A Genetic Approach for Solving Bi-Level Programming Problems

1Eghbal Hosseini*, Isa Nakhai Kamalabadib

aPh.D. Candidate at Payamenur University of Tehran, Department of Mathematics, Tehran, Iran
bProfessor of Industrial Engineering at University of Kurdistan, Department of Industrial Engineering, Sanandaj, Iran

Abstract

The bi-level programming problem (BLPP) however theoretically is interested by many researchers but in recent years it is known as a suitable method for solving the real and complex problems in applicable areas. There are several forms of the BLPP as an NP-hard problem. The literature shows a few attempts for using meta-heuristic algorithms and in this paper we show an effective method based on genetic algorithm (GA) for solving such problems. To obtain efficient upper bounds and lower bounds we use the Karush -Kuhn -Tucker (KKT) conditions for transforming the BLPP into single level problem. Thus by using the proposed GA, the single problems are solved. The proposed approach achieves ficient and feasible solutions and they are evaluated by comparing with references and test problems.


1. Introduction

* Corresponding author
Email address: eghbal_math@yahoo.com

AMO - Advanced Modeling and Optimization. ISSN: 1841-4311

719
The bi-level programming problem (BLPP) is a nested optimization problem, which has two levels in hierarchy. The first level is called leader and the second one is called follower. They have their own objective functions and constraints. It has been proved that the BLPP is NP-Hard problem even to seek for the locally optimal solution [1, 2]. Nonetheless the BLPP is an applicable problem and practical tool to solve decision making problems. It is used in several areas such as economic, traffic, finance and so on. Therefore finding the optimal solution has a special importance to researchers.

Several algorithms have been presented for solving the BLPP [3, 4, 11, 12, 13]. These algorithms are divided into the following classes: Transformation methods [3, 4], Fuzzy methods [5, 6, 7, 8], Global techniques [9, 10, 11, 12], Primal–dual interior methods [13], Enumeration methods [14], Meta heuristic approaches [15, 16, 17, 18, 19].

In this paper, we consider two forms of the BLPP: the linear-quadratic bi-level programming (LQBP) which the objective function of the upper level is linear and the objective function of the lower level is quadratic and the linear-fractional bi-level programming (LFBP) which the objective function of the upper level is fractional and the objective function of the lower level is linear. We present a procedure based on genetic algorithm to solve these two problems. In the remaining of pages, in section 2, basic concepts of LQBP and LFBP are proposed. We provide the GA to solve LQBP and LFBP in section 3. Section 4 describes the steps of presented algorithm. Computational results are proposed in Section 5 and finally, the paper is finished in section 6.

2. The concepts and properties
A Genetic Approach for Solving Bi-Level Programming Problems

We research two special classes of bi-level programming: linear-quadratic bi-level programming (LQBP) and Linear-fractional bi-level programming (LFBP). The LQBP is formulated as follows [16]:

$$\begin{align*}
\max_x & \quad f(x, y) = a^T x + b^T y \\
\text{s.t} & \quad \max_y \quad g(x, y) = c^T x + d^T y + (x^T, y^T)Q(x^T, y^T)^T \\
& \quad s.t \quad Ax + By \leq r, \\
& \quad x, y \geq 0.
\end{align*}$$

Where $a, c \in \mathbb{R}^{n_1}, b, d \in \mathbb{R}^{n_2}, A \in \mathbb{R}^{m \times n_1}, B \in \mathbb{R}^{m \times n_2}, r \in \mathbb{R}^m, x \in \mathbb{R}^{n_1}, y \in \mathbb{R}^{n_2}$ and $f(x, y), g(x, y)$ are the objective functions of the leader and the follower, respectively. Also $Q \in \mathbb{R}^{n_1+n_2 \times n_1+n_2}$ is symmetric positive semi–definite matrix.

Suppose that

$$Q = \begin{bmatrix}
Q_2 & Q_1^T \\
Q_1 & Q_0
\end{bmatrix}$$

Which

$$Q_0 \in \mathbb{R}^{n_1 \times n_1}, Q_1 \in \mathbb{R}^{n_1 \times n_2}, Q_2 \in \mathbb{R}^{n_2 \times n_2}.$$ 

Then the follower problem of the LQBP is

$$\begin{align*}
\max_y & \quad g(x, y) = d^T y + 2Q_1 xy + y^T Q_0 y \\
\text{s.t} & \quad By \leq r - Ax, \\
& \quad y \geq 0.
\end{align*}$$

The LFBP problem is formulated as follows [21]:

$$\begin{align*}
\max_x & \quad f(x, y) = \frac{a_1 + a_2^T x + a_3^T y}{b_1 + b_2^T x + b_3^T y} \\
\text{s.t} & \quad \max_y \quad g(x, y) = c^T x + d^T y \\
& \quad s.t \quad Ax + By \leq r, \\
& \quad x, y \geq 0.
\end{align*}$$
Which $a_i, b_i \in R, a_2, b_2, c \in R^{n_1}, a_3, b_3, d \in R^{n_2}, A \in R^{m \times n_1}, B \in R^{m \times n_2}, r \in R^m, x \in R^{n_1}, y \in R^{n_2}$.

