Differential Evolution to find Stackelberg-Nash Equilibrium in Bilevel Problems with Multiple Followers

Jaqueline S. Angelo*
*Laboratório Nacional de Computação Científica
Petrópolis, RJ, Brazil
Email: jsangelo@lncc.br

Helio J. C. Barbosa*†
†Universidade Federal de Juiz de Fora
Juiz de Fora, MG, Brazil
Email: hcbm@lncc.br

***Abstract*—This paper considers bilevel programming problems with one leader and multiple followers when there is information shared among the followers. In this model, the objective function and the set of constraints of each follower include the leader's variables and the variables of the other followers. We propose a differential evolution method to solve the so called Stackelberg–Nash problem, in which the leader announces a decision first and then, in response to this decision, the followers make their decisions simultaneously solving a Nash equilibrium problem. The algorithm performance is illustrated by means of several test problems.**

***Keywords*—bilevel optimization, multiple followers, Stackelberg-Nash equilibrium, differential evolution.**

I. Introduction

Bilevel programming (BLP) problems were first introduced by Stackelberg [1] in the field of game theory, and became known as Stackelberg problems. Those problems are used to model decentralized decision making in which two agents, the leader in the upper level and the follower in the lower level, are hierarchically related. The constraint region of the leader is determined by the solution of the follower's problem, which in turn, makes a decision considering the leader's move. In such nested structure, unless the solution to the lower level optimization problem is optimal, the solution to the upper level problem cannot be feasible. Also, bilevel problems are generally non-convex, even when all functions involved are linear [2].

In BLP, the decision variables are partitioned among the decision makers who seek to optimize their individual payoff functions. The main feature of such problems is that the leader's decisions affect both the follower's payoff and allowable actions, and vice-versa. In many real-word applications such hierarchical structure appears naturally, as for example in the areas of economics, engineering, transportation planning, management, etc [3]. Although much research has been conducted in this field, the studies in BLP have been mainly focused on situations comprising one leader and one follower.

In this paper, we are interested in the more general case of bilevel problems involving one leader and multiple followers, also referred to as Stackelberg-Nash problem [4]. In this problem the leader's decision is affected by the strategies of multiple followers. The followers are considered to be (inter)dependent, that is, they share some decision variables and constraints but have different objective functions.

In the model considered here, a Stackelberg solution is generated for the upper level problem, in which the leader chooses an optimal strategy knowing that the followers react by reaching the Nash equilibrium, which we assume to be unique.

The purpose of this paper is to develop an optimization method based on differential evolution (DE) which generates Stackelberg-Nash solutions to the BLP problem while, for each individual of the leader's population, multiple followers compute a Nash equilibrium solution. The DE used to obtain Stackelberg-Nash solutions is an extension of an algorithm previously proposed by the authors in [5] in which two DE algorithms were used, each one responsible for the optimization of one level. To obtain the Nash equilibrium solution for the lower level problems, an optimization scheme is developed, based in the so called "Nash strategy" proposed in [6].

In the next section we present the formulation of the bilevel optimization problem with one leader and multiple followers, in which information is shared among the followers. Also in section III, definitions about the Stackelberg-Nash solution will be presented. In Section IV, the description of the proposed algorithm, named BLDE$_{MDF}$, is presented, including the proposed scheme to obtain the Nash equilibrium solution to the lower level problem. Section V describes the test-problems and the computational experiments conducted to analyze and validate the performance of the proposed algorithm. In addition, the algorithm convergence is illustrated in each test case. The paper ends with conclusions presented in Section VI.

II. Related works

Compared with a number of studies on handling bilevel problems with one follower, there are few works which handle the case with multiple followers. When considering multiple followers, who may share different kinds of information (decision variables, objectives and constraints), the leader's

978-1-4799-7492-4/15/$31.00

decision will be affected not only by the reactions of these followers, but also by the relationship among them. The relationship among these multiple followers can be varied and each situation requires a specific model for describing the problem, and a specific approach for deriving an optimal solution for the leader's problem. A framework of bilevel problems with multiple followers and their relationship can be found in [7].

The model considering independent followers is the most common one, in which the objective function and the set of constraints of each follower only include the leader's variables and his own variables. One can see for example [7], [8], [9], [10] where such model is explored.

In this paper we consider the case with information shared among the followers, meaning that, at the lower level, the decision variables and constraints are shared among the followers. Since all the followers are of equal status, and they must reveal their strategies simultaneously, we use the Nash equilibrium concept as a solution to the lower level problem.

Such model was investigated in [11] and [4] where a genetic algorithm (GA) was used to solve the problem. In [11] the authors used a GA to evolve and generate the leader's variables while the Nash equilibrium solution was obtained in two different forms: by an iterative method, or by another GA, if the iterative method fails to obtain the Nash equilibrium point.

The approach proposed in [4] applies a GA on both levels of the optimization process. The authors proposed the *Nash Genetic Algorithm*, which consist of two populations. Each population is associated with a follower/player, where the optimization task of each player is performed by one of the populations. The structure of this method is similar to those adopted by other researchers [6], [12], and is also used here.

III. The Bilevel Programming Model

We consider the BLP with one leader and m interdependent followers. Let x and y_i be the control variables of the leader and the i-th follower, respectively, with $i = 1, ..., m$. Also, assume that $F(x, y_1, ..., y_m)$ and $f_i(x, y_1, ..., y_m)$, $i = 1, ..., m$, are the objective function of the leader and the i-th follower, respectively. In this model, the followers are considered to be (inter)dependent, that is, they share decision variables and constraint but have separate objective functions.

