2013 IEEE Congress on Evolutionary Computation
June 20-23, Cancún, México

Differential Evolution for Bilevel Programming

Jaqueline S. Angelo*, Eduardo Krempser*†, Helio J.C. Barbosa*‡
*Laboratório Nacional de Computação Científica, Petrópolis - RJ, Brazil
†Faculdade de Educação Tecnológica do Estado do Rio de Janeiro (FAETERJ-Petrópolis)
‡Universidade Federal de Juiz de Fora, Juiz de Fora, MG, Brazil
Email: {jsangelo, krempser, hcbm}@lncc.br

***Abstract*—Bilevel programming problems are characterized by two optimization problems which are hierarchically related, where to each feasible upper level solution an optimal solution in the lower level problem must be associated. These problems appear in many practical applications, and a variety of solution techniques can be found in the literature. In this paper, an algorithm is proposed which uses differential evolution to solve both the upper and lower level optimization problems. Several test problems from the literature are solved in order to assess the performance of the proposed method.**

I. Introduction

The Bilevel programming problem (BLP) is characterized as an optimization problem that has in its constraints another optimization problem. In this hierarchical structure each level of the hierarchy seeks to optimize an objective function, controlling a set of variables subject to certain constraints. One can think of a leader (upper level) that optimizes an objective function considering both his constraints and the reaction of the follower (lower level), which in turn makes his decision to optimize its objective function taking into account what has been imposed by the leader.

Due to this hierarchical structure, even bilevel problems where all functions involved are linear are, in general, quite difficult to be solved, and were shown to be NP-hard by Hansen et. al [1]. A variety of exact methods have been developed, involving the use of enumerative schemes, application of a penalty function, or gradient methods, among others. The most used exact method investigate the properties of the so-called inducible region to obtain (necessary and sufficient) conditions to replace the follower's problem with its Karush-Kunh-Tucker (KKT) conditions, appending the resultant system to the leader's problem, transforming the bilevel problem into a standard single-level programming problem. However, those approaches assume (explicit) knowledge of the analytical form of the objective function and the constraints of the problem and, sometimes, the transformed problem is not equivalent to the original bilevel problem.

The investigations in multilevel programming problems are strongly motivated by real-world applications found in economics, engineering, transportation, networks, planning, etc [2]. Due to the complexity of those applications, intelligent heuristics such as evolutionary computation, become attractive and even enable the resolution of such problems. Although these techniques cannot guarantee to find the optimal solution, but only a good approximation to it, they can help overcome the challenges of bilevel programming problems, such as nonconvexity and nondifferentiability, large number of variables and/or constraints, and non-unique optimal solution for the follower's problem. Heuristic methods, can also deal with cases where it is not possible to make strong assumptions about the objective function and/or constraints of the problem. Those cases, known as "black box" optimization, occur often in practice, for example, when the computation of the objective functions and/or constraints require a potentially expensive computer simulation.

Differential Evolution (DE) is an evolutionary algorithm, proposed by Storn and Price (1997), aimed at solving continuous optimization problems. It is a simple, stochastic, population-based technique that was initially formulated to deal with unconstrained problems, but which can be efficiently applied to constrained problems [3,4]. In this paper, a DE algorithm (BlDE) for solving general BLPs is proposed, and its performance is assessed using a collection of well known test-problems from the literature.

The paper is organized as follows: Section II describes the structure of a general bilevel optimization problem; Section III presents a brief literature review on related work emphasizing some evolutionary methods already proposed. The DE technique is summarized in Section IV, while in Section V the proposed solution method BlDE is presented. The description of each test problem and the results of the computational experiments are presented in Section VI; the conclusions are left for Section VII.

II. The Bilevel Programming Problem

The bilevel programming problem was initially proposed by the economist Von Stackelberg in 1934 to describe oligopoly situations in economy markets. The Stackelberg's model describes a competitive game between two companies that make decisions sequentially.

In the BLP, two decision makers, the leader in the upper level and the follower in the lower level, are hierarchically related, where the leader's decisions affect both the follower's payoff function and allowable actions, and vice-versa. Each decision maker has control over a set of variables, seeking to optimize its own objective function. Once the leader chooses the value of its variables, the follower reacts by choosing the value of its own variables in order to optimize its objective function.

The bilevel programming problem can be written as:

$$\begin{array}{rll} (L) & \min_{x\in X} & f_1(x, y(x)) \\ & \text{subject to} & g_1(x, y(x)) \leq 0 \\ & (F) & y(x) \in \arg\min_{y\in Y} f_2(x, y) \\ & & \quad \text{subject to} \quad g_2(x, y) \leq 0 \end{array} \tag{1}$$

The higher level decision maker (L) –the leader– has control over the x variables, and makes his decision first, fixing x. The lower level decision maker (F) –the follower– controls the y variables. Reacting to the decision of the leader, the y variables are set in response to the given x.

In the experiments performed here, it is assumed that $X = R^{n_1}$ and $Y = R^{n_2}$, n_1 and n_2 being the number of upper and lower level variables, respectively.

In problem (1) the reaction set of the follower, namely $R(x)$ such that $y(x) \in R(x)$, defines the follower's response given an x fixed by the leader. One difficulty that arises in solving BLP is that, if $R(x)$ is not single-valued for all possible x, the leader may not achieve his minimum payoff, since the follower has multiple minimum solutions to choose. In this case, there is no guarantee that the follower's choice is the best for the leader, leading to sub-optimal solutions in the leader's problem.

Another challenge lies in the fact that unless a solution is optimal for the lower level problem, it cannot be feasible for the overall problem. This suggests that approximate optimization methods cannot be used to solve the lower level problem, as they do not guarantee that the optimal solution is actually found. However, from the practical point of view near-optimal solutions are often acceptable [5], especially when the lower level problem is too costly to be exactly solved thus rendering the use of exact methods impractical.

III. Literature Review

There have been numerous methods proposed for solving bilevel programming problems. Most of them are specialized to the linear case, where all functions involved are linear. In general, for solving the BLP one can find three different approaches: enumerative schemes, mainly for the linear BLP [6]–[9], penalty methods [10]–[13], and methods based on the KKT conditions [1,14]–[17]. All of these methods are very time consuming, especially when dealing with large sized problems. For this reason, the use of intelligent heuristics is growing, as they are capable of finding good solutions for complex single-level optimization problems in reasonable computer times.

Mathieu et al. [18] firstly developed a genetic algorithm (GA) to solve linear BLP. In their proposal, the leader's decision variables are generated using a modified GA, which only used the mutation operation, while the follower's decision variables are obtained using an algorithm based on the simplex method. The algorithm proposed presented better results when compared to Bard's grid search technique [19] in randomly generated test-problems with different sizes. However, the time consumed was higher, varying between 24.82s and 3281.59s against 4.23 and 63.99 provided by the grid search technique. The same happens with a simulated annealing algorithm (SA) when applied to the same test-problems [20]. SA presented better results than the grid search technique–however, worse than those produced by the GA–in a higher computational time.

