INTERACTIVE FUZZY PROGRAMMING FOR MULTI-LEVEL NONLINEAR INTEGER PROGRAMMING PROBLEMS THROUGH GENETIC ALGORITHMS

M.A.K. Azad, M. Sakawa, K. Kato and H. Katagiri

Received March 2, 2005; revised January 10, 2006

Abstract. In the present paper, we focus on multi-level nonlinear integer programming problems (MLNLIPPs) in which there exist decision makers (DMs) with integer decision variables at all levels. Various approaches for multi-level programming problems could exist according to situations which the DMs are placed in. Assuming the cooperative relationship among the DMs at all levels, in this paper, an interactive fuzzy programming technique through genetic algorithms for MLNLIPPs is proposed to obtain a satisfactory solution for all DMs. Furthermore, the feasibility of the proposed method is shown by applying it to an illustrative numerical example.

1 Introduction In the real world, we often encounter decision making situations involving multiple decision makers (DMs). Especially in industrial or governmental decision making situations, those DMs have different interest and decision priority. A multi-level programming problem is one of mathematical optimization models for them. In this paper, we focus on multi-level nonlinear integer programming problems (MLNLIPPs) in which there exist a DM with integer decision variables at each level.

Various approaches for multi-level programming problems could exist according to situations which the DMs are placed in. Under the assumption that the DMs do not have motivation to cooperate mutually, a Stackelberg solutions is adopted as a reasonable solution for the situation. It is assumed that the DM at the upper level (leader) and the DM at the lower level (follower) completely know their objective functions and the constraints of the problem and they do not have any motivation to cooperate with each other, and the leader first makes a decision and then the follower specifies a decision so as to optimize the objective function of itself with full knowledge of the decision of the leader. Under this assumption, the leader also makes a decision such that the objective function of the leader is optimized. Then, a solution defined as the above mentioned procedure is called a Stackelberg (equilibrium) solutions, which has been employed as a solution concept for two-level mathematical programming problems [4, 6, 8, 9, 16].

Bialas and Karwan [4] proposed four algorithms based on vertex enumeration and Kuhn-Tucker approches to solve two-level linear programming problems for obtaining Stackelberg solutions where two of them can provide local optimal solutions, and remaining two yield global optima. It is known that finding Stackelberg solutions of the two-level programming problems is strongly NP-hard [16], i.e., in proportion as scale of the problem, computational time exceedingly increases. To reduce computational time, Nishizaki et al. [8, 9] proposed computational methods through genetic algorithms for obtaining Stackelberg solutions to two-level zero-one and mixed zero-one linear programming problems. Jan et al. [6] proposed a solution method for obtaining Stackelberg solutions to nonlinear integer

2003 *Mathematics Subject Classification.* Primary 90C15, 90C59; Secondary 90C10, 90C29.

Key words and phrases. multi-level nonlinear integer programming problem, interactive fuzzy programming, genetic algorithm.

bilevel programming problems. For obtaining Stackelberg solutions to multi-level linear programming problems, Bard [2] and Wen and Bialas [17] proposed algorithms for three-level programming problems. Bard [2] proposes a cutting plane algorithm employing a vertex search procedure to solve three-level linear programming problems. Wen and Bialas [17] develop a hybrid algorithm based on K^{th} best algorithm to solve three-level linear programming problems.

On the other hand, under the assumption that the DMs have motivation to cooperate mutually, a satisfactory solution for all the DMs is adopted as a reasonable solution for the situation. For obtaining a satisfactory solution, Lai [7] and Shih, Lai and Lee [15] proposed solution methods based on fuzzy concepts for multi-level linear programming problems such that decisions of DMs in all levels are sequential and all of the DMs essentially cooperate with each other. In their method, the DMs identify membership functions of fuzzy goals for their objective functions, and especially, the DMs at the upper levels also specifies those of fuzzy goals for decision variables. The DM at the lowest level solves a fuzzy programming problem with constraints on fuzzy goals of the DMs at the upper levels. Unfortunately, however, there is a possibility that their method leads a final solution to an undesirable one because of inconsistency between the fuzzy goals of the objective function and those of the decision variables. To overcome the problem in the method of Shih et al., eliminating the fuzzy goals for the decision variables, Sakawa et al. have proposed interactive fuzzy programming for multi-level linear programming problems [12, 13, 14]. But they considered only a linear case. For obtaining a satisfactory solution to two-level nonlinear integer programming problems the authors [1] proposed an interactive fuzzy programming technique through genetic algorithms with double strings using continuous relaxation based on reference solution updating (GADSCRRSU) [10, 11].

Multi-level nonlinear integer programming problems (MLNLIPPs) can be formulated as large-scale mathematical programming problems involving integer decision variables, nonlinear objective functions and nonlinear constraint functions. Since a general solution method does not exist for nonlinear integer programming problems like the branch and bound method for linear ones, a solution method peculiar to each problem has been proposed. As a general-purpose solution method for nonlinear integer programming problems, we propose the usage of genetic algorithms with double strings using continuous relaxation based on reference solution updating (GADSCRRSU) [10, 11].