The general case of a BLP problem with multiple (inter)dependent followers (BLMDF) can be written as[1]:

$$\begin{array}{ll} \min\limits_{x \in X} & F(x, y_1, ..., y_m) \\ \text{subject to} & G(x, y_1, ..., y_m) \leq 0 \\ & \min\limits_{y_i \in Y_i} f_i(x, y_1, ..., y_m) \\ & \text{subject to} \quad g(x, y_1, ..., y_m) \leq 0 \end{array} \tag{1}$$

where $x \in X \subset R^n$ and $y_i \in Y_i \subset R^{m_i}$, while $G(x, y_1, ..., y_m)$ and $g(x, y_1, ..., y_m)$ describe the constraints set of the leader and of the m followers, respectively. In (1), the objective function of the i-th follower includes the leader's variables, its own variables and the variables of the other followers.

The feasible set of the decision variables x of the leader is defined by

$$\begin{array}{rcl} \Omega & := & \{(x, y_1, y_2, ..., y_m) : x \in X, y_i \in Y_i, \\ & & G(x, y_1, y_2, ..., y_m) \leq 0, \\ & & g(x, y_1, y_2, ..., y_m) \leq 0, i = 1, ..., m\} \end{array} \tag{2}$$

For each x chosen by the leader, the feasible set of the decision variables y_i of the i-th follower is defined by

$$\Omega_{y_i} := \{y_i \in Y_i : g(x, y_1, y_2, ..., y_m) \leq 0, i = 1, ..., m\} \tag{3}$$

and the projection of Ω onto the leader's decision space is

$$\begin{array}{rcl} \Omega(X) & := & \{x \in X : \exists y_i \in Y_i, \\ & & G(x, y_1, y_2, ..., y_m) \leq 0, \\ & & g(x, y_1, y_2, ..., y_m) \leq 0, i = 1, ..., m\} \end{array} \tag{4}$$

Hence, the rational reaction set of the i-th follower for $x \in \Omega(X)$ is

$$\begin{array}{rcl} R_i(x) & = & \{y_i \in Y_i : y_i(x) \in \\ & & \arg\min\{f_i(x, \hat{y}_i, y_j, j = 1, ..., m, j \neq i) \\ & & : \hat{y}_i \in \Omega_{y_i}\}, \text{ where } i = 1, ..., m. \end{array} \tag{5}$$

Since all followers, at the same level, may share information, they are assumed to define their strategies simultaneously for a given x. Hence, the optimal solution of the followers for a given $x \in X$, the so called *Nash equilibrium solution* [13], denoted as $(x, y_1^*, y_2^*, ..., y_m^*)$, must satisfy

$$f_i(x, y_1^*, ..., y_{i-1}^*, y_i^*, y_{i+1}^*, ..., y_m^*) \leq \tag{6}$$
$$f_i(x, y_1^*, ..., y_{i-1}^*, y_i, y_{i+1}^*, ..., y_m^*) \tag{7}$$

for any $(y_1^*, ..., y_{i-1}^*, y_i, y_{i+1}^*, ..., y_m^*) \in Y_i$ and $i = 1, ..., m$.

In Nash equilibrium, no follower can improve its own objective function by altering its strategy unilaterally, according to the Nash equilibrium definition [13].

Finally, the inducible region (or feasible set) of problem (1) is defined as:

$$\begin{array}{rcl} RI & := & \{(x, y_1, y_2, ..., y_m) : (x, y_1, y_2, ..., y_m) \in \Omega, \\ & & y_i \in R_i(x), i = 1, ..., m\} \end{array} \tag{8}$$

The *Stackelberg-Nash* solution of the bilevel problem (1) is then defined as $(x^*, y_1^*, y_2^*, ..., y_m^*)$ where $x^* \in \Omega$ and $(y_1^*, y_2^*, ..., y_m^*)$ is the Nash equilibrium solution of the followers with respect to x^*. Hence, $(x^*, y_1^*, y_2^*, ..., y_m^*)$ is a Stackelberg-Nash solution if and only if,

$$F(x^*, y_1^*, y_2^*, ..., y_m^*) \leq F(\overline{x}, \overline{y}_1, \overline{y}_2, ..., \overline{y}_m) \tag{9}$$

for any $\overline{x} \in \Omega$ and the Nash equilibrium $(\overline{y}_1, \overline{y}_2, ..., \overline{y}_m)$ with respect to $\overline{x}$.

[1] Although the formulation (1) presents constraints at the upper level, in the test problems analyzed, there are no constraints at that level.

IV. Bilevel Differential Evolution for Stackelberg-Nash Equilibrium

A. Differential Evolution Method

Differential Evolution (DE) [14] is a simple stochastic population based algorithm for continuous global optimization. Its simplicity comes from the fact that it requires very few control parameters, basically the population size (pop), a scale factor (θ), and the crossover rate (CR). As many evolutionary methods, DE starts with a set of candidate solutions, represented by vectors, randomly distributed in the search space. New solutions are generated by perturbing the current population members with scaled differences of distinct randomly selected population members. The number of differences applied, the way in which the individuals are selected, and the type of recombination determine the different strategy or variant of the DE.

The DE variant adopted in this paper is the DE/rand/1/bin in which the mutation operation performed to obtain a new individual $u_{i,G+1}$, is given by

$$u_{i,G+1} = x_{r_1,G} + \theta.(x_{r_2,G} - x_{r_3,G}) \tag{10}$$

where x_{r_1}, x_{r_2} and x_{r_3} are randomly selected individuals in the current population, with $r_1 \neq r_2 \neq r_3 \neq i$, and F is the scale factor used to control the amplitude of the search in the direction of the applied difference. In addition, a crossover operation is performed, using the parameter CR. After those procedures the target vector $x_{i,G}$ (the one to be compared with the new individual) is compared with the vector $u_{i,G+1}$ and the one with the best function value is admitted to the next generation. Mutation, recombination and selection continue until a stopping criterion is reached.