Hejazi et al. [21] proposed a GA for the linear BLP. The second level problem was replaced by the KKT optimality conditions yielding a single-level optimization problem which was solved by the GA. The proposed method was compared to a hybrid tabu-ascent algorithm [22] in randomly generated problems of different sizes. The proposed GA was faster than the tabu-ascent algorithm in all test problems, except for one. However, for the largest problems, the quality of the solutions obtained by the tabu-ascent algorithm was superior to that achieved by the GA proposed.

Oduguwa and Roy [23] proposed a coevolutionary GA for solving the general BLP. The proposed method implemented two populations, one to solve the leader's problem, and another to solve the follower's problem. A coevolutionary operator synchronizes one population with the fittest members of the other population. Hence, the leader's variables x are optimized with a subset of the best follower's variables, and the follower's variables y are optimized with all x variables fixed. An additional elite population was maintained to identify the elite members from both populations, and was updated in every generation with the best members from the current generation. The algorithm proposed presented similar solutions when compared to different methods for solving four test problems available in literature [24].

In [25], Zhu et. al proposed a DE algorithm combined with an interior point method for solving nonlinear problems with linear constraints. The DE algorithm was designed to operate in the leader's problem while the interior point method was responsible for solving the follower's problem. The DE was used to generate and modify the leader's variables. For each individual with a fixed x, y is obtained by solving the follower's problem using the interior point method. The algorithm was capable of finding the same results reported in the original references for the six problems tested.

The DE was also used by Koh [26] to solve a transportation BLP, where a gradient based algorithm was used to solve the lower level problem. The algorithm is similar to the one proposed in [25]: the DE operates in the upper level, evolving x, while the gradient based method generates the optimal y for each x fixed by the upper level problem. Seven BLPs taken from the literature, as well as specific applications in transportation problems, were used to assess the quality of the algorithm proposed. The algorithm was capable of finding solutions identical to the ones reported in the original references for most test problem, and provided better solutions in two of them. For the transportation problems tested the algorithm generated competitive results.

IV. Differential Evolution

DE was originally proposed by Storm and Price [27] and has been shown to be a simple and effective algorithm for global optimization, specially for continuous variables.

The basic operation performed is the addition to each design variable in a given candidate solution of a term which is the scaled difference between the values of such variable in other candidate solutions in the population. The number of differences applied, the way in which the individuals are selected, and the distribution of recombination determine the DE variant. As with other evolutionary algorithms, DE performance is influenced by its particular parameters, namely, the chosen variant, and the scale factor (F) adopted.

Here, the DE variant DE/target-to-rand/1/bin was adopted. In preliminary experiments this strategy presented the best results to the overall test problems considered in this paper. This variant uses randomly selected individuals (r_1, r_2, r_3) of the population and the target individual x_i (the one that will be used in the comparison after the mutation, also called current individual), leading to:

$$u_{i,j,G+1} = x_{i,j,G} + F.(x_{r_3,j,G} - x_{i,j,G}) + F.(x_{r_1,j,G} - x_{r_2,j,G})$$

In addition, a crossover operation is performed, using the parameter CR.

For each design variable, lower and upper bounds are assumed, and whenever a given component $x_{i,j}$ of a candidate solution x_i is generated outside its prescribed range, a standard projection operation is performed:

If $x_{i,j} > x_j^U$ then $x_{i,j} = x_j^U$ and if $x_{i,j} < x_j^L$ then $x_{i,j} = x_j^L$.

V. The Proposed Solution Method

In the proposed Bilevel Differential Evolution algorithm (BlDE), the lower level problem is solved for each upper level point. The solution method uses a DE to generate and evolve the upper level variables. For each upper level point, another DE is used to solve the lower level problem, by generating and evolving the lower level variables. Algorithms 1 and 2 describe the proposed method.

Algorithm 1: Algorithm DE Leader.

```
input : np (population size), gen (# of generations),
        F (mutation scaling), CR (crossover rate)
1  G ← 0;
2  CreateRandomInitialPopulation(np);
3  for $i \leftarrow 1$ to np do
4      $\vec{y}$ = DEFollower(npf, genf, F, CR, $\vec{x}_{i,G}$);
5      Evaluate $f_1$($\vec{x}_{i,G}$, $\vec{y}$) ;   /* $\vec{x}_{i,G}$ is an individual in the population */
6  for G ← 1 to gen do
7      for $i \leftarrow 1$ to np do
8          SelectRandomly($r_1, r_2, r_3$) ;
           /* $r_1 \neq r_2 \neq r_3 \neq i$ */
9          $jRand$ ←RandInt(1, n) ;   /* n is the number of variables of the leader problem */
10         for $j \leftarrow 1$ to n do
11             if Rand(0,1) < CR or $j = jRand$ then
12                 $u_{i,j,G+1} = x_{i,j,G} + F.(x_{r_3,j,G} - x_{i,j,G})+$
13                         $F.(x_{r_1,j,G} - x_{r_2,j,G})$;
14             else
15                 $u_{i,j,G+1} = x_{i,j,G}$;
16         $\vec{y}$ = DEFollower(npf, genf, F, CR, $\vec{u}_{i,G+1}$);
17         if $f_1(\vec{u}_{i,G+1}, \vec{y}) \leq f_1(\vec{x}_{i,G}, \vec{y})$ then
18             $\vec{x}_{i,G+1} = \vec{u}_{i,G+1}$;
19         else
20             $\vec{x}_{i,G+1} = \vec{x}_{i,G}$;
```

The main steps of the algorithm are summarized as follows:

Algorithm 2: Algorithm DE Follower.

```
input : npf (follower population size), genf (# of
        generations to follower DE), F (mutation
        scaling), CR (crossover rate), $\vec{V}$ (leader
        variables)
1  G ← 0;
2  CreateRandomInitialPopulation(npf);
3  for $i \leftarrow 1$ to npf do
4      Evaluate $f_2$($\vec{V}$, $\vec{x}_{i,G}$) ;   /* $\vec{x}_{i,G}$ is an individual in the population */
5  for G ← 1 to genf do
6      for $i \leftarrow 1$ to npf do
7          SelectRandomly($r_1, r_2, r_3$) ;
           /* $r_1 \neq r_2 \neq r_3 \neq i$ */
8          $jRand$ ←RandInt(1, nf) ;   /* nf is the number of variables in the follower problem */
9          for $j \leftarrow 1$ to nf do
10             if Rand(0,1) < CR or $j = jRand$ then
11                 $u_{i,j,G+1} = x_{i,j,G} + F.(x_{r_3,j,G} - x_{i,j,G})+$
12                         $F.(x_{r_1,j,G} - x_{r_2,j,G})$;
13             else
14                 $u_{i,j,G+1} = x_{i,j,G}$;
15         if $f_2(\vec{V}, \vec{u}_{i,G+1}) \leq f_2(\vec{V}, \vec{x}_{i,G})$ then
16             $\vec{x}_{i,G+1} = \vec{u}_{i,G+1}$;
17         else
18             $\vec{x}_{i,G+1} = \vec{x}_{i,G}$;
19 return SelectBestIndividual
```

Step 0: Initialization. The algorithm starts with a population, of size np, of vectors containing the upper level variables $x \in R^{n_1}$. The upper level variables are initialized with random values and the lower level variables are determined by executing the lower level procedure (Algorithm 2), which generates the vector $y \in R^{n_2}$ of lower level variables.

