
<td>0.5</td>
<td>449.7</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>50</td>
<td>10</td>
<td>30</td>
<td>399.2</td>
<td>1.6</td>
<td>384.6</td>
</tr>
<tr>
<td>8</td>
<td>30</td>
<td>60</td>
<td>20</td>
<td>40</td>
<td>384.8</td>
<td>2.0</td>
<td>377.0</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>70</td>
<td>30</td>
<td>50</td>
<td>364.1</td>
<td>2.8</td>
<td>357.0</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
<td>80</td>
<td>40</td>
<td>60</td>
<td>364.6</td>
<td>3.5</td>
<td>356.3</td>
</tr>
<tr>
<td>11</td>
<td>60</td>
<td>90</td>
<td>50</td>
<td>70</td>
<td>345.9</td>
<td>9.8</td>
<td>319.3</td>
</tr>
<tr>
<td>12</td>
<td>70</td>
<td>100</td>
<td>60</td>
<td>80</td>
<td>358.5</td>
<td>9.6</td>
<td>329.5</td>
</tr>
<tr>
<td>13</td>
<td>80</td>
<td>110</td>
<td>70</td>
<td>90</td>
<td>353.1</td>
<td>8.8</td>
<td>323.5</td>
</tr>
<tr>
<td>14</td>
<td>90</td>
<td>120</td>
<td>80</td>
<td>100</td>
<td>345.8</td>
<td>16.1</td>
<td>328.8</td>
</tr>
<tr>
<td>15</td>
<td>100</td>
<td>130</td>
<td>90</td>
<td>110</td>
<td>358.5</td>
<td>21.3</td>
<td>351.6</td>
</tr>
<tr>
<td>16</td>
<td>110</td>
<td>140</td>
<td>100</td>
<td>120</td>
<td>334.6</td>
<td>20.5</td>
<td>323.4</td>
</tr>
<tr>
<td>17</td>
<td>120</td>
<td>150</td>
<td>110</td>
<td>130</td>
<td>346.4</td>
<td>67.0</td>
<td>330.1</td>
</tr>
<tr>
<td>18</td>
<td>130</td>
<td>160</td>
<td>120</td>
<td>140</td>
<td>351.9</td>
<td>75.1</td>
<td>330.5</td>
</tr>
<tr>
<td>19</td>
<td>140</td>
<td>170</td>
<td>130</td>
<td>150</td>
<td>354.1</td>
<td>90.3</td>
<td>324.5</td>
</tr>
<tr>
<td>20</td>
<td>150</td>
<td>180</td>
<td>140</td>
<td>160</td>
<td>349.9</td>
<td>131.7</td>
<td>339.3</td>
</tr>
</tbody>
</table>

Mean 3%  Mean 5%

---

a  The number of variables in the first level model
b  The number of variables in the second level model
c The number of constraints in the first level model
d The number of constraints in the second level model
e The value of the objective function of the first level model
f Calculation time in seconds

Table 3: The values of problem parameters

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C1(j1) = Uniform(10,20)</td>
<td>C2(j1) = Uniform(10,20)</td>
</tr>
<tr>
<td>D1(j2) = Uniform(10,20)</td>
<td>D2(j2) = Uniform(10,20)</td>
</tr>
<tr>
<td>A1(i1,j1) = Uniform(1,5)</td>
<td>A2(i2,j1) = Uniform(1,5)</td>
</tr>
<tr>
<td>B1(i1,j2) = 0</td>
<td>B2(i2,j2) = Uniform(1,5)</td>
</tr>
<tr>
<td>G1(i1) = Uniform(50,90)</td>
<td>G2(i2) = Uniform(50,90)</td>
</tr>
</tbody>
</table>
Table 4: Comparing the CE method with the GPBLO