For each design variable, lower and upper bounds are applied. Whenever a given component j of a candidate solution x_i is generated outside its prescribed range, a standard projection operation is performed:

$$\begin{aligned} &\text{If } x_{i,j} > x_j^U \quad \text{then} \quad x_{i,j} = x_j^U; \\ &\text{If } x_{i,j} < x_j^L \quad \text{then} \quad x_{i,j} = x_j^L. \end{aligned}$$

B. Strategy to Compute the Nash Equilibrium Solution

Inspired by the work of Sefrioui and Periaux [6], a new scheme is proposed to generate Nash equilibrium solutions.

Let $s = \{y, z\}$ be the string (or individual) representing a potential Nash equilibrium solution for two decision makers (or players). Denote by y the subset of variables handled by player 1, which optimizes the objective function f_1. Similarly, z denotes the subset of variables handled by player 2, which optimizes the objective function f_2. Thus, according to the Nash equilibrium definition [13], player 1 optimizes s with respect to f_1 by modifying y while z is fixed by the player 2; symmetrically, player 2 optimizes s with respect to f_2 by modifying z, while y is fixed by player 1.

Let y_{k-1} and z_{k-1} be the best values found by players 1 and 2, respectively, at generation $k-1$. In [6], at generation k, player 1 optimizes y_k while using z_{k-1} in order to evaluate s (in this case, $s = \{y_k, z_{k-1}\}$). At the same time player 2 optimizes z_k while using y_{k-1} in order to evaluate s (now, $s = \{y_{k-1}, z_k\}$). That is, at each generation, two optimization processes are carried out, one to optimize y and the other to optimize z at the same time. The main difference between the scheme proposed in [6] is that here we eliminate one optimization process at each generation, as given in Fig. 1. As a consequence, in a sequential computer, this procedure becomes less expensive.

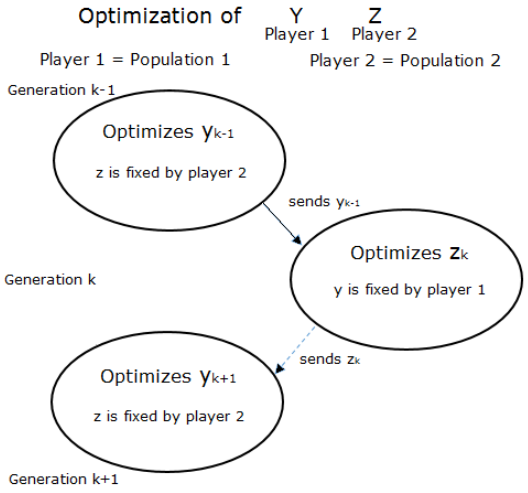

Fig. 1. New scheme to generate Nash equilibrium solution based in [6].

C. Constraint Handling

The upper and lower level constraints of the bilevel problems are handled by the method proposed in [15], which enforces the following criteria: (i) any feasible solution is preferred to any infeasible solution; (ii) between two feasible solutions, the one having better objective function value is preferred, and (ii) among two infeasible solutions, the one having smaller constraint violation is preferred.

These criteria are rigorously enforced in both upper and lower levels for the treatment of the constraints at that level. In addition, every time the final lower level solution violates any lower level constraint, the corresponding upper level objective function value is penalized via

$$v_j(x,y,z) = \begin{cases} |g_j(x,y,z)|, & \text{(a)} \\ \max\{0, g_j(x,y,z)\}, & \text{(b)} \end{cases} \tag{11}$$

where (a) is used for an equality constraint and (b) for an inequality constraint violation. Hence, when there is a violation of any lower level constraint, the upper level objective function is penalized as follows:

$$F(x,y,z) = F(x,y,z) + \sum_{j=1}^{p_1} v_j(x,y,z) + \sum_{j=1}^{p_2} v_j(x,y,z) \tag{12}$$

where p_1 and p_2 denote the number of constraints of the first and the second follower, respectively.

D. The Bilevel Algorithm Description

The pseudo-code of the algorithm proposed is presented in Algorithms 1 and 2. The solution method uses two DE algorithms to solve the upper and the lower level problems. The algorithms description is as follows:

Initialization process: The algorithm starts with two random populations of size np and s, which generate the required number of upper and lower level variables. The population s contains the decision variables of both followers ($s = \{y, z\}$). For the upper level population, the fitness is assigned based on upper level function value and constraints, while for the lower level population, the fitness is assigned based on lower level function value and constraints both corresponding to each follower.

Upper level optimization process: Following the basic DE algorithm described in Algorithm 1, the upper level individuals are mutated and recombined. Then, for each upper level variable, the lower level algorithm (BLDE-Follower) is executed n_cycles times for each follower in order to obtain the Nash equilibrium solution, which we assume to be unique. The Nash equilibrium solution returned, that is, the y and z values associated with the given x, is used to evaluate the upper level individuals. If the lower level solution returned violates any lower level constraints, equation (12) is used to evaluate the upper level solution. Given the lower level variables, the upper level constraints are analyzed[2] and the upper level solution is evaluated. If the new upper level solution ($u_{i,G+1}$) is better than the one generated in the previous generation ($u_{i,G}$), then $u_{i,G+1}$ and its associated Nash solution (y and z) are considered for the next generation. Finally, when gen generations are executed, the algorithm returns the best upper level solution found in the last generation, which is a Stackelberg-Nash solution.

Computing the Nash equilibrium solution: First, Algorithm 2 starts with random populations of size npf, which generate the required lower level variables y. The fitness of each individual is assigned based on lower level objective function value and constraints, with variables x and z fixed. Following the basic DE algorithm described in Algorithm 2, the lower level individuals are mutated and recombined. The individuals are evaluated based on the lower level function with variables x and z fixed. The constraints are checked following Deb's selection criteria presented in section IV-C. Finally, after $genf$ generation the algorithm returns the best value of y. Next, Algorithm 2 executes again to generate the lower level variables z. The description of the optimization process for the second follower is analogous to the first follower, where now variables z are generated with x and y (previously obtained) fixed. This process is repeated n_cycles times. After that, a Nash equilibrium solution is generated.