Step 1: Evolution at the upper level. Following the basic DE algorithm described in Algorithm 1, the upper level individuals are mutated and recombined.

Step 2: Evaluation of each upper level individual. To evaluate the individuals in the upper level, where fitness is assigned based on the upper level function and constraints, the lower level procedure is performed. The solution returned, that is, the best individual obtained in the lower level procedure, is used to evaluate the upper level individual.

Step 3: Evolution at lower level. For fixed upper level variables, a new DE algorithm is executed, as described in Algorithm 2. The individuals are evaluated based on the lower level function and constraints. Finally, the procedure returns the best value of the lower level problem.

In order to handle the constraints, the method proposed by Deb in [28] was applied, which enforces the following criteria: (i) any feasible solution is preferred to any infeasible solution, (ii) among two feasible solutions, the one having better objective function value is preferred, and (iii) among two infeasible solutions, the one having smaller constraint violation is preferred.

VI. Computational Experiments

In this section, eighteen test-problems taken from different sources in the literature are used in order to assess the performance of the proposed algorithm. The BlDE algorithm was also tested in a set of six unconstrained test problems proposed in [29]. The test-collection (SMD1 to SMD6) in [29] aims to induce difficulties at both levels of the BLP, independently and collectively, such that the performance of the algorithms could be better evaluated when handling the two levels.

For the BlDE algorithm, parameters F and CR were set to $F = 0.7$ and $CR = 0.9$. For all test-problems, the population size was set to 30 individuals and 30 independent runs were performed.

A. *Test problems*

The eighteen test problems initially considered are:

1) Shimizu and Aiyoshi (1981) [10]:

$$\begin{array}{ll} \min_x f_1(x,y) = x^2 + (y-10)^2 & \\ \text{s.t.} \quad -x + y \le 0 & \\ \quad x \in [0,15] & \\ \quad \min_y f_2(x,y) = (x + 2y - 30)^2 & \\ \quad \text{s.t.} \quad x + y - 20 \le 0 & \\ \quad\quad y \in [0,20] & \end{array}$$

2) Shimizu and Aiyoshi (1981) [10]:

$$\begin{array}{l} \min_x f_1(x,y) = (x_1-30)^2 + (x_2-20)^2 - 20y_1 + 20y_2 \\ \text{s.t.} \quad x_1 + 2x_2 \ge 30 \\ \quad x_1 + x_2 \le 25 \\ \quad x_2 \le 15 \\ \quad \min_y (x_1 - y_1)^2 + (x_2 - y_2)^2 \\ \quad \text{s.t.} \quad y_1, y_2 \in [0,20] \end{array}$$

3) Candler and Townsley (1982) [6]:

$$\begin{array}{l} \max_x f_1(x,y) = 8x_1 + 4x_2 - 4y_1 + 40y_2 + 4y_3 \\ \text{s.t.} \quad \max_y f_2(x,y) = -x_1 - 2x_2 - y_1 - y_2 - 2y_3 \\ \quad \text{s.t} \quad -y_1 + y_2 + y_3 \le 1 \\ \quad\quad 2x_1 - y_1 + 2y_2 - 0.5y_3 \le 1 \\ \quad\quad 2x_2 + 2y_1 - y_2 - 0.5y_3 \le 1 \\ \quad\quad x, y \ge 0 \end{array}$$

4) Bard and Falk (1982) [30]:

$$\begin{array}{l} \max_x f_1(x,y) = 2x_1 - x_2 - 0.5y_1 \\ \text{s.t.} \quad \max_y f_2(x,y) = -x_1 - x_2 + 4y_1 - y_2 \\ \quad \text{s.t.} \quad 2x_1 - y_1 + y_2 \ge 2.5 \\ \quad\quad -x_1 + 3x_2 - y_2 \ge -2 \\ \quad\quad -x_1 - x_2 \ge -2 \\ \quad\quad x, y \ge 0 \end{array}$$

5) Aiyoshi and Shimizu (1984) [31]:

$$\begin{array}{l} \min_x f_1(x,y) = 2x_1 + 2x_2 - 3y_1 - 3y_2 - 60 \\ \text{s.t.} \quad x_1 + x_2 + y_1 - 2y_2 - 40 \le 0 \\ \quad \min_y f_2(x,y) = (y_1 - x_1 + 20)^2 + (y_2 - x_2 + 20)^2 \\ \quad \text{s.t.} \quad 2y_1 - x_1 + 10 \le 0 \\ \quad\quad 2y_2 - x_2 + 10 \le 0 \\ \quad\quad x \in [0,50], y \in [-10,20] \end{array}$$

6) Bard (1988) [32]:

$$\begin{array}{l} \min_x f_1(x,y) = (x-5)^2 + (2y+1)^2 \\ \text{s.t.} \quad \min_y f_2(x,y) = (y-1)^2 - 1.5xy \\ \quad \text{s.t.} \quad 3x - y \ge 3 \\ \quad\quad -x + 0.5y \ge -4 \\ \quad\quad -x - y \ge -7 \\ \quad\quad x, y \ge 0 \end{array}$$

7) Bard (1988) [32]:

$$\begin{array}{l} \min_x f_1(x,y) = -x_1^2 - 3x_2 - 4y_1 + y_2^2 \\ \text{s.t.} \quad x_1^2 + 2x_2 \le 4 \\ \quad \min_y f_2(x,y) = 2x_1^2 + y_1^2 - 5y_2 \\ \quad \text{s.t.} \quad x_1^2 - 2x_1 + x_2^2 - 2y_1 + y_2 \ge 3 \\ \quad\quad x_2 + 3y_1 - 4y_2 \ge 4 \\ \quad\quad x, y \ge 0 \end{array}$$

8) Anandalingam and White (1990) [11]:

$$\begin{array}{l} \max_x f_1(x,y) = x + 3y \\ \text{s.t.} \quad \max_y f_2(x,y) = x - 3y \\ \quad \text{s.t.} \quad -x - 2y \le -10 \\ \quad\quad x - 2y \le 6 \\ \quad\quad 2x - y \le 21 \\ \quad\quad x + 2y \le 38 \\ \quad\quad -x + 2y \le 18 \\ \quad\quad x, y \ge 0 \end{array}$$

