Journal of Mathematics and Statistics 5 (4): 352-359, 2009
ISSN 1549-3644
© 2009 Science Publications

A Solution Methodology of Bi-Level Linear Programming Based on Genetic Algorithm

[1]M.S. Osman, [2]W.F. Abd El-Wahed, [3]Mervat M.K. El Shafei and [4]Hanaa B. Abd El Wahab
[1]Department of Basic Science, Higher Technological Institute, Ramadan 10th City, Egypt
[2]Department of Operations Researches and Decision Making,
Faculty of Computers and Information, El-Menoufia University, Shebin El-Kom, Egypt
[3]Department of Mathematics, Faculty of Science, Helwan University, Cairo, Egypt
[4]Department of Basic Science, Special Studies Academy, Cairo, Egypt

Abstract: Problem statement: We deal with the bi-level linear programming problem. A bi-level programming problem is formulated for a problem in which two Decision-Makers (DMs) make decisions successively. **Approach:** In this research we studied and designs a Genetic Algorithm (GA) of Bi-Level Linear Programming Problems (BLPP) by constructing the fitness function of the upper-level programming problems based on the definition of the feasible degree. This GA avoids the use of penalty function to deal with the constraints, by changing the randomly generated initial population into an initial population satisfying the constraints in order to improve the ability of the GA to deal with the constraints. Also we designed software to solve this problem. A comparative study between proposed method and previous methods through numerical results of some examples. Finally, parametric information of the GA was introduced. **Results:** Results of the study showed that the proposed method is feasible and more efficient to solve (BLPP), also there exist package to solve (BLPP) problem. **Conclusion:** This GA avoids the use of penalty function to deal with the constraints, by changing the randomly generated initial population into an initial population satisfying the constraints in order to improve the ability of the GA to deal with the constraints.

Key words: Bi-level linear programming problems, feasible degree, fitness function, genetic algorithm

INTRODUCTION

Multi-level programming techniques are developed to solve decentralized planning problems with multiple decision makers in a hierarchal organization[1]. The Bi-Level Programming (BLP) problem is a special case of multilevel programming problems with a two-level structure. This problem is an important case in non-convex optimization and a leader-follower game in which play is sequential and cooperation is not permitted[2]. A bi-level programming problem is formulated for a problem in which two Decision-Makers (DMs) make decisions successively. For example, in a decentralized firm, top management, an executive board, or headquarters makes a decision such as a budget of the firm and then each division determines a production plan in the full knowledge of the budget[3-12]. Also, the Stackelberg duopoly can be cited: two firms supply homogenous goods to a market. Suppose one firm dominates the other in the market and consequently the predominant firm decides its level of supply and then the other firm determines that of itself after it realizes that of the predominant firm. Stackelberg solution has been employed as a solution concept to two-level programming problems and a considerable number of algorithms for obtaining the solution have been developed [13]. Many instances of decision problems can be fined, which are formulated as two-level programming problems and concerning the above mentioned hierarchical decision problem in the decentralized firm, it is natural that decision makers behave cooperatively rather than non-cooperatively. Recently, Lai and Lee [14] have proposed a solution concept, which is different from the concept of the Stackelberg solution, for multi level linear programming problems such that decisions of DMs in both levels are sequential and all of the DMs essentially cooperate with each other[15,16].

Their method is based on the idea that the DM at the lower level optimizes the lower level objective function, taking a goal or preference of the upper level into consideration. DMs elicit membership functions of fuzzy goals for theirs objective functions and especially, the DM at the upper level also specifies

Corresponding Author: Mervat M.K. El Shafei, Department of Mathematics, Faculty of Science, Helwan University, Cairo, Egypt

those of fuzzy goals for decision variables. The DM at the lower level solves a fuzzy programming problem with constraints on fuzzy goals of the DM at the upper level. Unfortunately, there is possibility that their method leads to an undesirable final solution because of inconsistency between the fuzzy goals of the objective function and the decision variables.

In this study, we deal with the bi-level linear programming problem. First, we construct the fitness function of the upper-level programming problems based on the definition of the feasible degree. After that the genetic operators are developed for designing of the GA for BLPP[17]. Numerical examples are given to clarify the developed algorithm. Finally, parametric study of the GA is introduced. Parametric study of the mathematical programming problems is important, very necessary and enhances the scope of application of the obtained solutions of those problems. In other words, our problem can be restudied in different variations (parameters values) with no need to resolve it.