V. Computational experiments

In order to validate the effectiveness and the performance of the algorithm proposed, we analyze five test-problems, four of them collected from the literature. The experiments were performed on a personal computer with an Intel Core i7, 16GB of memory, under Windows 8.

For the upper and lower level algorithms, the population size is set to 30, the mutation scale is $\theta = 0.8$ and the crossover rate is $CR = 0.9$. For all test-problems 30 independent runs were performed. The total number of generations in the upper level was set to $gen = 50$. In order to calculate the number of

[2]Although there are no constraints in the upper level of the test problems, the algorithm proposed was designed to also deal with upper level constraints.

Algorithm 1: BLDE$_{MDF}$.

Input : np (leader population size), s ("Nash population" size), θ (mutation scale), CR (crossover rate), gen (# of generation), n_cycles (number of cycles)

```
G ← 0;
RandonInitPopulation(np);        /* Values for x */
RandonInitPopulation(s);  /* Values for y an z */
for i ← 1 to np do
    Evaluate F(x_{i,G}, y_{i,G}, z_{i,G});
for G ← 1 to gen do
    for i ← 1 to np do
        SelectRandomly(r1, r2, r3) /* r1 ≠ r2 ≠ r3 ≠ i */
        jRand ←RandInt(1, n); /* n is the number of
        variables of the leader's problem */
        for j ← 1 to n do
            if Rand(0,1) < CR or j = jRand then
                u_{i,j,G+1} = x_{r1,j,G} + θ.(x_{r2,j,G} − x_{r3,j,G});
            else
                u_{i,j,G+1} = x_{i,j,G};

        // Computing Nash equilibrium solution
        z = SelectBest (s);
        for c ← 1 to n_cycles do
            y=BLDE-F.(npf1, genf1, θ, CR, u_{i,G+1}, z);
            z=BLDE-F.(npf2, genf2, θ, CR, u_{i,G+1}, y);

        AnalyzeConstraints (u_{i,G+1}, y, z);
        if F(u_{i,G+1}, y, z) ≤ F(x_{i,G}, y, z) then
            x_{i,G+1} = u_{i,G+1};
            y_{i,G+1} = y; z_{i,G+1} = z;
        else
            x_{i,G+1} = x_{i,G};
            y_{i,G+1} = y_{i,G}; z_{i,G+1} = z_{i,G};
```

Output: SelectBest (np)

Algorithm 2: BLDE-Follower (BLDE-F).

Input : npf (follower population size), genf (# of generation), F (mutation scale), CR (crossover rate), $\vec{v}$ (leader variables), $\vec{w}$ (variables of the other follower)

```
G ← 0;
CreateRandonInitialPopulation(npf);
for i ← 1 to npf do
    Evaluate f(v, x_{i,G}, w);
for G ← 1 to genf do
    for i ← 1 to npf do
        SelectRandomly(r1, r2, r3) /* r1 ≠ r2 ≠ r3 ≠ i */
        jRand ←RandInt(1, nf) /* nf is the number of
            variables of the follower's problem */
        for j ← 1 to nf do
            if Rand(0,1) < CR or j = jRand then
                u_{i,j,G+1} = x_{r3,j,G} + F.(x_{r1,j,G} − x_{r2,j,G});
            else
                u_{i,j,G+1} = x_{i,j,G};

        Evaluate f(v, u_{i,G+1}, w);
        AnalyzeFollowerConstraints (v, u_{i,j,G+1}, w);
        if f(v, u_{i,G+1}, w) ≤ f(v, x_{i,G}, w) then
            x_{i,G+1} = u_{i,G+1};
        else
            x_{i,G+1} = x_{i,G};
```

Output: SelectBest (npf)

generations in the lower level we use the following expression, with $genf_1 = genf_2 = genf$:

$$ng = n\_cycles \times (genf_1 + genf_2). \quad (13)$$

We analyze the proposed method with a fixed value for ng. Hence, we adopted three different configurations for n_cycles and $genf$ in order to obtain $ng = 200$, which are: $2\times(50+50)$, $5 \times (20 + 20)$ and $10 \times (10 + 10)$.

Tables I–IV present the summary of the results obtained by the proposed BLDE$_{MDF}$ for the test-problems 1, 2, 3 and 4. The column headers denoted by "Min", "Med", "Avg", "Max" and "Std.Dev" indicate respectively the minimum, median, average, maximum, and standard deviation values. The line headers denoted by "UL", "LL$_1$" and "LL$_2$" indicate the values for the upper and lower level problems, respectively, where "LL$_1$" and "LL$_2$" are related to the first and the second follower.

A. Test-problem 1

In the first test-problem, proposed in this paper, the leader controls variables $x = (x_1, x_2) \in [-5, 5]^2$, and the followers control variables y and z, respectively, with $y, z \in [0, 10]$. The problem is as follows:

$$\begin{array}{ll} \min\limits_{x} & F(x,y,z) = -(x_1 + x_2)^2 + \frac{1}{2}(y^2 + z^2) \\ \text{s.t.} & \min\limits_{y} \quad f_1(x,y,z) = (y - (x_1 + 2))^2 + (z - (x_2 + 1))^2 \\ & \min\limits_{z} \quad f_2(x,y,z) = (y - (x_1 + 1))^2 + (z - (x_2 + 2))^2 \end{array} \quad (14)$$

In order to obtain the Nash equilibrium solution for the lower level problem we need first to define the rational reaction sets of the followers. For a given x, the rational reaction set of the first and second followers, with strategy sets Y and Z, respectively, are defined as:

$$R_y = \{(x, y^*, z) \in Y \times Z : f_1(x, y^*, z) \leq f_1(x, y, z)\}$$
$$R_z = \{(x, y, z^*) \in Y \times Z : f_2(x, y, z^*) \leq f_2(x, y, z)\}$$