9) Savard and Gauvin (1994) [33]:

$$\begin{array}{l} \min_x f_1(x,y) = (x-1)^2 + 2y_1^2 - 2x \\ \text{s.t.} \quad \min_y f_2(x,y) = (2y_1 - 4)^2 + (2y_2 - 1)^2 + xy_1 \\ \quad \text{s.t.} \quad 4x + 5y_1 + 4y_2 \le 12 \\ \quad\quad -4x - 5y_1 + 4y_2 \le -4 \\ \quad\quad 4x - 4y_1 + 5y_2 \le 4 \\ \quad\quad -4x + 4y_1 + 5y_2 \le 4 \\ \quad\quad x, y \ge 0 \end{array}$$

10) Falk and Liu (1995) [34]:

$$\begin{array}{l} \min_x f_1(x,y) = x_1^2 - 2x_1 + x_2^2 - 2x_2 + y_1^2 + y_2^2 \\ \text{s.t.} \quad \min_y f_2(x,y) = (y_1 - x_1)^2 + (y_2 - x_2)^2 \\ \quad \text{s.t.} \quad x \ge 0, y \in [0.5, 1.5] \end{array}$$

11) Shimizu and Lu (1995) [35]

$$\begin{array}{l} \min_x f_1(x,y) = 16x^2 + 9y^2 \\ \text{s.t.} \quad -4x + y \le 0 \\ \quad \min_y f_2(x,y) = (x + y - 10)^4 \\ \quad \text{s.t.} \quad 4x + y - 50 \le 0 \\ \quad\quad x, y \ge 0 \end{array}$$

12) Bard (1998) [24]:

$$\begin{array}{l} \min_x f_1(x,y) = x - 4y \\ \text{s.t.} \quad \min_y f_2(y) = y \\ \quad \text{s.t.} \quad -x - y \le -3 \\ \quad\quad -2x + y \le 0 \\ \quad\quad 2x + y \le 12 \\ \quad\quad -3x + 2y \ge -4 \\ \quad\quad x, y \ge 0 \end{array}$$

13) Bard (1998) [24]:

$$\begin{array}{l} \min_x f_1(x,y) = x + y \\ \text{s.t.} \quad \min_y f_2(x,y) = -5x - y \\ \quad \text{s.t.} \quad -x - y/2 \le -2 \\ \quad\quad -x/4 + y \le 2 \\ \quad\quad x + y/2 \le 8 \\ \quad\quad x - 2y \le 4 \\ \quad\quad x, y \ge 0 \end{array}$$

14) Bard (1998) [24]:

$$\begin{array}{l} \min_x f_1(x,y) = (x-1)^2 + (y-1)^2 \\ \text{s.t.} \quad \min_y f_2(x,y) = 0.5y^2 + 500y - 50xy \\ \quad x, y \ge 0 \end{array}$$

15) Oduguwa and Roy (2002) [23]:

$$\begin{array}{l} \max_x f_1(x,y) = 100x + 1000y_1 \\ \text{s.t.} \quad \max_y f_2(x,y) = y_1 + y_2 \\ \quad \text{s.t.} \quad x + y_1 - y_2 \le 1 \\ \quad\quad y_1 + y_2 \le 1 \\ \quad\quad x \in [0,1], y \in [0,1] \end{array}$$

16) Rajesh et. al (2003) [36]:

$$\begin{aligned}&\min_x f_1(x,y) = (x-3)^2 + (y-2)^2\\ &\text{s.t.}\quad \min_y f_2(x,y) = (y-5)^2\\ &\qquad\text{s.t.}\quad 2x - y \geq -1\\ &\qquad\qquad -x + 2y \geq 2\\ &\qquad\qquad -x - 2y \geq -14\\ &\qquad\qquad x \in [0,8], y \geq 0\end{aligned}$$

17) Rajesh et. al (2003) [36]:

$$\begin{aligned}&\min_x f_1(x,y) = (x-3)^2 + (y-2)^2\\ &\text{s.t.}\quad 2x - y \geq -1\\ &\qquad -x + 2y \geq 2\\ &\qquad -x - 2y \geq -14\\ &\qquad\text{s.t.}\quad \min_y f_2(x,y) = (y-5)^2\\ &\qquad\qquad x \in [0,8], y \geq 0\end{aligned}$$

18) Rajesh et. al (2003) [36]:

$$\begin{aligned}&\max_x f_1(x,y) = -2x + 11y\\ &\text{s.t.}\quad \max_y f_2(x,y) = -x - 3y\\ &\qquad\text{s.t.}\quad x - 2y \leq 4\\ &\qquad\qquad 2x - y \leq 24\\ &\qquad\qquad 3x + 4y \leq 96\\ &\qquad\qquad x + 4y \leq 126\\ &\qquad\qquad -4x + 5y \leq 65\\ &\qquad\qquad x + 4y \geq 8\\ &\qquad\qquad x, y \leq 0\end{aligned}$$

The second part of the experiments consists in solving the test-collection (SMD1 to SMD6) proposed in [29]. For those tests, the upper and the lower level functions $F(x,y)$ and $f(x,y)$, respectively, are split into three components, as

$$F(x,y) = F1(x_1) + F2(y_1) + F3(x_2,y_2) \quad (2)$$

$$f(x,y) = f1(x_1,x_2) + f2(y_1) + f3(x_2,y_2) \quad (3)$$

where $x = (x_1, x_2)$ and $y = (y_1, y_2)$. The dimensions of the variables are set to $dim(x_1) = p$, $dim(x_2) = r$, $dim(y_1) = q$ and $dim(y_2) = r$. For SMD6, $dim(y_1) = q + s$.

The test problems are described as:

SMD1:
$$\begin{aligned}F1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ F2 &= \textstyle\sum_{i=1}^{q}(y_1^i)^2\\ F3 &= \textstyle\sum_{i=1}^{r}(x_2^i)^2 + \sum_{i=1}^{r}(x_2^i - \tan y_2^i)^2\\ f1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ f2 &= \textstyle\sum_{i=1}^{q}(y_1^i)^2\\ f3 &= \textstyle\sum_{i=1}^{r}(x_2^i - \tan y_2^i)^2\end{aligned}$$

The ranges of the variables are:
$x_1^i \in [-5,10], \forall i \in \{1,2,...,p\}$
$x_2^i \in [-5,10], \forall i \in \{1,2,...,r\}$
$y_1^i \in [-5,10], \forall i \in \{1,2,...,q\}$
$y_2^i \in (\frac{-\pi}{2}, \frac{\pi}{2}), \forall i \in \{1,2,...,r\}$

The optimal values are $x = 0$ and y given by: $y_1^i = 0, \forall i \in \{1,2,...,q\}$ and $y_2^i = \tan^{-1} x_2^i, \forall i \in \{1,2,...,r\}$. The upper and lower level functions are equal to 0 at the optimum.