There are three main different approaches to handle the parametric optimization problem, namely the sensitive analysis approach that concerns the minor changes in the parameters values and its effects on the obtained solutions; the stability sets approach that deals with the stability of the optimal solutions in different cases and finally the parametric solution approach that studies the major variations of the parameters in the GA and its relations with the obtained solutions. In this study we shall exhibit and apply the last approach.

Problem formulation: Let us consider the following BLPP with the following form[18-22]:

$$P_1 : \max_{x_1} F(x_1, x_2) = c_1^T x_1 + c_2^T x_2 \tag{1}$$

where, x_2 solves:

$$P_2 : \max_{x_2} f(x_1, x_2) = d_1^T x_1 + d_2^T x_2 \tag{2}$$

subject to:

$$A_1 x_1 + A_2 x_2 \le b, x_1 \ge 0, x_2 \ge 0 \tag{3}$$

where, $A_1 \in R^{m\times n_1}$, $A_2 \in R^{m\times n_2}$, $c_1, d_1, x_1 \in R^{n_1}$, $c_2, d_2, x_2 \in R^{n_2}$, $b \in R^m$.

Once x_1 is fixed, the term $d_1^T x_1$ in the objective function of the lower-level problem is a constant. So the objective function of the lower-level problem is simply denoted as:

$$f = d_2^T x_2 \tag{4}$$

Let $X = \{(x_1, x_2) \mid A_1x_1 + A_2x_2 \le b\}$ denote the constraint region of BLPP.

Here, we assume X is nonempty and bounded.

Let $Q(x_1) = \{x_2 \mid A_2x_2 \le b - A_1x_1, x_2 \ge 0\}$ nonempty and bounded. Let $Y(x_1)$ denote the optimal solution set of problem:

$$\max_{x_2\in Q(x_1)} \bar{f} = d_2^T x_2 \tag{5}$$

We assume the element of the set $Y(x_1)$ exists and is unique and then the inducible region is:

$$\Psi_f(X) = \{(x_1, x_2) \mid (x_1, x_2) \in X, x_2 \in Y(x_1)\} \tag{6}$$

Definition 1: The point (x_1,x_2) is feasible if $(x_1,x_2)\in \Psi_f(X)$.

Definition 2: The feasible point $(x_1^*, x_2^*)\in \Psi_f(X)$ is the optimal solution of the BLPP if $F(x_1^*,x_2^*)\ge F(x_1,x_2)$ for each point $(x_1,x_2)\in \Psi_f(X)$.

Design of GA for BLPP: It is not easy to know the upper-level objective of BLPP has no explicit formulation, since it is compounded by the lower-level solution function which has no explicit formulation. Thus, it is hard to express the definition of the derivation of the function in common sense. Also it is difficult to discuss the conditions and the algorithms of the optimal solution with the definition. The GA is a numerical algorithm compatible for the optimization problem science it has no special requirements for the differentiability of the function. Hence the study solves BLPP by GA[2,17,22,23].

The basic idea for solving BLPP by GA is: firstly, choose the initial population satisfying the constraints, then the lower-level decision maker makes the corresponding optimal reaction and evaluate the individuals according to the fitness function constructed by the feasible degree, until the optimal solution is searched by the genetic operation over and over.

Coding and constraints: At present, the coding often used are binary vector coding and floating vector coding. But the floating vector coding is more near for the space of the problem compared with the former and the experiments show that it converges faster and has higher computing precision. This study adopts the floating vector coding. Hence the individual is expressed by:

$$v_k = (v_{k1}, v_{k2}, ..., v_{km}) \quad (7)$$

The individuals of the initial population are generally randomly generated in GA, which tends to generate off-springs who are not in the constraint region. Hence, we must deal with them.

Here, we deal with the constraints as follows: generate a group of individuals randomly and then retain the individuals satisfying the constraints in Eq. 3 as the initial population and drop out the ones not satisfying the constraints. The individuals generated by this way all satisfy the constraints and the off-springs satisfy the constraints by corresponding crossover and mutation operators.

Design of the fitness function: To solve the problem BLPP by GA, the definition of the feasible degree is firstly introduced and the fitness function is constructed to solve the problem by GA. Let d denote the large enough penalty interval of the feasible region for each $(x_1,x_2)\in X$.