Due to the smoothness of the problem and the absence of constraints, R_y and R_z can be built by finding the y and z that satisfy the following equations:

$$R_y = \Big\{y | \frac{\partial f_1(x,y,z)}{\partial y} = 0\Big\}, \quad R_z = \Big\{z | \frac{\partial f_2(x,y,z)}{\partial z} = 0\Big\}.$$

The Nash equilibrium is the intersection of this two rational reaction sets. Returning to problem (14), the rational reaction sets R_y and R_z are obtained from the solution of the equations:

$$\frac{\partial f_1(x,y,z)}{\partial y} = 0 \Leftrightarrow 2(y - (x_1 + 2)) = 0 \Leftrightarrow y = x_1 + 2.$$

$$\frac{\partial f_2(x,y,z)}{\partial z} = 0 \Leftrightarrow 2(z - (x_2 + 2)) = 0 \Leftrightarrow z = x_2 + 2.$$

Hence, for a given x, the Nash equilibrium solution of the lower level problem is $y^N = x_1 + 2$ and $z^N = x_2 + 2$. Replacing these values in the upper level objective function, we have:

$$\begin{aligned} F(x,y,z) &= -(x_1 + x_2)^2 + \tfrac{1}{2}(y^2 + z^2) \\ &= -x_1 - x_2 + \tfrac{1}{2}((x_1 + 2)^2 + (x_2 + 2)^2) \\ &= -x_1 - x_2 + \tfrac{1}{2}(x_1^2 + 4x_1 + 4) + \tfrac{1}{2}(x_2^2 + 4x_2 + 4) \end{aligned}$$

The Stackelberg-Nash solution of problem (14) is then obtained by solving:

$$\frac{\partial F(x,y,z)}{\partial x_1} = 0 \Leftrightarrow -1 + \tfrac{1}{2}(2x_1 + 4) = 0 \Leftrightarrow x_1 = -1.$$

$$\frac{\partial F(x,y,z)}{\partial x_2} = 0 \Leftrightarrow -1 + \tfrac{1}{2}(2x_2 + 4) = 0 \Leftrightarrow x_2 = -1.$$

resulting in $(x^*, y^*, z^*) = (-1, -1, 1, 1)$, which gives $F^* = 3$, $f_1^* = 1$, and $f_2^* = 1$.

By using the proposed algorithm we obtain the solutions summarized in Table I for three different configurations of n_cycles and $genf$, indicated in the table as $n\_cycles \times (genf + genf)$.

The results demonstrate that the proposed algorithm was capable of reaching the optimal solution in all executions. The results also indicate that the algorithm generates better solutions when using $n\_cycles = 2$ with $genf = 50$ and $n\_cycles = 5$ with $genf = 20$.

Fig. 2 shows the history of the solution value of the best individual in the leader's population for each configuration of n_cycles and $genf$. This plot demonstrates the convergence of the solutions toward the optimal one, in this case, $F^* = 3$.

TABLE I. SUMMARY OF THE RESULTS – TEST-PROBLEM 1.

Level	Objective Function Value Min	Med	Avg	Max	Std.Dev.	Exact solution	Time in (s)
$2 \times (50 + 50)$							
UL	2.9979	2.9983	2.9982	2.9985	0.0002		
LL$_1$	0.9975	0.9982	0.9982	0.9989	0.0003		2.47
LL$_2$	0.9975	0.9980	0.9980	0.9986	0.0002		
$5 \times (20 + 20)$							
UL	2.9976	2.9981	2.9980	2.9983	0.0002	F^* = 3	
LL$_1$	0.9974	0.9980	0.9980	0.9986	0.0004	f_1^* = 1	2.56
LL$_2$	0.9972	0.9977	0.9978	0.9985	0.0004	f_2^* = 1	
$10 \times (10 + 10)$							
UL	2.9708	2.9825	2.9817	2.9878	0.0035		
LL$_1$	0.9523	0.9880	0.9835	1.0061	0.0172		2.76
LL$_2$	0.9415	0.9746	0.9761	1.0037	0.0171		

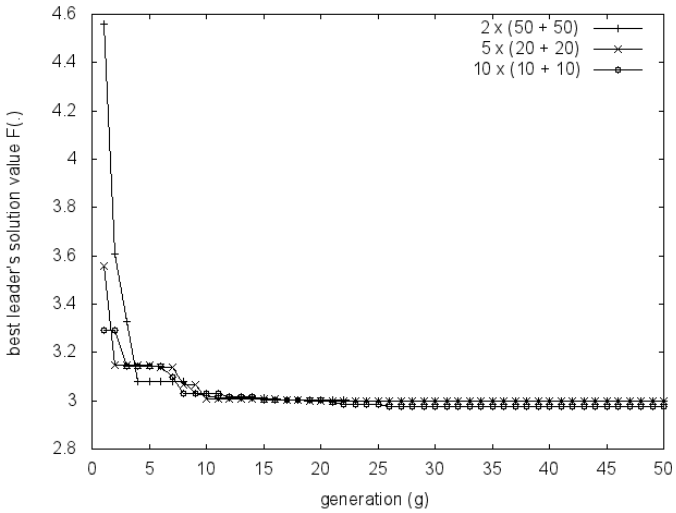

Fig. 2. History of the solution value of the best leader's individual, for three different configurations of n_cycles and $genf$ in test-problem 1.

B. Test-problem 2

In the second test-problem, described in [16], the lower level constraints are shared between the followers. The problem is as follows:

$$\begin{aligned} \max_{x} \quad & F(x,y,z) = 7x + 5y + 8z \\ \text{s.t.} \quad & \max_{y} \quad f_1(x,y,z) = 3y + z \\ & \max_{z} \quad f_2(x,y,z) = y - z \\ & \text{s.t.} \quad x + y + z \leq 3, \quad -x + y \leq 0, \quad y + z \leq 2, \\ & \qquad x - y - z \leq 1, \quad x,y,z \geq 0. \end{aligned} \tag{15}$$

The best solution reported in [16] occurs at point $(x^*, y^*, z^*) = (2, 0, 1)$, which gives $F^* = 22$, $f_1^* = 1$, and $f_2^* = -1$. In this reference solution, two of the four constraints are active.