SMD2:
$$\begin{aligned}F1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ F2 &= -\textstyle\sum_{i=1}^{q}(y_1^i)^2\\ F3 &= \textstyle\sum_{i=1}^{r}(x_2^i)^2 - \sum_{i=1}^{r}(x_2^i - \log y_2^i)^2\\ f1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ f2 &= \textstyle\sum_{i=1}^{q}(y_1^i)^2\\ f3 &= \textstyle\sum_{i=1}^{r}(x_2^i - \log y_2^i)^2\end{aligned}$$

The ranges of the variables are:
$x_1^i \in [-5,10], \forall i \in \{1,2,...,p\}$
$x_2^i \in [-5,1], \forall i \in \{1,2,...,r\}$
$y_1^i \in [-5,10], \forall i \in \{1,2,...,q\}$
$y_2^i \in (0,e], \forall i \in \{1,2,...,r\}$

The optimal values are $x = 0$ and y given by: $y_1^i = 0, \forall i \in \{1,2,...,q\}$ and $y_2^i = \log^{-1} x_2^i, \forall i \in \{1,2,...,r\}$. The upper and lower level functions are equal to 0 at the optimum.

SMD3:
$$\begin{aligned}F1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ F2 &= \textstyle\sum_{i=1}^{q}(y_1^i)^2\\ F3 &= \textstyle\sum_{i=1}^{r}(x_2^i)^2 + \sum_{i=1}^{r}((x_2^i)^2 - \tan y_2^i)^2\\ f1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ f2 &= q + \textstyle\sum_{i=1}^{q}((y_1^i)^2 - \cos 2\pi y_1^i)\\ f3 &= \textstyle\sum_{i=1}^{r}((x_2^i)^2 - \tan y_2^i)^2\end{aligned}$$

The ranges of the variables are:
$x_1^i \in [-5,10], \forall i \in \{1,2,...,p\}$
$x_2^i \in [-5,10], \forall i \in \{1,2,...,r\}$
$y_1^i \in [-5,10], \forall i \in \{1,2,...,q\}$
$y_2^i \in (\frac{-\pi}{2}, \frac{\pi}{2}), \forall i \in \{1,2,...,r\}$

The optimal values are $x = 0$ and y given by: $y_1^i = 0, \forall i \in \{1,2,...,q\}$ and $y_2^i = \tan^{-1} x_2^i, \forall i \in \{1,2,...,r\}$. The upper and lower level functions are equal to 0 at the optimum.

SMD4:
$$\begin{aligned}F1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ F2 &= -\textstyle\sum_{i=1}^{q}(y_1^i)^2\\ F3 &= \textstyle\sum_{i=1}^{r}(x_2^i)^2 - \sum_{i=1}^{r}(|x_2^i| - \log(1 + y_2^i))^2\\ f1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ f2 &= q + \textstyle\sum_{i=1}^{q}((y_1^i)^2 - \cos 2\pi y_1^i)\\ f3 &= \textstyle\sum_{i=1}^{r}(|x_2^i| - \log(1 + y_2^i))^2\end{aligned}$$

The ranges of the variables are:
$x_1^i \in [-5,10], \forall i \in \{1,2,...,p\}$
$x_2^i \in [-1,1], \forall i \in \{1,2,...,r\}$
$y_1^i \in [-5,10], \forall i \in \{1,2,...,q\}$
$y_2^i \in [0,e], \forall i \in \{1,2,...,r\}$

The optimal values are $x = 0$ and y given by: $y_1^i = 0, \forall i \in \{1,2,...,q\}$ and $y_2^i = \log^{-1} |x_2^i| - 1, \forall i \in \{1,2,...,r\}$. The upper and lower level functions are equal to 0 at the optimum.

SMD5:
$$\begin{aligned}F1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ F2 &= -\textstyle\sum_{i=1}^{q-1}((y_1^{i+1} - (y_1^i)^2)^2 + (y_1^i - 1)^2)\\ F3 &= \textstyle\sum_{i=1}^{r}(x_2^i)^2 - \sum_{i=1}^{r}(|x_2^i| - (y_2^i)^2)^2\\ f1 &= \textstyle\sum_{i=1}^{p}(x_1^i)^2\\ f2 &= \textstyle\sum_{i=1}^{q-1}((y_1^{i+1} - (y_1^i)^2)^2 + (y_1^i - 1)^2)\\ f3 &= \textstyle\sum_{i=1}^{r}(|x_2^i| - (y_2^i)^2)^2\end{aligned}$$

The ranges of the variables are:
$x_1^i \in [-5,10], \forall i \in \{1,2,...,p\}$
$x_2^i, y_2^i \in [-5,10], \forall i \in \{1,2,...,r\}$
$y_1^i \in [-5,10], \forall i \in \{1,2,...,q\}$

The optimal values are $x = 0$ and y given by: $y_1^i = 1, \forall i \in \{1,2,...,q\}$ and $y_2^i = \sqrt{|x_2^i|}, \forall i \in \{1,2,...,r\}$. The upper and lower level functions are equal to 0 at the optimum.

SMD6: $F1 = \sum_{i=1}^{p}(x_1^i)^2$

$$F2 = -\sum_{i=1}^{q}(y_1^i)^2 + \sum_{i=q+1}^{q+s}(y_1^i)^2$$
$$F3 = \sum_{i=1}^{r}(x_2^i)^2 - \sum_{i=1}^{r}(x_2^i - y_2^i)^2$$
$$f1 = \sum_{i=1}^{p}(x_1^i)^2$$
$$f2 = \sum_{i=1}^{q}(y_1^i)^2 + \sum_{i=q+1}^{q+s-1}(y_1^{i+1} - y_1^i)^2$$
$$f3 = \sum_{i=1}^{r}(x_2^i - y_2^i)^2$$

The ranges of the variables are:
$x_1^i \in [-5, 10], \forall i \in \{1, 2, ..., p\}$
$x_2^i, y_2^i \in [-5, 10], \forall i \in \{1, 2, ..., r\}$
$y_1^i \in [-5, 10], \forall i \in \{1, 2, ..., q+s\}$

The optimal values are $x = 0$ and y given by: $y_1^i = 0, \forall i \in \{1, 2, ..., q\}$ and $y_2^i = x_2^i, \forall i \in \{1, 2, ..., r\}$. The upper and lower level functions are equal to 0 at the optimum.

For the test-collection (SMD1 to SMD6) the instances considered have 10 dimensions and correspond to setting $p = 3$, $q = 2$, and $r = 2$ for problems SMD1 to SMD5, and $p = 3$, $q = 1$, $r = 2$ and $s = 2$ for problem SMD6.