The feasible region of the LQBP and LFBP problems is

$$S = \{(x, y) | Ax + By \leq r, x, y \geq 0\}. \quad (4)$$

On the other hand if $x$ be fixed, the feasible region of the follower can be explained as

$$S(x) = \{y | By \leq r - Ax, x, y \geq 0\}. \quad (5)$$

Based on the above assumptions the follower rational reaction set is

$$P(x) = \{y | y \in \arg \max \{g(x, y) | y \in S(x)\}\}. \quad (6)$$

Where the inducible region is as follows

$$IR = \{(x, y) \in S, y \in P(x)\}. \quad (7)$$

Finally the bi-level programming problem can be written as

$$\max \{f(x, y) | (x, y) \in IR\}. \quad (8)$$

If there is finite solution for the BLP problem, we define feasibility and optimality for the BLP problem as

$$S = \{(x, y) | Ax + By \leq r, x, y \geq 0\}. \quad (9)$$

**Definition 1:**

$(x, y)$ is a feasible solution to bi-level problem if $(x, y) \in IR$. 

722
Definition 2:

\((x^*, y^*)\) is an optimal solution to the bi-level problem if

\[ f(x^*, y^*) \leq f(x, y) \quad \forall (x, y) \in IR. \]  

(10)

3. Genetic algorithm to solve BLPP

In this section, basic and general concepts related to genetic proposed algorithm are discussed. Genetic algorithms are global methods that are used for global searches. As the previous researchers indicate [11, 15, 16] the basic characteristics of these algorithms consist of:

1. Initial population of solution is produced randomly. Some of the genetic algorithms use other Meta heuristic method to produce the initial population.
2. Genetic algorithms use a lot of feasible solutions. Therefore they usually avoid local optimal solutions.
3. Genetic algorithms used to solve very large problems with many variables.
4. These algorithms are simple and do not need extra conditions such as continuity and differentiability of objective functions.
5. Genetic algorithms usually gain several optimal solutions instead unique optimal solution. This property is useful for multi objective function and multi-level programming.

In the proposed genetic algorithm, each feasible solution of BLPP usually is transformed by string of characters from the binary alphabet that is called chromosome. The genetic algorithm works as follows:

Initial generation, that is generated randomly, is divided in overall the feasible space similarly. Then chromosomes are composed together to construct new generation. This process continues till to get appropriate optimal solution. The general genetic algorithm process as follows:
Algorithm 1: GA to solve BLPP

1: \( t = 0 \)
2: initialize \( P(t) \)
3: evaluate \( P(t) \)
4: While not terminate do
5: \( P'(t) = \text{recombine } P(t) \)
6: \( P''(t) = \text{mutate } P'(t) \) \hspace{1cm} (11)
7: evaluate \( P''(t) \)
8: \( P(t+1) = \text{select } (P''(t) \cup Q) \)
9: \( t = t + 1 \)
10: End of While
11: End.

Where \( P(t) \) is a population of chromosomes in \( t\)-th generation and \( Q \) is a set of chromosomes in the current generation which are selected.

In the suggested method, every chromosome is demonstrated by a string. This string consists of \( m + n_2 \) binary components. Also these chromosomes are applied in the following problems that they are created by using Karush -Kuhn –Tucker (KKT) conditions for LQBP and LFBP respectively:

\[
\begin{align*}
\text{max} & \quad a^T x + b^T y \\
\text{s.t} & \quad Ax + By + w = r \\
& \quad 2Q_1 x + 2Q_0 y - Bu + v = -d \\
& \quad uw = 0, \quad vy = 0, \quad x, y, u, v, w \geq 0.
\end{align*}
\]
A Genetic Approach for Solving Bi-Level Programming Problems

Now the chromosomes are applied according the following rules [15]:

If the \( i \)-th component of the chromosome is equal to zero, then \( u_i = 0 \), \( w_i \geq 0 \).

Else \( u_i \geq 0 \), \( w_i = 0 \).

If the \( j \)-th component of the chromosome is equal to zero, then \( v_j = 0 \), \( y_j \geq 0 \).

Else \( v_j \geq 0 \), \( y_j = 0 \).

**Theorem 1:**

\((x^*, y^*)\), is the optimal solution to the problem (1) if and only if there exists 
\(u^*\), \(w^*\), \(v^*\) such that \((x^*, y^*, u^*, w^*, v^*)\) is the solution of the problem (12).