For this test-problem, the computational results are summarized in Table II. The proposed method was able to reach the reference solution in more than 80% of the executions, in all configurations of n_cycles and $genf$. By examining the value of the standard deviation, the algorithm presented a more stable behavior when $n\_cycles = 10$ with $genf = 10$.

TABLE II. SUMMARY OF THE RESULTS – TEST-PROBLEM 2.

Objective Function Value						Reference	Time
Level	Min	Med	Avg	Max	Std.Dev.	solution	in (s)
$2 \times (50 + 50)$							
UL	0	21.9636	18.3041	21.9991	8.3258		
LL_1	0	0.9976	0.8314	0.9999	0.3782		2.46
LL_2	-0.9999	-0.9976	-0.8314	0	0.3782		
$5 \times (20 + 20)$							
UL	0	21.9952	18.3248	22	8.3352	F^* = 22	
LL_1	0	0.9997	0.8332	1.0096	0.3790	f_1^* = 1	2.61
LL_2	-1.0096	-0.9997	-0.8332	0	0.3790	f_2^* = -1	
$10 \times (10 + 10)$							
UL	0	22.0170	19.7882	22.1726	6.7119		
LL_1	0	1.0270	0.9286	1.1753	0.3166		2.85
LL_2	-1.1753	-1.0270	-0.9286	0	0.3166		

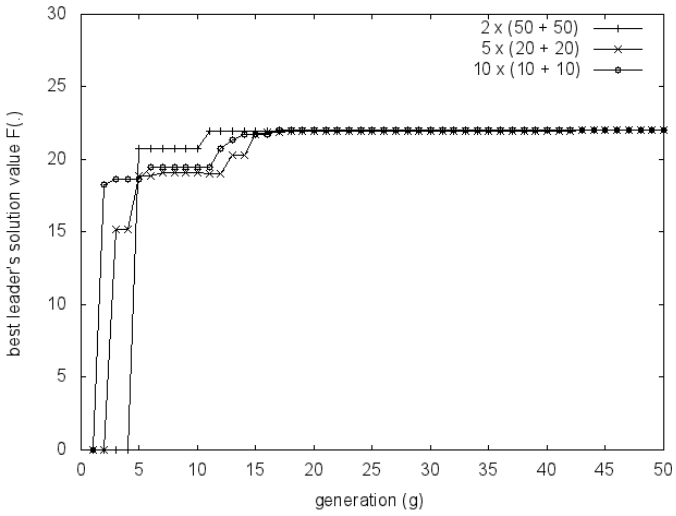

Fig. 3. History of the solution value of the best leader's individual, for three different configurations of n_cycles and $genf$ in test-problem 2.

The history of the best leader's solution value is shown in Fig. 3 for each configuration of n_cycles and $genf$, which indicates the convergence of the solutions toward the optimal solution $F^* = 22$.

C. Test-problem 3

In the third test-problem, the lower level constraints are shared between the followers. The problem is stated as follows [16]:

$$\begin{aligned} \max_{x} \quad & F(x,y,z) = 7x + 5y + 8z \\ \text{s.t.} \quad & \max_{y} f_1(x,y,z) = -2y^2 - 2yz + 3y \\ & \max_{z} f_2(x,y,z) = -yz - z^2 + 6z \\ & \text{s.t.} \quad x + y + z \leq 3, \quad -x + y \leq 0, \quad y + z \leq 2, \\ & \qquad x - y - z \leq 1, \quad x,y,z \geq 0. \end{aligned} \tag{16}$$

The best solution reported in [16] occurs at point $(x^*, y^*, z^*) = (1.5, 0, 1.5)$, which gives $F^* = 22.5$, $f_1^* = 0$, and $f_2^* = 6.75$. In this reference solution, one of the four constraints is active.

The summary of the results generated is presented in Table III. For this problem, the proposed method was able to find a solution better than the reference solution reported in [16]. The new solution generated was $(x^*, y^*, z^*) = (1, 0, 2)$ which gives $F^* = 23$, $f_1^* = 0$, and $f_2^* = 8$. In this new solution, two of the four constraints are active.

For test-problem 3, the history of the solution value of the best leader's individual for each configuration of n_cycles and $genf$ is shown in Fig. 4, indicating the convergence of the solutions toward the optimal one, in this case $F^* = 23$.

TABLE III. SUMMARY OF THE RESULTS – TEST-PROBLEM 3.

Objective Function Value						Reference	Time
Level	Min	Med	Avg	Max	Std.Dev.	solution	in (s)
$2 \times (50 + 50)$							
UL	16	23	21.9617	26.8506	2.8008		
LL_1	0	0	0.0297	0.8908	0.1626		2.44
LL_2	5.1480	8	7.9049	8	0.5207		
$5 \times (20 + 20)$							
UL	16	22.9997	21.8614	23.8525	2.6706	F^* = 23	
LL_1	0	0	0.0022	0.0648	0.0118	f_1^* = 0	2.61
LL_2	6.4064	7.9997	7.9465	8	0.2909	f_2^* = 8	
$10 \times (10 + 10)$							
UL	15.9994	22.9937	22.5108	26.2046	2.3026		
LL_1	0	0	0.0559	1.0479	0.2034		2.87
LL_2	3.6647	7.9930	7.7478	8	0.8356		

D. Test-problem 4 and 5

In this section we validate the proposed method in the test-problem proposed in [4]. This is a unconstrained problem with the leader variable $x \in [0, 1]$ and the followers variables $y, z \in [0, 1]$. The test-problem is described as:

$$\begin{aligned} \min_{x} \quad & F(x,y,z) = z(x - y - \tfrac{1}{2})^2 \\ \text{s.t.} \quad & \min_{y} f_1(x,y,z) = (z - y)^2 + (2y - x)^2 \\ & \min_{z} f_2(x,y,z) = (y - z)^2 + \frac{(2z - x)^2}{2} \end{aligned} \tag{17}$$