B. Results

Table I presents the results obtained from solving the eighteen test-problems, showing the best solutions (x^*, y^*), and the corresponding values f_1^* and f_2^* attained at the upper and lower levels, respectively, obtained by BlDE and other methods in different sources of the literature. Table II summarizes the statistics for BlDE in the 30 independent runs performed in the same test problems. The column headers denoted by "Min", "Median", "Mean", "Max", and "std" denote the minimum, median, average, maximum, and standard deviation values. The number of function evaluations for the upper and lower levels were 6000 and 18000000, respectively.

For those test-problems the proposed method was capable of reaching the optimal solution, or very close to the optimal, in all problems tested, and found better solutions for problem 4. As demonstrated in Table II the algorithm presented a stable behavior, since for most problems tested the algorithm was able to reach the optimal solution in all 30 executions.

The results for the test-problems SMD1 to SMD6 are reported in Table III, which presents the median values of function evaluations for each level, and the accuracy achieved for both BlDE and the GA adopted in the paper [29] where the problems were proposed. Table IV summarizes the results obtained by BlDE in the SMD's problems. For those tests the number of function evaluations for the upper and lower levels were 2400 and 7200000, respectively.

The results described in Table III, demonstrate that BlDE generates the optimal solution for all test problems, and reaches a better accuracy on both levels in all SMD's problems. For those test problems, the algorithm stability was also observed, as shown in Table IV by the standard deviation values very close to zero in all problems tested.

C. Discussion

It is important to mention that different DE variants as well as different parameter values for F, CR, population size, and number of generations could be adopted in the different levels of the optimization. In fact, a more efficient algorithm would

TABLE I. COMPARISON OF BEST SOLUTIONS OBTAINED BY BLDE AND OTHER METHODS. A QUESTION MARK "?" INDICATES A VALUE NOT PROVIDED IN THE REFERENCE.

Pr.	Ref	(x^*, y^*)	f_1^*	f_2^*
1	[10,31,37]	(10, 10)	100	0
	[24,26,36]	(10, 10)	100	0
	[23]	(10.03, 9.969)	100.58	0.001
	[25]	(10.01, 9.93)	100.2050	0.0169
	BlDE	(10, 10)	100	0
2	[10,25,37]	(20, 5, 10, 5)	225	100
	BlDE	(19.9999, 5.00004, 10, 4.99941)	224.988	99.9987
3	[6,24,30]	(0, 0.9, 0, 0.6, 0.4)	29.2	-3.2
	[38]	(0, 0.899, 0, 0.06, 0.394)	29.172	-3.186
	[39]	(0, 0.898, 0, 0.599, 0.399)	29.148	-3.193
	[25]	(0, 0.9, 0, 0.6, 0.3999)	29.198	-3.2
	[40]	(0, 0.8993, 0, 0.5995, 0.3981)	29.1709	-3.1805
	BlDE	(0, 0.899998, 2.15e-6, 0.600001, 0.4000001)	29.2	-3,2
4	[30,39]	(1, 0, 0.5, 1)	1.75	0
	[40]	(0.9993, 0, 0.4993, 1.0008)	1.7488	-0.0028
	BlDE	(2, 0, 1.5, 0)	3.25	4
5	[24,31,41]	(25, 30, 5, 10)	5	0
	[12,37]	(0, 0, -10, -10)	0	200
	[42]	(25.00125, 30, ?, ?)	4.999375	?
	[25]	(24.87, 29.98, 5.1, 10.31)	3.47	0.1618
	[26]	(0, 30, 10, -10)	0	100
	BlDE	(0, 0, -10, -10)	0	200
6	[24,32]	(1.778, 2.333)	42.5	-4.445
	[26,36,37]	(1, 0)	17	1
	BlDE	(1, 0)	17	1
7	[42]	(0, 2, ?, ?)	-12.67871	?
	BlDE	(0, 1.99996, 1.87549, 0.906606)	-12.6799	-1.01557
8	[11,43]	(16, 11)	49	-17
	[38]	(15.959, 10.972)	48.875	-16.957
	[39]	(15.9972, 10.9945)	48.9806	-16.9863
	[40]	(15.9966, 10.9933)	48.9764	-16.9833
	BlDE	(16, 11)	49	-17
9	[24,33]	(1.889, 0.889, 0)	-1.21	7.61
	[23]	(?, ?, ?)	3.57	2.4
	[25]	(1.87, 0.89, 0)	-1.2031	7.5927
	BlDE	(1.88889, 0.888889, 0)	-1.209876568	7.61728
10	[26,34,42]	(0.5, 0.5, 0.5, 0.5)	-1	0
	BlDE	(0.5, 0.5, 0.5, 0.5)	-1	0
11	[24,35]	(11.25, 5)	2250	197.75
	BlDE	(11.25, 5)	2250	197.754
12	[24,43]	(4, 4)	-12	4
	BlDE	(4, 4)	-12	4
13)	[24,25,36]	(0.889, 2.222)	3.111	-6.667
	BlDE	(0.888889, 2.22222)	3.11111	-6.66667
14	[24]	(1, 0)	1	0
	[23]	(10.04, 0.1429)	82.44	-0.271
	[44]	(?, ?)	81.33	0.34
	BlDE	(1, 0)	1	0
15	[23]	(?, ?, ?)	1000	1
	[41]	(1, 0, 1)	1000	1
	BlDE	(0, 0.999657, 0.000342909)	999.657	1
16	[36]	(1, 3)	5	4
	[26]	(4, 3)	2	4
	BlDE	(1, 3)	5	4
17	[26,36]	(3, 5)	9	0
	BlDE	(3, 5)	9	0
18	[36]	(17.4545, 10.90909)	85.0909	?
	BlDE	(17.4545, 10.9091)	85.0909	-50.1818

probably arise by adapting the values of the DE parameters as well as by closely monitoring the convergence process in both populations in order to save function evaluations by stopping the iterations earlier.

Furthermore, since the algorithmic idea proposed used two DE algorithms independently, other evolutionary methods could be incorporated in different levels of the BLP, e.g., by using DE on the upper level and an ant colony algorithm on the lower level, or vice-versa. The implementation of different algorithms in different levels of the BLP could be interesting when, for instance, the lower level problem can be more

TABLE II. SUMMARY OF RESULTS OBTAINED BY BLDE FOR TEST-PROBLEMS 1 TO 18.