**Proof:**

The proof of this theorem was given by [16].

**Theorem 2:**

\((x^*, y^*)\), is the optimal solution to the problem (3) if and only if there exists 
\(u^*\), \(w^*\), \(v^*\) such that \((x^*, y^*, u^*, w^*, v^*)\) is the solution of the problem (13).

**Proof:**

The proof of theorem 2 was proposed by [12].

**4. Steps of our algorithm**

In this section, the algorithm steps are proposed.
Step 1: Generating the initial population.

The initial population includes solutions in the feasible region that are called achievable chromosomes. These chromosomes are generated by solving the following problem $S$ to the LQBP and LFBP respectively:

$$\begin{align*}
\max & \quad (d + 2Q_s x)^T y + y^T Q_0 y \\
\text{s.t} & \quad Ax + By \leq r, \\
& \quad x, y \geq 0.
\end{align*}$$

Where $r$ is a random vector by changing it, the optimal solution changes too.

Step 2: Keeping the present best chromosome in an array.

The best chromosome is kept in the array at the each iteration. This process continues till the algorithm is finished, then the best chromosome is found in the array as the optimal solution.

Step 3: Crossover operation

Crossover is a major operation to compose a new generation. In this stage two chromosomes are selected randomly and they are combined to generate a new chromosome. In the new generation components are created by the following rules:
1. The i-th component of the first child is replaced by the sum of the i-th components of parents \((i=1,2,\ldots,m)\). The operation sum is defined as follows:

<table>
<thead>
<tr>
<th></th>
<th>1+0=1</th>
<th>0+1=1</th>
<th>0+0=0</th>
<th>1+1=0</th>
</tr>
</thead>
</table>

The other components are remained the same as the first parent.

2. The \((m+i)\)-th component of the second child is replaced by the sum of the \((m+i)\)-th components of parents \((i = 1,2,\ldots,n_z)\). The operation sum is defined as above. The other components are remained the same as the second parent.

For example, by applying the present method to the following parents, and \(m=5, n_z=4\) we generate the following children:

<table>
<thead>
<tr>
<th>Parents:</th>
<th>Children:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10110 1001</td>
<td>01100 1001</td>
</tr>
<tr>
<td>11010 0111</td>
<td>11010 1110</td>
</tr>
</tbody>
</table>

**Step 4: Mutation**

The main goal of mutation in GA is to avoid trapping in local optimal solutions. In this algorithm each chosen gene of every chromosome, mutates as follows:

If the value of the chosen gene be 0, it will be changed to 1 and if the value of the chosen gene be 1, it will be changed to 0.

**Step 5: Selection**

The chromosomes of the current population are arranged in descending order of fitness values. Then we select a new population similar to the size of the first
generation. If the number of the generations is sufficient we go to the next step, otherwise the algorithm is continued by the step3.

**Step 6: Termination**

The algorithm is terminated after a maximum generation number. The best produced solution that has been recorded in the algorithm is reported as the best solution to BLPP by proposed GA algorithm.

**Computational results**

Two following examples are solved by use of the genetic algorithm proposed in this article to illustrate the feasibility and efficiency of the proposed algorithm. The first example is LQBP and the second example is LFBP.

**Example 1:**

Consider the following linear quadratic bi-level programming problem [16].

\[
\begin{aligned}
    \max & \quad x_1 + x_2 + 3y_1 - y_2 \\
    \text{s.t} & \quad 5y_1 + 8y_2 + (x_1, x_2, y_1, y_2) \begin{pmatrix} 1 & 3 & 2 & 0 \\ 3 & 1 & 4 & -2 \\ 2 & 4 & -2 & 1 \\ 0 & -2 & 1 & 5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \end{pmatrix} \\
    \text{s.t} & \quad x_1 + x_2 + y_1 + y_2 \leq 12, \\
    & \quad -x_1 + x_2 \leq 2, \\
    & \quad 3x_1 - 4y_2 \leq 5, \\
    & \quad y_1 + y_2 \leq 4, \\
    & \quad x_1, x_2, y_1, y_2 \geq 0.
\end{aligned}
\]
Using KKT conditions following problem is obtained:

\[
\begin{align*}
\max_{x} & \quad x_1 + x_2 + 3y_1 - y_2 \\
\text{s.t.} & \quad x_1 + x_2 + y_1 + y_2 + w_1 = 12, \\
& \quad -x_1 + x_2 + w_2 = 2, \\
& \quad 3x_1 - 4y_2 + w_3 = 5, \\
& \quad y_1 + y_2 + w_4 = 4, \\
& \quad 4x_1 + 8x_2 - 4y_1 + 2y_2 - u_1 - u_4 = -5 \\
& \quad -4x_2 + 10y_2 - u_1 + 4u_3 - u_4 = -8 \\
& \quad u_1w_1 + u_2w_2 + u_3w_3 + u_4w_4 = 0 \\
& \quad v_1y_1 + v_2y_2 = 0 \\
& \quad x_1, x_2, y_1, y_2, v_1, v_2, u_i, w_i \geq 0, i = 1,2,3,4.
\end{align*}
\]