Definition 3: Let $\theta\in[0,1]$ denotes the feasible degree of satisfying the feasible region and describe it by the following function:

$$\theta = \begin{cases} 1, & \text{if } \|x_2 - Y(x_1)\| = 0 \\ 1 - \left\| \dfrac{x_2 - Y(x_1)}{d} \right\|, & \text{if } 0 < \|x_2 - Y(x_1)\| \le d \\ 0, & \text{if } \|x_2 - Y(x_1)\| > d \end{cases} \quad (8)$$

Where, $\|.\|$ denotes the norm.

Furthers, the fitness function of the GA can be stated as:

$$\text{eval}(v_k) = (F(x_1,x_2) - F_{\min}) * \theta$$

where, F_{min} is the minimal value of $F(x_1,x_2)$ on X.

MATERIALS AND METHODS

Step 1: Initialization, give the population scale M, the maximal iteration generation MAXGEN and let the generation t = 0

Step 2: Initialization of the initial population, M individuals are randomly generated in X, making up the initial population.

Step 3: Computation of the fitness function. Evaluate the fitness value of the population according to formula (8)

Step 4: Selection, Select the individual by roulette wheel selection operator[24,25]

Step 5: Crossover, in this step, first, a random number $P_c\in[0,1]$ is generated. This number is the percentage of the population on which the crossover is performed. Then, two individuals are selected randomly from the population as parents. Children are generated using the following procedure:

Random integer c is generated in the interval [1,l-1], where l is the number of components of an individual. The c first components of the children are the same components as respective parents (i.e., the first child from the first parent and the second child from the second parent). The remaining components are selected according to the following rules:

- The (c+i)th component of the first child is replaced by the (l-i+1)th component of the second parent (for i = 1,2,….l-c)
- The (c-i)th component of the second child is replaced by the (l-i+1)th component of the first parent (for i = 1,2, …l-c)

For example we assume c = 5, we obtained the following children:

Parents	Children
10110 1100	10110 0100
11010 0010	11010 0011

Note that the proposed operator generates individuals with more variety in comparison with the standard operator, because this operator can generate different children from similar parents, where standard operators cannot.

Step 6: Mutation, In this step, first, a random number $P_m\in[0,1]$ is generated. This number is the percentage of the population on which the mutation is performed. Then one individual is selected randomly from the population. An integer random number u is generated in the interval [1,l], where l is the length of the individual. For generating the new individual, the uth component is changed to 0, if it was initially 1 and to 1 if it was initially 0 [11]

Step 7: Terminations, Jude the condition of the termination. When t is larger than the maximal iteration number, stop the GA and output the optimal solution. Otherwise, let t = t+1, turn to Step 3

RESULTS

Numerical example: In order to compare the performance of GA which introduced in this study with the other existing methods, proposed three examples solved by proposed method in this study and compare our results with the results in the previous methods[14,23].

Example 1:

$$\max_{x_1} F(x_1, x_2) = 2x_1 - x_2$$

where x^2 solve:

$$\max_{x_2} f(x_1, x_2) = x_1 + 2x_2$$

subject to:

$$\begin{aligned} &3x_1 - 5x_2 \le 15 \\ &3x_1 - x_2 \le 21 \\ &3x_1 + x_2 \le 27 \\ &3x_1 + 4x_2 \le 45 \\ &x_1 + 3x_2 \le 30 \\ &x_1, x_2 \ge 0 \end{aligned}$$

Example 2:

$$\max_{x_1} F(x_1, x_2) = x_1 + 3x_2$$

where x_2 solves:

$$\max_{x_2} f(x_1, x_2) = x_1 - 3x_2$$

subject to:

$$\begin{aligned} -x_1 - 2x_2 &\le -10 \\ x_1 - 2x_2 &\le 6 \\ 2x_1 - x_2 &\le 21 \\ -x_1 + 2x_2 &\le 18 \\ x_1 + 2x_2 &\le 38 \\ x_1, x_2 &\ge 0 \end{aligned}$$

Example 3:

$$\max_{x_1} F(x_1, x_2) = -8x_{11} - 4x_{12} + 4x_{21} - 40x_{22} - 4x_{23}$$

where, x_2 solves:

$$\max_{x_2} f(x_1, x_2) = x_{11} + 2x_{12} + x_{21} + x_{22} + 2x_{23}$$

subject to:

$$-x_{21} + x_{22} + x_{23} \le 1, 2x_{11} - x_{21} + 2x_{22} + 0.5x_{23} \le 1,$$
$$2x_{12} + 2x_{21} - x_{22} - 0.5x_{23} \le 1$$

The comparison of the results through 20 generations by the algorithm in the study and the results in the references are as follows:

From the numerical results in Table 1, the results by applying our algorithm is very efficient since we obtained the optimal solution in minimum time and minimum effort comparing with the previous methods as shown in Table 1 (this results obtain by using the package which designed in this study).