Pr.	Level	Min	Median	Mean	Max	std
1	UL	100	100	100	100	0
	LL	0	0	0	0	0
2	UL	224.9880	224.9920	224.9919	224.9940	1.6e-03
	LL	99.9808	99.9976	99.9967	99.9994	3.4e-03
3	UL	16	29.2	28.7600	29.2002	2.4
	LL	-6.5	-3.2	-3.3100	-3.1999	6.0e-01
4	UL	3.25	3.6142	3.6247	3.9892	2.6e-01
	LL	-1.9141	0.9842	0.9863	4	2.1
5	UL	-7.38e-08	0	-2.64e-09	0	1.3e-08
	LL	100	200	193.3333	200	2.53e+01
6	UL	16.8612	17	16.995417	17.0023	2.5e-02
	LL	0.9939	1	0.9997993	1	1.0e-03
7	UL	-12.6905	-12.6795	-12.6800	-12.6792	20e-03
	LL	-1.0156	-1.01545	-1.0153	-1.0133	4.1e-04
8	UL	49	49	49	49	0
	LL	-17	-17	-17	-17	0
9	UL	-1.4074	-1.4074	-1.4074	-1.4074	0
	LL	7.6172	7.6172	7.6172	7.6172	0
10	UL	-1	-1	-1	-1	1.8e-06
	LL	1.09e-12	4.00e-12	1.21e-11	1.44e-10	2.9e-11
11	UL	2250	2250	2250	2250	0
	LL	197.754	197.754	197.754	197.754	0
12	UL	-12	-12	-12	-12	0
	LL	4	4	4	4	0
13	UL	3.1111	3.1111	3.1111	3.1111	0
	LL	-6.6666	-6.6666	-6.6666	-6.6666	0
14	UL	1	1	1	1	0
	LL	0	0	0	0	0
15	UL	980.1060	993.6590	993.2020	999.6570	4.6
	LL	1	1	1	1	0
16	UL	5	5	5	5	0
	LL	4	4	4	4	0
17	UL	9	9	9	9	0
	LL	0	0	0	0	0
18	UL	79.8455	85.0909	84.4802	85.0909	1.5
	LL	-50.1818	-50.1818	-49.9680	-48.3459	5.2e-01

TABLE III. MEDIAN VALUES FOR FUNCTION EVALUATIONS (FE) AND ACCURACY (ACC) FOR THE UPPER AND LOWER LEVELS.

	GA [29]				BlDE	
Pr.	UL FE	LL FE	UL Acc	LL Acc	UL Acc	LL Acc
SMD1	2644	1724241	3.4e-05	1.6e-05	3.491e-06	1.935e-06
SMD2	2404	1568099	5.0e-06	5.0e-06	1.294e-06	6.514e-07
SMD3	2338	1483884	5.9e-05	2.6e-05	4.096e-06	2.917e-06
SMD4	1720	1187981	2.6e-05	2.7e-05	2.296e-05	5.140e-05
SMD5	3010	2093391	4.0e-06	3.0e-06	1.581e-06	1.379e-06
SMD6	3212	2429352	1.45e-04	7.1e-05	3.470e-06	2.067e-06

TABLE IV. SUMMARY OF RESULTS OBTAINED BY BLDE FOR TEST-COLLECTION SMD1 TO SMD6.

Pr.	Level	Min	Median	Mean	Max	std
SMD1	UL	3.73e-07	3.49e-06	3.83e-06	1.23e-05	3.05e-06
	LL	2.46e-07	1.93e-06	2.59e-06	8.00e-06	2.36e-06
SMD2	UL	2.32e-07	1.29e-06	1.68e-06	8.18e-06	1.50e-06
	LL	1.57e-07	6.51e-07	9.43e-07	2.74e-06	7.09e-07
SMD3	UL	1.06e-06	4.09e-06	4.78e-06	9.90e-06	2.56e-06
	LL	4.59e-07	2.91e-06	3.93e-06	1.62e-05	3.62e-06
SMD4	UL	-4.01e-04	-2.29e-05	-4.76e-05	1.24e-06	8.14e-05
	LL	2.4e-07	5.10e-05	1.0e-04	7.76e-04	1.60e-04
SMD5	UL	2.97e-07	1.58e-06	2.93e-06	1.10e-05	2.67e-06
	LL	1.61e-07	1.37e-06	2.06e-06	6.50e-06	1.93e-06
SMD6	UL	3.90e-07	3.47e-06	3.77e-06	1.24e-05	3.08e-06
	LL	2.81e-07	2.06e-06	2.62e-06	8.07e-06	2.39e-06

efficiently solved by an existing problem specific technique.

VII. CONCLUSION

In this paper a simple method using the differential evolution technique was proposed for solving general bilevel programming problems in continuous variables. The algorithm uses two DE algorithms, one responsible for optimizing the upper level problem, and another for optimizing the lower level problem. The algorithm was tested in a variety of test-problems taken from the literature including linear, nonlinear, constrained, and unconstrained optimization problems. Although no tuning procedure for the DE parameters has been attempted in the experiments, the results demonstrated that the proposed algorithm successfully solved the test problems considered.

For future work, it is intended to extend this research in order to propose ways of reducing the number of function evaluations required by the algorithm, specially in the lower level problem.

ACKNOWLEDGMENT

The authors would like to thank the support from CNPq (grants 141519/2010-0, 140785/2009-4, 308317/2009-2).

REFERENCES

[1] P. Hansen, B. Jaumard, and G. Savard, "New branch-and-bound rules for linear bilevel programming," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, no. 5, pp. 1194–1217, 1992.

[2] S. Dempe, "Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints," *Optimization*, vol. 52, no. 3, pp. 333–359, 2003.

[3] S. Das and P. Suganthan, "Differential evolution: A survey of the state-of-the-art," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[4] E. K. da Silva, H. J. C. Barbosa, and A. C. C. Lemonge, "An adaptive constraint handling technique for differential evolution with dynamic use of variants in engineering optimization," *Optimization and Engineering*, vol. 12, pp. 31–54, 2011.

[5] K. Deb and A. Sinha, "Solving bilevel multi-objective optimization problems using evolutionary algorithms," in *Proc. of the 5th International Conference on Evolutionary Multi-Criterion Optimization*, ser. EMO '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 110–124.

[6] W. Candler and R. Townsley, "A linear two-level programming problem," *Computers & Operations Research*, vol. 9, no. 1, pp. 59–76, 1982.

[7] W. Bialas and M. Karwan, "On two-level optimization," *Automatic Control, IEEE Transactions on*, vol. 27, no. 1, pp. 211–214, feb 1982.

[8] H. Onal, "A modified simplex approach for solving bilevel linear programming problems," *European Journal of Operational Research*, vol. 67, no. 1, pp. 126–135, 1993.

[9] M. Campelo and S. Scheimberg, "A note on a modified simplex approach for solving bilevel linear programming problems," *European Journal of Operational Research*, vol. 126, no. 2, pp. 454–458, 2000.

[10] K. Shimizu and E. Aiyoshi, "A new computational method for stackelberg and min-max problems by use of a penalty method," *IEEE Trans. on Automatic Control*, vol. AC-26, no. 2, pp. 460–466, 1981.

[11] G. Anandalingam and D. White, "A solution method for the linear static stackelberg problem using penalty functions," *Automatic Control, IEEE Transactions on*, vol. 35, no. 10, pp. 1170–1173, oct 1990.

[12] Y. Ishizuka and E. Aiyoshi, "Double penalty method for bilevel optimization problems," *Annals of Operations Research*, vol. 34, pp. 73–88, 1992.