It was shown in [4] that this problem has two optimal solutions: $(x^*, y^*, z^*)_1 = (1, 1/2, 1/2)$ and $(x^*, y^*, z^*)_2 =$

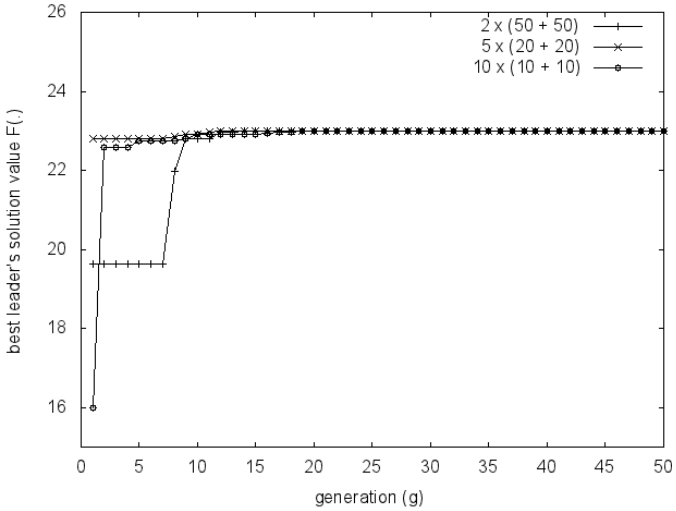

Fig. 4. History of the solution value of the best leader's individual, for three different configurations of n_cycles and $genf$ in test-problem 3.

$(0, 0, 0)$, which gives $F^* = 0$, $f_1^* = 0$ and $f_2^* = 0$ on both solutions. Fig. 5 illustrate the leader's objective function history generated by the genetic algorithm proposed in [4]. Such figure shows both optimal solutions values for the upper level problem (SN solutions), in this case when $x = 0$ and $x = 1$.

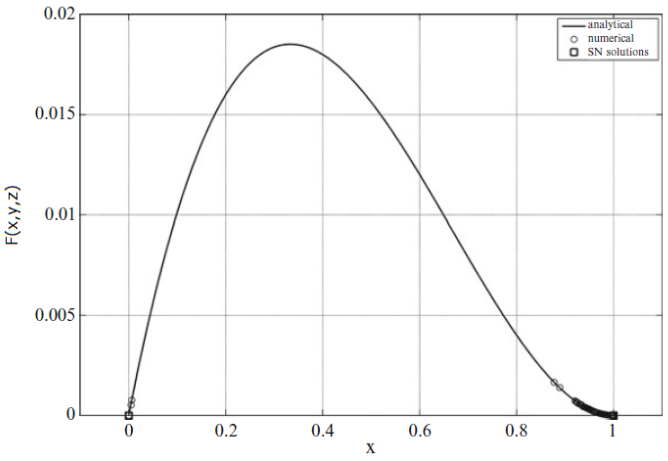

Fig. 5. History of the leader's cost function reported in [4].

By examining Fig. 5, one can observe that if in problem (17) a maximization problem were considered in the upper level, one could generate a test-problem which has only one optimal solution at this level. In this case, the optimal solution x^* would occur between 0.2 and 0.4 which gives an optimal upper level solution value between 0.015 and 0.02.

Hence, we first analyze the proposed method in test-problem (17) considering a maximization problem at the upper level. In such case, the proposed method should find an optimal upper level solution between 0.015 and 0.02. This problem will be called test-problem 4. Next, the proposed method will be analyzed in the original problem described in (17), which has two optimal solutions. This last problem will be called test-problem 5.

For test-problem 4 the computational results are summarized in Table IV. In all configurations considered for the parameters n_cycles and $genf$, the proposed method converges to the solution $(x, y, z) = (0.333, 0.167, 0.167)$, which gives $F = 0.0185$. The convergence of the upper level solutions toward this optimal value is shown in Fig. 6.

TABLE IV. SUMMARY OF THE RESULTS – TEST-PROBLEM 4.

	Objective Function Value					Reference	Time
Level	Min	Med	Avg	Max	Std.Dev.	solution	in (s)
			$2 \times (50 + 50)$				
UL	0.01852	0.01857	0.01857	0.01859	0.00002		
LL_1	0	0	0	0	0		2.45
LL_2	0	0	0	0	0		
			$5 \times (20 + 20)$				
UL	0.01856	0.01858	0.01858	0.0186	0.00001	$F^* = 0.0185$	
LL_1	0	0	0	0	0	$f_1^* = 0$	2.61
LL_2	0	0	0	0	0	$f_1^* = 0$	
			$10 \times (10 + 10)$				
UL	0.01855	0.0186	0.0186	0.01863	0.00002		
LL_1	0	0	0	0	0		2.80
LL_2	0	0	0	0	0		

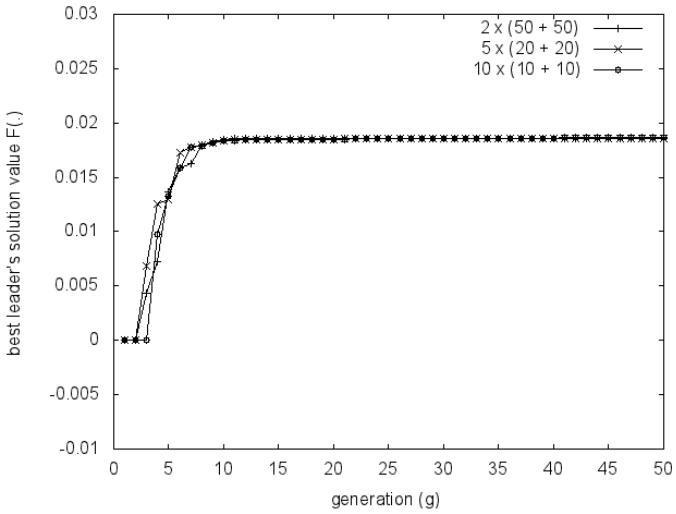

Fig. 6. History of the solution value of the best leader's individual, for three different configurations of n_cycles and $genf$ in test-problem 4.