Parametric information of the GA: In this study we make some changes in the parameters (M, P_c, P_m) of the GA and study its relations with the obtained solutions as shown in Table 2.

Table 1: The comparison of our results and the results in the references for the used examples

	Parameters				Results in the paper			Results in the references			
Example	M	P_c	P_m	(x_1, x_2, θ)	F	f	Elapsed time (sec)	(x_1, x_2)	F	f	Elapsed time (min)
1	100	0.17	0.3	(7.26, 5.23, 0.17)	9.29	17.72	7.72	(7.26, 5.23)	9.29	17.72	20
2	100	0.17	0.3	(16, 11, 0.951)	49.00	-17.00	21.70	(15.95, 10.9)	49.00	-17.00	15
3	100	0.17	0.3	(0, 0.89, 0, 0.6, 0.39,0)	-29.17	3.18	2.65	(0, 0.9, 0, 0.6, 0.4)	-29.20	3.20	5

Notes: M: Denotes the population scale, P_c: Denotes crossover probability, P_m: Denotes mutation probability, F and f: Are the objective function value of the upper-level and the lower-level programming problem, respectively

Table 2: Parametric study of the GA for the used examples

	Parameters					
Example	M	P_c	P_m	Elapsed time (sec)	(x_1, x_2, θ)	Fig. No.
1	100	0.17	0.30	6.50	(7026, 5.23, 0.17)	1, 2
1	50	0.70	0.15	13.60	(7.26, 5.23, 0.34)	3,4
1	50	0.60	0.20	27.20	(7.26, 5.23, 1)	5, 6
1	100	0.80	0.30	23.42	(7.26, 5.23, 0.26)	
1	200	0.80	0.30	16.50	(7.26, 5.23, 1)	
2	100	0.17	0.30	27.50	(16, 11, 0.951)	7,8
2	50	0.70	0.15	7.69	(16, 11, 1)	9, 10
2	50	0.60	0.20	9.19	(16, 11, 0.43)	11, 12
2	100	0.80	0.30	17.48	(16, 11, 0.59)	
2	200	0.80	0.30	7.14	(16, 11, 0.47)	
3	100	0.17	0.30	9.56	(0, 0.89, 0, 0.6, 0.39, 0)	
3	50	0.70	0.15	11.97	(0, 0.89, 0, 0.6, 0.39, 0)	
3	50	0.60	0.20	4.44	(0, 0.89, 0, 0.6, 0.39, 0)	
3	100	0.80	0.30	2.65	(0, 0.89, 0, 0.6, 0.39,0)	
3	200	0.80	0.30	4.16	(0, 0.89, 0.6, 0.39, 0)	

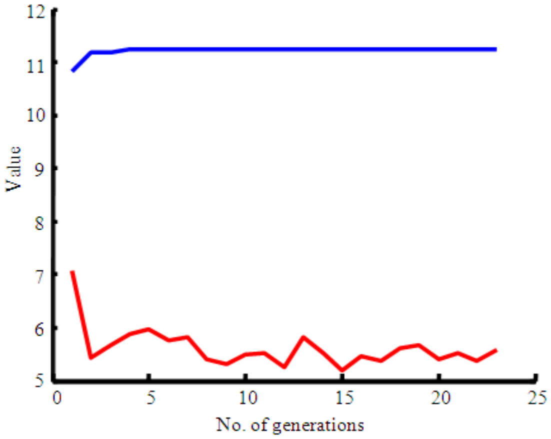

Fig. 1: Performance the best value and mean value of the GA for example 1

DISCUSSION

Here we discuss the result that we have acquired from the previous section. In example 1 (Fig. 1), according to the changes in the parameters of genetic algorithm where ($P_c = 0.17$, $P_m = 0.3$, $P_c = 0.7$, $P_m = 0.15$ and $P_c = 0.6$, $P_m = 0.2$), Note that when the number of generations are increased the best value of Genetic Algorithm (GA) nearly fixed and converges to 11.24 and the mean value of the GA is increased, decreased, increased, decreased and so on during the GA is running. Also in Fig. 2, the maximum value of the best solution for the GA is fixed in the beginning after that it is increased, fixed, increased and fixed in the final until reach to the maximum value of 11.24.