<table>
<thead>
<tr>
<th>Ins.</th>
<th># X^a</th>
<th># Y^b</th>
<th># C1^c</th>
<th># C2^d</th>
<th>CE f</th>
<th>GPBLO</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F1^g</td>
<td>Time^h</td>
<td>F1^g</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>284</td>
<td>2.1</td>
<td>284</td>
</tr>
<tr>
<td>22</td>
<td>6</td>
<td>10</td>
<td>1</td>
<td>5</td>
<td>476</td>
<td>4.1</td>
<td>474</td>
</tr>
<tr>
<td>23</td>
<td>7</td>
<td>15</td>
<td>1</td>
<td>10</td>
<td>431</td>
<td>5.6</td>
<td>426</td>
</tr>
<tr>
<td>24</td>
<td>8</td>
<td>20</td>
<td>1</td>
<td>15</td>
<td>360</td>
<td>12.1</td>
<td>356</td>
</tr>
<tr>
<td>25</td>
<td>9</td>
<td>25</td>
<td>1</td>
<td>20</td>
<td>336</td>
<td>7.9</td>
<td>336</td>
</tr>
<tr>
<td>26</td>
<td>10</td>
<td>30</td>
<td>1</td>
<td>25</td>
<td>325</td>
<td>39.6</td>
<td>314</td>
</tr>
<tr>
<td>27</td>
<td>11</td>
<td>35</td>
<td>1</td>
<td>30</td>
<td>313</td>
<td>22.0</td>
<td>313</td>
</tr>
<tr>
<td>28</td>
<td>12</td>
<td>40</td>
<td>1</td>
<td>35</td>
<td>336</td>
<td>27.9</td>
<td>333</td>
</tr>
<tr>
<td>29</td>
<td>13</td>
<td>45</td>
<td>1</td>
<td>40</td>
<td>301</td>
<td>15.2</td>
<td>298</td>
</tr>
<tr>
<td>30</td>
<td>14</td>
<td>50</td>
<td>1</td>
<td>45</td>
<td>295</td>
<td>42.9</td>
<td>282</td>
</tr>
<tr>
<td>31</td>
<td>15</td>
<td>55</td>
<td>1</td>
<td>50</td>
<td>313</td>
<td>112.2</td>
<td>313</td>
</tr>
<tr>
<td>32</td>
<td>16</td>
<td>60</td>
<td>1</td>
<td>55</td>
<td>303</td>
<td>287.0</td>
<td>294</td>
</tr>
<tr>
<td>33</td>
<td>17</td>
<td>65</td>
<td>1</td>
<td>60</td>
<td>279</td>
<td>483.9</td>
<td>279</td>
</tr>
<tr>
<td>34</td>
<td>18</td>
<td>70</td>
<td>1</td>
<td>65</td>
<td>270</td>
<td>375.6</td>
<td>250</td>
</tr>
<tr>
<td>35</td>
<td>19</td>
<td>75</td>
<td>1</td>
<td>70</td>
<td>293</td>
<td>248.6</td>
<td>293</td>
</tr>
<tr>
<td>36</td>
<td>20</td>
<td>80</td>
<td>1</td>
<td>75</td>
<td>278</td>
<td>515.7</td>
<td>272</td>
</tr>
<tr>
<td>37</td>
<td>21</td>
<td>85</td>
<td>1</td>
<td>80</td>
<td>319</td>
<td>590.4</td>
<td>315</td>
</tr>
<tr>
<td>38</td>
<td>22</td>
<td>90</td>
<td>1</td>
<td>85</td>
<td>---*</td>
<td>---*</td>
<td>301</td>
</tr>
<tr>
<td>39</td>
<td>23</td>
<td>95</td>
<td>1</td>
<td>90</td>
<td>---*</td>
<td>---*</td>
<td>263</td>
</tr>
<tr>
<td>40</td>
<td>24</td>
<td>100</td>
<td>1</td>
<td>95</td>
<td>---*</td>
<td>---*</td>
<td>282</td>
</tr>
</tbody>
</table>