For test-problem 5, the proposed method was capable of finding both optimal solutions, $x^* = 0$ and $x^* = 1$, for all configurations of the parameters n_nycles and $genf$. Fig. 7 presents the number of times the optimal solutions $x^* = 0$ and $x^* = 1$ were found by the proposed method in 30 executions. It is possible to see that the algorithm's configuration corresponding to $n\_cycles = 10$ and $genf = 10$ presented a more stable behavior with relation to the number of solutions $x^* = 0$ and $x^* = 1$ found. In this parameter configuration, in 30 runs, the algorithm converges 18 times to solution $x^* = 0$ and 12 times to solution $x^* = 1$. The computational time for this experiments were 2.43s, 2.59s and 2.81s in parameter configurations $2 \times (50 + 50)$, $5 \times (20 + 20)$ and $10 \times (10 + 10)$, respectively.

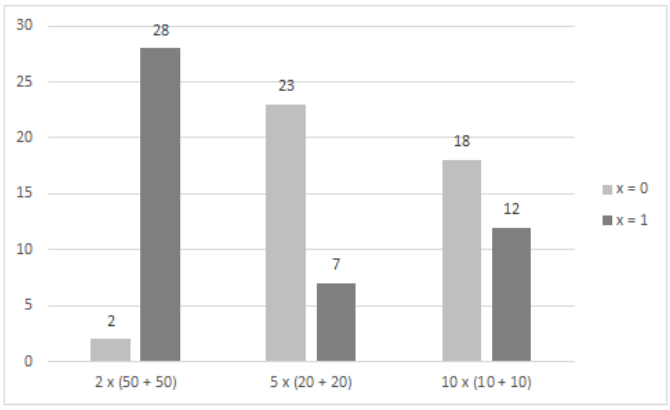

Fig. 7. Number of solutions $x = 0$ and $x = 1$ found in 30 executions, for three different configurations of n_cycles and $genf$ in test-problem 5.

VI. Conclusion

In this paper we focused on bilevel optimization problems with multiple followers with shared information among them. Based on a previous method developed in [5], a new bilevel differential evolution algorithm was proposed which generates the Stackelberg-Nash solution for the problems considered.

The algorithm proposed here has provided reliable results, and was also able to find a better solution for one of the test-problems considered. In the computational experiments, three different parameter configurations were tested in order to analyze the performance of the proposed method. The results demonstrated that the algorithm was capable of finding good quality solutions when using any of the parameter configurations considered.

Although the proposed algorithm has been designed for the case of two followers, it can be applied to solve the Nash equilibrium with a general number of followers with minor adjustments. In such case, at each generation, the i-th follower optimizes its own decision variables considering the best values found by all other followers at the previous generation.

For future work, we intend to investigate and improve the proposed method by adopting a more efficient termination criteria, and a new scheme to obtain multiple solutions for both upper and lower levels of the problem.

Acknowledgment

The authors would like to thank the reviewers for their suggestions and comments, and the support from CNPq/PCI (grant 314054/2014-6) and CNPq (grant 310778/2013-1).

References

[1] H. V. Stackelberg, *The Theory of the Market Economy*. Oxford University Press, 1952.

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

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

[4] E. D´Amato, E. Daniele, L. Mallozzi, G. Petrone, and S. Tancredi, "A hierarchical multimodal hybrid Stackelberg-Nash GA for a leader with multiple followers game," in *Dynamics of Information Systems: Mathematical Foundations*. Springer, 2012, vol. 20, pp. 267–280.

[5] J. S. Angelo, E. Krempser, and H. J. C. Barbosa, "Differential evolution for bilevel programming," in *2013 IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp. 470–477.

[6] M. Sefrioui and J. Periaux, "Nash genetic algorithms: examples and applications," in *Proceedings of the 2000 Congress on Evolutionary Computation*, vol. 1, 2000, pp. 509–516.

[7] J. Lu, C. Shi, and G. Zhang, "On bilevel multi-follower decision making: General framework and solutions," *Information Sciences*, vol. 176, no. 11, pp. 1607 – 1627, 2006.

[8] H. Calvete and C. Galé, "Linear bilevel multi-follower programming with independent followers," *Journal of Global Optimization*, vol. 39, no. 3, pp. 409–417, 2007.

[9] C. Shi, G. Zhang, and J. Lu, "The kth-best approach for linear bilevel multi-follower programming," *Journal of Global Optimization*, vol. 33, no. 4, pp. 563–578, 2005.

[10] S. R. Arora and R. Narang, "0-1 bilevel fractional programming problem with independent followers," *International Journal of Optimization: Theory, Methods and Applications*, vol. 1, no. 2, 2009.

[11] B. Liu, "Stackelberg-Nash equilibrium for multilevel programming with multiple followers using genetic algorithms," *Computers & Mathematics with Applications*, vol. 36, no. 7, pp. 79 – 89, 1998.

[12] J. F. Wang and J. Periaux, "Multi-point optimization using gas and nash/stackelberg games for high lift multi-airfoil design in aerodynamics," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 1, 2001, pp. 552–559.

[13] J. Nash, "Non-Cooperative Games," *Annals of Mathematics*, vol. 54, no. 2, pp. 286–295, 1951.

[14] R. M. 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.

[15] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, pp. 311–338, 2000.

[16] Q. Wang, F. Yang, S. Wang, and L. Yi-Hsin, "Bilevel programs with multiple followers," *Journal of Systems Science and Complexity*, vol. 13, no. 3, p. 265, 2000.