Note that the optimal solution for example 1 is the same for the figures from Fig. 3-6.

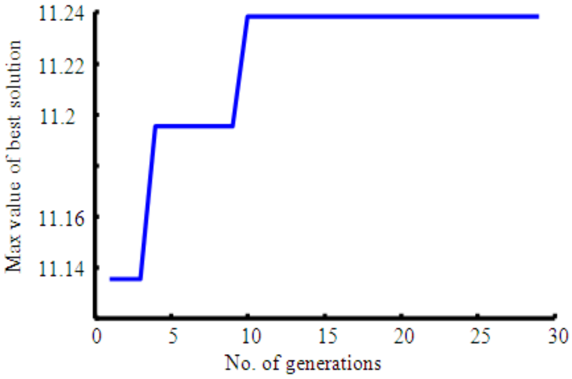

Fig. 2: Performance the maximum value of the best solution of the GA for example 1

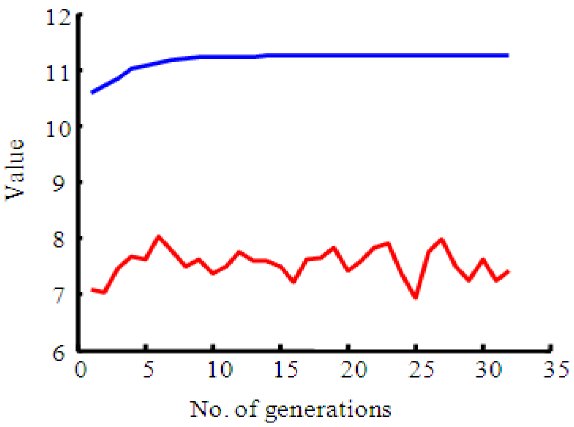

Fig. 3: Performance the best value and mean value of the GA for example 1

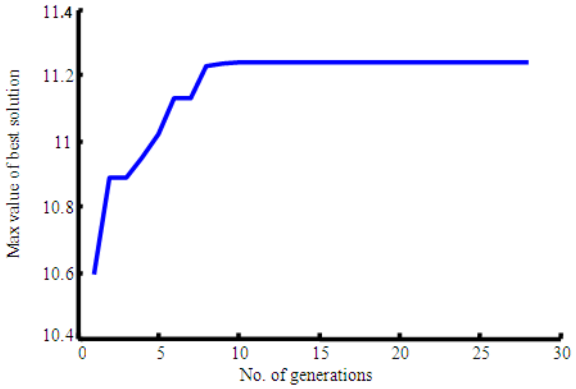

Fig. 4: Performance the maximum value of the best

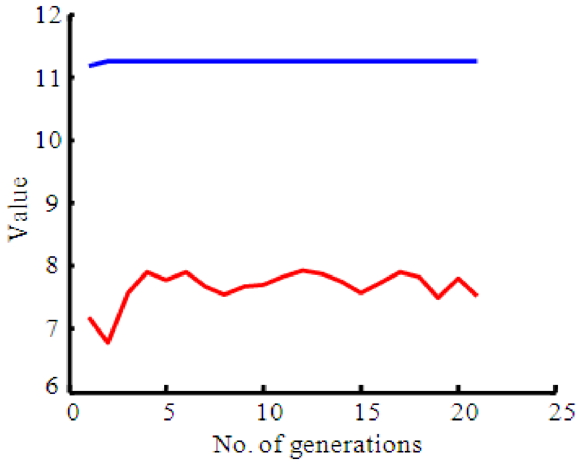

Fig. 5: Performance the best value and mean value of the GA for example 1

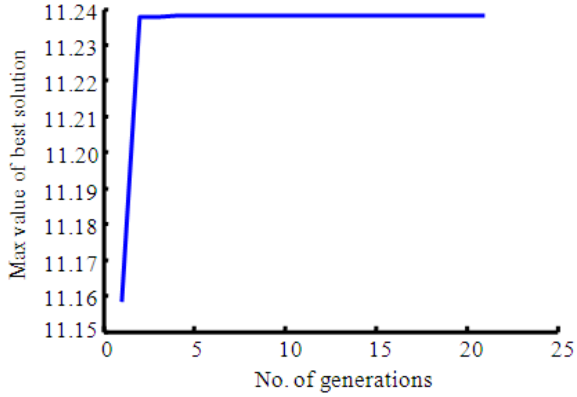

Fig. 6: Performance the maximum value of the best solution of the GA for example 1