| Mean | 1% | Mean | 4% |

a  The number of variables in the first level model  
b  The number of variables in the second level model  
c  The number of constraints in the first level model  
d  The number of constraints in the second level model  
e  Number of possible solutions  
f  Complete Enumerations  
g  The value of the objective function of the first level model  
h  Calculation time in seconds  
x  The algorithm was not performed due to the long calculation time.
Table 5: The best solutions of instance 1 for different methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>$x^*$</th>
<th>$y^*$</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Zheng et al., 2014) [16]</td>
<td>(0.897, 1.128)</td>
<td>(0.000, 0.075, 1.048, 0.534)</td>
<td>39.789</td>
<td>-30.561</td>
</tr>
<tr>
<td>(Huang et al. 2015) [16]</td>
<td>(0.890, 1.125)</td>
<td>(0.000, 0.071, 1.045, 0.529)</td>
<td>39.613</td>
<td>-30.625</td>
</tr>
<tr>
<td>(Zhao et al. 2017) [16]</td>
<td>(0.894, 1.127)</td>
<td>(0.000, 0.074, 1.047, 0.532)</td>
<td>39.724</td>
<td>-30.463</td>
</tr>
<tr>
<td>KKT</td>
<td>(1.326, 1.289)</td>
<td>(0.000, 0.332, 1.257, 0.926)</td>
<td>51.311</td>
<td>-53.582</td>
</tr>
<tr>
<td>GPBLO</td>
<td>(1.326, 1.289)</td>
<td>(0.000, 0.332, 1.257, 0.926)</td>
<td>51.311</td>
<td>-53.582</td>
</tr>
</tbody>
</table>

Table 6: The best solutions of instance 2 for different methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>$x^*$</th>
<th>$y^*$</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Zheng et al., 2014) [16]</td>
<td>(0.000, 0.000)</td>
<td>(2.000, 0.000)</td>
<td>8.000</td>
<td>10.000</td>
</tr>
<tr>
<td>(Huang et al. 2015) [16]</td>
<td>(1.000, 0.000)</td>
<td>(1.000, 0.000)</td>
<td>9.000</td>
<td>5.000</td>
</tr>
<tr>
<td>(Zhao et al. 2016) [16]</td>
<td>(0.438, 0.000)</td>
<td>(1.563, 0.000)</td>
<td>8.437</td>
<td>7.813</td>
</tr>
<tr>
<td>KKT</td>
<td>(2.000, 0.000)</td>
<td>(0.000, 0.000)</td>
<td>10.000</td>
<td>6.000</td>
</tr>
<tr>
<td>GPBLO</td>
<td>(2.000, 0.000)</td>
<td>(0.000, 0.000)</td>
<td>10.000</td>
<td>6.000</td>
</tr>
</tbody>
</table>
Table 7: The best solutions of instance 3 for different methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>x*</th>
<th>y*</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>HGAPSO-1 (Kuo et al., 2011) [27]</td>
<td>(999.265, 496.392)</td>
<td>(999.265, 496.392)</td>
<td>104229.91</td>
<td>201325.49</td>
</tr>
<tr>
<td>HGAPSO-2 (Kuo et al., 2011) [27]</td>
<td>(999.470, 499.503)</td>
<td>(999.470, 499.503)</td>
<td>104775.37</td>
<td>202175.61</td>
</tr>
<tr>
<td>HGAPSO-3 (Kuo et al., 2011) [27]</td>
<td>(999.932, 499.933)</td>
<td>(999.932, 499.933)</td>
<td>104990.20</td>
<td>202481.01</td>
</tr>
<tr>
<td>KKT</td>
<td>(1000.000, 500.000)</td>
<td>(1000.000, 500.000)</td>
<td>105000.00</td>
<td>202500.00</td>
</tr>
<tr>
<td>GPBLO</td>
<td>(1000.000, 500.000)</td>
<td>(1000.000, 500.000)</td>
<td>105000.00</td>
<td>202500.00</td>
</tr>
</tbody>
</table>

Table 8: The best solutions of instance 4 for different methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>x*</th>
<th>y*</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Tahernejad et al., 2020) [25]</td>
<td>2.000</td>
<td>2.000</td>
<td>22.00</td>
<td>2.00</td>
</tr>
<tr>
<td>CE</td>
<td>2.000</td>
<td>2.000</td>
<td>22.00</td>
<td>2.00</td>
</tr>
<tr>
<td>GPBLO</td>
<td>2.000</td>
<td>2.000</td>
<td>22.00</td>
<td>2.00</td>
</tr>
</tbody>
</table>