[13] Y. Zheng, Z. Wan, K. Sun, and T. Zhang, "An exact penalty method for weak linear bilevel programming problem," *Journal of Applied Mathematics and Computing*, pp. 1–9, 2012.

[14] J. Fortuny-Amat and B. McCarl, "A representation and economic interpretation of a two-level programming problem," *Journal of the Operational Research Society*, vol. 32, no. 9, pp. 783–792, 1981.

[15] J. F. Bard and J. T. Moore, "A branch and bound algorithm for the bilevel programming problem," *SIAM J. Sci. Stat. Comput.*, vol. 11, no. 2, pp. 281–292, 1990.

[16] J. Júdice and A. Faustino, "A sequential lcp method for bilevel linear programming," *Annals of Operations Research*, vol. 34, pp. 89–106, 1992.

[17] C. Shi, J. Lu, G. Zhang, and H. Zhou, "An extended branch and bound algorithm for linear bilevel programming," *Applied Mathematics and Computation*, vol. 180, no. 2, pp. 529–537, 2006.

[18] R. Mathieu, L. Pittard, and G. Anandalingam, "Genetic algorithm based approach to bi-level linear programming," *RAIRO - Operations Research - Recherche Opérationnelle*, vol. 28, no. 1, pp. 1–21, 1994.

[19] J. F. Bard, "An efficient point algorithm for a linear two-stage optimization problem," *Operations Research*, vol. 31, no. 4, pp. 670–684, Jul. - Aug. 1983.

[20] G. Anandalingam, R. Mathieu, L. Pittard, and R. Sinha, *Impact of Recent Computer Advances in Operations Research*. North- Holland, New York, 1989, ch. Artificial Intelligence Based Approaches for Hierarchical Optimization.

[21] S. Hejazi, A. Memariani, G. Jahanshahloo, and M. Sepehri, "Linear bilevel programming solution by genetic algorithm," *Computers & Operations Research*, vol. 29, no. 13, pp. 1913–1925, 2002.

[22] M. Gendreau, P. Marcotte, and G. Savard, "A hybrid tabu-ascent algorithm for the linear bilevel programming problem," *Journal of Global Optimization*, vol. 8, pp. 217–233, 1996.

[23] V. Oduguwa and R. Roy, "Bi-level optimisation using genetic algorithm," in *Proc. of the 2002 IEEE International Conference on Artificial Intelligence Systems (ICAIS'02)*, ser. ICAIS'02. Washington, DC, USA: IEEE Computer Society, 2002, pp. 322–327.

[24] J. F. Bard, *Practical Bilevel Optimization*. Kluwer Academic Publisher, 1998.

[25] X. Zhu, Q. Yu, and X. Wang, "A hybrid differential evolution algorithm for solving nonlinear bilevel programming with linear constraints," in *Cognitive Informatics, 2006. ICCI 2006. 5th IEEE International Conference on*, vol. 1, july 2006, pp. 126–131.

[26] A. Koh, "Solving transportation bi-level programs with differential evolution," in *IEEE Congress on Evolutionary Computation*. IEEE, 2007, pp. 2243–2250.

[27] R. Storn and K. V. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[28] K. Deb, "An efficient constraint handling method for genetic algorithms," *Comput. Methods Appl. Mech. Engrg*, vol. 186, pp. 311–338, 2000.

[29] A. Sinha, P. Malo, and K. Deb, "Unconstrained scalable test problems for single-objective bilevel optimization," in *Evolutionary Computation (CEC), 2012 IEEE Congress on*, June, pp. 1–8.

[30] J. F. Bard and J. E. Falk, "An explicit solution to the multi-level programming problem," *Computers & Operations Research*, vol. 9, no. 1, pp. 77–100, 1982.

[31] E. Aiyoshi and K. Shimizu, "A solution method for the static constrained stackelberg problem via penalty method," *Automatic Control, IEEE Transactions on*, vol. 29, no. 12, pp. 1111–1114, dec 1984.

[32] J. Bard, "Convex two-level optimization," *Mathematical Programming*, vol. 40, pp. 15–27, 1988.

[33] G. Savard and J. Gauvin, "The steepest descent direction for the nonlinear bilevel programming problem," *Operations Research Letters*, vol. 15, no. 5, pp. 265–272, 1994.

[34] J. E. Falk and J. Liu, "On bilevel programming, part i: General nonlinear cases," *Mathematical Programming*, vol. 70, pp. 47–72, 1995.

[35] K. Shimizu and M. Lu, "A global optimization method for the stackelberg problem with convex functions via problem transformation and concave programming," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 25, no. 12, pp. 1635–1640, dec 1995.

[36] J. Rajesh, K. Gupta, H. Kusumakar, V. Jayaraman, and B. Kulkarni, "A tabu search based approach for solving a class of bilevel programming problems in chemical engineering," *Journal of Heuristics*, vol. 9, pp. 307–319, 2003.

[37] V. Visweswaran, C. A. Floudas, M. G. Ierapetritou, and E. N. Pistikopoulos, "A decomposition-based global optimization approach for solving bilevel linear and quadratic programs," in *in C. A. Floudas eds., State of the Art in Global Optimization*. Kluwer Academic Publishers, 1996, pp. 139–162.

[38] W. Guang-Min, W. Zhong-Ping, and W. Xian-Jia, "Solving method for a class of bilevel linear programming based on genetic algorithms," Wuhan University, Tech. Rep., 2003.

[39] W. Guang-Min, W. Zhong-Ping, W. Xian-Jia, and C. Ya-Lin, "Genetic algorithms for solving linear bilevel programming," in *Proc. of the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies*, ser. PDCAT '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 920–924.

[40] W. Guang-Min, W. Xian-Jia, W. Zhong-Ping, and J. Shi-Hui, "An adaptive genetic algorithm for solving bilevel linear programming problem," *Applied Mathematics and Mechanics*, vol. 28, pp. 1605–1612, 2007.

[41] H. Li and L. Fang, "An evolutionary algorithm for solving bilevel programming problems using duality conditions," *Mathematical Problems in Engineering*, vol. 2012, 2012.

[42] F. Facchinei, H. Jiang, and L. Qi, "A smoothing method for mathematical programs with equilibrium constraints," *Mathematical Programming*, vol. 85, pp. 107–134, 1999.

[43] M. S. Radjef and A. Anzi, "Solving linear bilevel programming by dc algorithm," in *Colloque sur l' Optimisation et les Systèmes d'Information (COSI'2010)*, 2010, pp. 434–445.

[44] A. Koh, *Natural Intelligence for Scheduling, Planning and Packing Problems. Studies in Computational Intelligence (250)*. Springer, 2009, ch. A Coevolutionary Particle Swarm Algorithm for Bi-Level Variational Inequalities: Applications to Competition in Highway Transportation Networks.