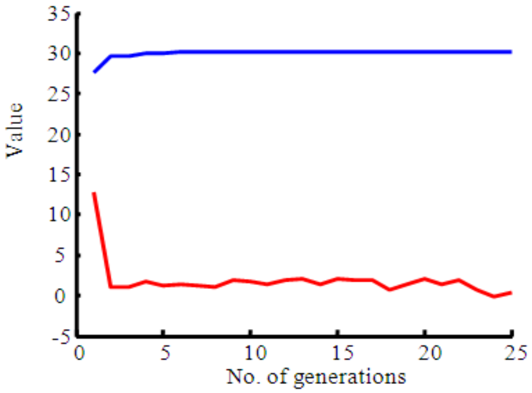

Fig. 7: Performance the best value and mean value of the GA for example 2

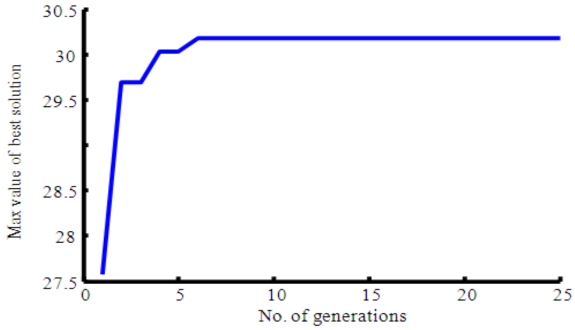

Fig. 8: Performance the maximum value of the best solution of the GA for example 2

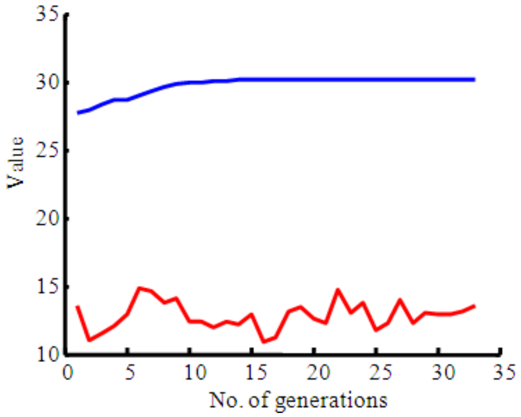

Fig. 9: Performance the best value and mean value of the GA for example 2

In example 2 (Fig. 7), Note that when the number of generations is increased the best value of Genetic Algorithm (GA) nearly fixed and converges to 30.18 and the mean value of the GA is increased, decreased, increased, decreased and so on during the GA is running.

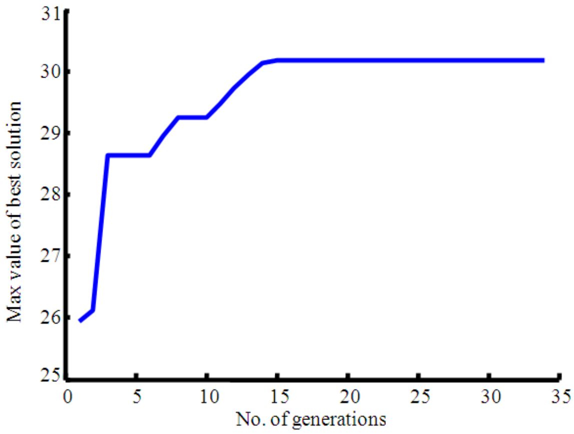

Fig. 10: Performance the maximum value of the best solution of the GA for example 2