It is easy to show that by relaxing the u’s and v variables (by fixing them on zero or one) in the main problem, we can obtain upper bounds for the problem which might be not promising as expected. By enumeration of possible relaxation the best upper bound is shown in Table 1.

\[
w_1 = w_2 = w_3 = w_4 = 0, \quad v_1 = v_2 = 0 \quad \Rightarrow \quad z^* = 23
\]

In genetic algorithm the initial population is created according to the proposed rules in section 4. Also the best solution is produced by the following chromosome:

\[
001100
\]

According to this chromosome and above rules, we have the following results:

\[
u_1 = u_2 = 0, \quad w_3 = w_4 = 0, \quad v_1 = v_2 = 0.
\]
Eghbal Hosseini, Isa Nakhai Kamalabadi

<table>
<thead>
<tr>
<th>Best solution by our method</th>
<th>Optimal solution by references [16, 20]</th>
<th>Relaxation upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>((x_1^<em>, x_2^</em>, y_1^<em>, y_2^</em>))</td>
<td>(z^*)</td>
<td>((x_1^<em>, x_2^</em>, y_1^<em>, y_2^</em>))</td>
</tr>
<tr>
<td>(6.31, 1.68, 4.00, 0.00)</td>
<td>19.99</td>
<td>(6.31250, 1.68750, 4.0000, 0.0000)</td>
</tr>
</tbody>
</table>

Table 1 comparison the best solutions - Example 1

According to the Table 1, the best solution by the proposed algorithm equals to the optimal solution exactly. It can be seen that the proposed method is efficient and feasible from the results.

Example 2:

The following problem is linear fractional bi-level programming problem [13].

\[
\begin{align*}
\max_{x} & \quad \frac{5 - 2x - y}{2 + x + y} \\
\text{s.t} & \quad \max_{y} y \\
\text{s.t} & \quad -5x - 3y \leq -15, \\
& \quad -x + 4y \leq 28, \\
& \quad 2x + 3y \leq 32, \\
& \quad 2x + 2y \leq 26, \\
& \quad 2x - y \leq 13, \\
& \quad x - 4y \leq 3, \\
& \quad x, y \geq 0.
\end{align*}
\]

Applying KKT conditions the above problem convert to this problem:
A Genetic Approach for Solving Bi-Level Programming Problems

By enumeration of possible relaxation, the best upper bound is

\[
\max_x \quad \frac{5 - 2x - y}{2 + x + y} \\
\text{s.t.} \quad -5x - 3y + w_1 = -15, \\
- x + 4y + w_2 = 28, \\
2x + 3y + w_3 = 32, \\
2x + 2y + w_4 = 26, \\
2x - y + w_5 = 13, \\
x - 4y + w_6 = 3, \\
-3u_1 + 4u_2 + 3u_3 + 2u_4 - u_5 - 4u_6 - v = -3, \\
w_iu_1 + w_2u_2 + w_3u_3 + w_4u_4 + w_5u_5 + w_6u_6 = 0, \\
yv = 0, \\
x, y, v, w_i, u_i \geq 0, \quad i = 1,\ldots, 6.
\]

By enumeration of possible relaxation, the best upper bound is

\[
w_1 = w_2 = w_3 = w_4 = 0, \quad u_5 = u_6 = 0, \quad v = 0 \quad \Rightarrow \quad z^* = 1.95
\]

In genetic algorithm the initial population is created according to the proposed rules in section 4. Also the best solution is produced by the following chromosome:

0011100

Choosing \( u_1 = u_2 = u_4 = 0, \quad w_3 = w_4 = w_5 = 0 \), \( v = 0 \), by the proposed genetic algorithm, the optimal solution is obtained. The best solution \((x^*, y^*) = (8.66, 4.33)\) is and the upper level’s objective function is 1.66 also the lower level’s objective function is 8.66. The results are all close to the exact values in Ref [12, 21]. It is easy to see that the proposed genetic algorithm is feasible according to the results.

6. Conclusion

We presented a genetic method for solving linear-quadratic bi-level programming and linear-fractional bi-level programming problems. Using the KKT conditions LQBP and LFBP are converted into single level problems. Then the problems are made simpler to linear programming by the chromosome according to the rule.
Finally the algorithm presents the best solution better than references. In fact the proposed algorithm is a novel assay for solving two important forms of bi-level programming problem.

References