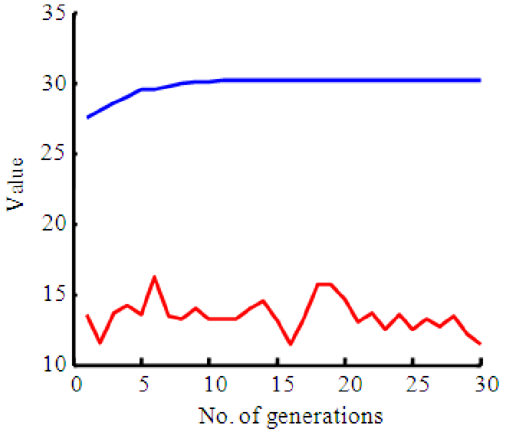

Fig. 11: Performance the best value and mean value of the GA for example 2

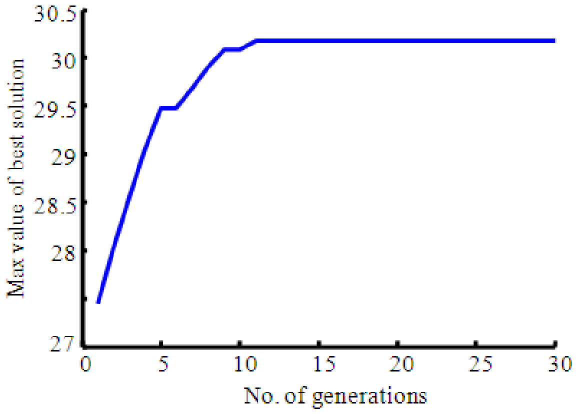

Fig. 12: Performance of the maximum value of the best solution of the GA for example 2

Also in Fig. 8, the maximum value of best solution of the GA is fixed in the beginning after that it is increased and fixed in the final until reach to the maximum value of 30.18.

Note that the optimal solution for example 2 is the same for the Fig. 9-12.

CONCLUSION

This study designs the GA for solving BLPP which the optimal solution of the lower-level problem is dependent on the upper-level problem. The numerical results show the method is feasible and efficient. Compared with the traditional methods, the method has the following characters:

- The method has no special requirement for the characters of the function and overcome the difficulty discussing the conditions and the algorithms of the optimal solution with the definition of the differentiability of the function
- This GA avoids the use of penalty function to deal with the constraints, by changing the randomly generated initial population into an initial population satisfying the constraints in order to improve the ability of the GA to deal with the constraints

Finally, we make some variations in the parameters of the GA and study its relations with the obtained solutions. Also we design a program to solve (BLPP) problem.

REFERENCES

1. Dempe, S. and T. Starostina, 2007. On the solution of fuzzy bilevel programming problems. http://www.optimization-online.org/DB_HTML/2007/09/1778.html
2. Langham, A.E. and P.W. Grant, 1999. A multilevel k-way portioning algorithm for finite element meshes using competing ant colonies. Proceedings of the Genetic and Evolutionary Computation Conference, July 13-17, San Francisco, CA., USA., pp: 1602-1608. ftp://ftp.cs.bham.ac.uk/pub/authors/W.B.Langdon/biblio/gp-html/langham_1999_AMPAFEMCAC.html
3. Dussault, J.P., P. Marcotte, S. Roch and G. Savard, 2003. A smoothing heuristic for the global optimization of a class of bilinear bilevel programs. http://www.iro.umontreal.ca/~marcotte/ARTIPS/Smoothing.pdf

4. Bard, J.F., 1983. An efficient point algorithm for a linear two-stage optimization problem. Operat. Res., 31: 670-684. http://or.journal.informs.org/cgi/content/abstract/31/4/670
5. Bard, J.F., 1983. Coordination of a multidivisional firm through two levels of management. Omega, 11: 457-468. http://ideas.repec.org/a/eee/jomega/v11y1983i5p457-468.html
6. Bard, J.F. and J.E. Falk, 1982. An explicit solution to the multi-level programming problems. Comput. Operat. Res., 9: 77-100.
7. Bard, J.F., J. Plummer and J.C. Sourie, 1998. Determining Tax Credits for Converting Nonfood Crops to Biofuels. In: Multilevel Optimization: Algorithms and Applications, Migdalas, A., P. Pardalos and P.V. Arbrand (Eds.). Springer, Dordrecht, ISBN: 0792346939, pp: 23-50.
8. Fortuny-Amat, J. and B. McCarl, 1981. A representation and economic interpretation of a two-level programming problem. J. Operat. Res. Soc., 32: 783-792. http://www.jstor.org/stable/2581394
9. Ben-Ayed, O., 1993. Bilevel linear programming. Com. Operat. Res., 20: 485-501. http://portal.acm.org/citation.cfm?id=159519
10. Dempe, S., 2003. Annotated bibliography on Bilevel programming and mathematical programs with equilibrium constraints. Optimization, 52: 333-359. DOI: 10.1080/0233193031000149894
11. Hejazi, S.R., A. Memariani, G. Jahanshahloo and M.M. Sepehri, 2002. Linear bilevel programming solution by genetic algorithm. Comput. Operat. Res., 29: 1913-1925. DOI: 10.1016/S0305-0548(01)00066-1
12. Bialas, W.F. and M.H. Karwan, 1984. Two-level linear programming. Manage. Sci., 30: 1004-1020. http://mansci.journal.informs.org/cgi/content/abstract/30/8/1004
13. Sakawa, M., I. Nishizaki and Y. Oka, 2000. 'Interactive fuzzy programming for multiobjective two-level linear programming problems with partial information of preference. Int. J. Fuzzy Syst., 2: 79-86. http://direct.bl.uk/bld/PlaceOrder.do?UIN=096911154&ETOC=RN&from=searchengine
14. Shih, H.S., Y.J. Lai and E.S. Lee, 1996. Fuzzy approach for multi-level programming problems. Comput. Operat. Res., 23: 73-91. http://portal.acm.org/citation.cfm?id=223639
15. Sakawa, M., I. Nishizaki and M. Hitaka, 2001. Interactive fuzzy programming for multi-level 0-1 programming problems with fuzzy parameters through genetic algorithms. Fuzzy Sets Syst., 117: 95-111. http://portal.acm.org/citation.cfm?id=370715.370734
16. Sakawa, M. and I. Nishizaki, 2001. Interactive fuzzy programming for two-level linear fractional programming problems. Fuzzy Sets Syst., 119: 31-40.
17. Abraham, A., L.C. Jain and R. Goldberg, 2005. Evolutionary Multi Objective Optimization: Theoretical Advances and Applications. 1st Edn., Springer, Berlin Heidelberg, ISBN: 1852337877, pp: 302.
18. B.Ozpineci , L.M. Tolbert and J.N. Chiassson, 'Harmonic optimization of multi-level converters using genetic algorithms' Proceeding of the IEEE 35th Annual Power Electronics Specialists Conference, June 20-25, 2004. IEEE Xplore Press, USA., pp. 3911-3916. DOI: 10.1109/PESC.2004.1355167

Ozpineci, B., L.M. Tolbert and J.N. Chiassson, 2004. Harmonic optimization of multi-level converters using genetic algorithms. Proceeding of the IEEE 35th Annual Power Electronics Specialists Conference, June 20-25, IEEE Xplore Press, USA., pp: 3911-3916. DOI: 10.1109/PESC.2004.1355167

20. Shimizu, K., Y. Ishizuka and J.F. Bard, 1997. Nondifferentiable and Two-Level Mathematical Programming. Springer, Boston, ISBN: 0792398211, pp: 470.
21. Vicente, L.N. and P.H. Calamai, 1994. Bilevel and multilevel programming a bibliography review. J. Global Optimiz., 5: 291-306. DOI: 10.1007/BF01096458
22. Marccote, P., 1986. Network design problem with congestion effects: A case of Bilevel programming. Math. Prog., 34: 142-162. DOI: 10.1007/BF01580580
23. Marcotte, P. and G. Savard, 1991. A note on the pareto optimality of solutions to the Bilevel programming problem. Comput. Operat. Res., 18: 355-359. http://portal.acm.org/citation.cfm?id=108286
24. Wen, U.P. and S.T. Hsu, 1991. Linear Bilevel programming problems-- a review. J. Operat. Res. Soc., 42: 125-133. http://www.jstor.org/stable/2583177
25. Van den Bergh, F. and A.P. Engelbrecht, 2002. An analysis of particle swarm optimizers. A Ph.D. Thesis, Pretoria University, Pretoria. http://portal.acm.org/citation.cfm?id=935867
26. Michalewicz, Z., 1996. Genetic Algorithms +Data Structures = Evolution Programs. 3rd Edn., Springer-Verlag, London UK., ISBN: 3540606769, pp: 387.