Under these circumstances, in this paper, for obtaining a satisfactory solution in cooperative relationship among the DMs, interactive fuzzy programming through proposed GADSCRRSU is presented for MLNLIPPs. Furthermore, the feasibility of the proposed method is shown through application of it to illustrative numerical example with different numbers of variables.

2 Problem Formulation

Multi-level nonlinear integer programming problems (MLNLIPPs) with t levels are generally formulated as follows:

$$(1)\qquad \begin{array}{rl}
\underset{\mathrm{DM1}}{\text{minimize}} & f_1(\boldsymbol{x}_1,\dots,\boldsymbol{x}_t)\\
\underset{\mathrm{DM2}}{\text{minimize}} & f_2(\boldsymbol{x}_1,\dots,\boldsymbol{x}_t)\\
\cdots\ \cdots & \cdots\ \cdots\\
\underset{\mathrm{DM}l}{\text{minimize}} & f_l(\boldsymbol{x}_1,\dots,\boldsymbol{x}_t)\\
\cdots\ \cdots & \cdots\ \cdots\\
\underset{\mathrm{DM}t}{\text{minimize}} & f_t(\boldsymbol{x}_1,\dots,\boldsymbol{x}_t)\\
\text{subject to} & g_i(\boldsymbol{x}_1,\dots,\boldsymbol{x}_t)\le 0,\ i=1,\dots,m\\
 & x_{lj}\in\{0,1,\dots,\nu_{lj}\},\ l=1,\dots,t,\ j=1,\dots,n_l,
\end{array}$$

where $\boldsymbol{x}_l$ is an n_l dimensional integer decision variable column vector for the decision maker at each level, $f_l(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t)$, $l = 1, \ldots, t$ and $g_i(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t)$, $i = 1, \ldots, m$ may be linear or nonlinear. For notational convenience, we use $\boldsymbol{x} = (\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_t^T)^T$ and denote the feasible region of problem (1) by X. Since the DMs have motivation to cooperate with each other, we denote the solution vector as $\boldsymbol{x}$ without partition.

For example, consider a project selection problem in an administrative office at the upper level and several autonomous divisions of a company. In this case, the situation that all the DMs can cooperate with each other seems natural rather than one that all the DMs do not have motivation to cooperate mutually.

Under the hypothesis of cooperation among all DMs, Sakawa et al. [12, 13, 14] proposed interactive fuzzy programming for multi-level linear programming problems in order to derive satisfactory solutions for the DMs through interactions with the DM at the upper level by introducing fuzzy goals to consider the imprecise nature of DMs' judgements for objective functions.

In this paper, for MLNLIPPs, focusing on the case of cooperative relation among all DMs, we present a new interactive fuzzy programming method through genetic algorithms with double strings using continuous relaxation based on reference solution updating (GADSCRRSU) in order to derive a satisfactory solution for the DMs.

3 Interactive Fuzzy Programming In this section, we describe a new interactive fuzzy programming method through genetic algorithms based on literatures by Sakawa et al. [12, 13, 14] is summarized as follows.

3.1 Interactive Fuzzy Programming Considering the ambiguity or fuzziness of the decision makers' judgements on each of the objective functions $f_l(\boldsymbol{x})$, $l = 1, \ldots, t$ in (1), it seems natural to introduce such fuzzy goals for objective functions as "$f_l(\boldsymbol{x})$ should be subjectively less than or equal to a certain value". First, we solve problems to obtain the individual minimum

$$f_l^{\min} = \min_{\boldsymbol{x} \in X} \ f_l(\boldsymbol{x}),\ l = 1, \ldots, t \tag{2}$$

and the individual maximum

$$f_l^{\max} = \max_{\boldsymbol{x} \in X} \ f_l(\boldsymbol{x}),\ l = 1, \ldots, t \tag{3}$$

of each of the objective functions which are referred to when the DMs elicit membership functions prescribing the fuzzy goals for the objective functions $f_l(\boldsymbol{x})$, $l = 1, \ldots, t$. Since these problems are single-objective nonlinear integer programming problems and it is difficult to obtain optimal solutions to them, we use genetic algorithms with double strings using continuous relaxation based on reference solution updating (GADSCRRSU) which is an extension of genetic algorithms with double strings based on reference solution updating (GADSRSU) for linear 0-1 programming problems [11].

The DMs determine the membership functions $\mu_l(f_l(\boldsymbol{x}))$, $l = 1, \ldots, t$ which are strictly monotone decreasing for $f_l(\boldsymbol{x})$, consulting the variation ratio of degree of satisfaction in the interval between the individual minimum of problem (2) and the individual maximum of problem (3). The domain of the membership functions is in the interval $[f_l^{\min}, f_l^{\max}]$, $l = 1, \ldots, t$ and the DM specifies the value f_l^0 of the objective function for which the degree of satisfaction is 0 and the value f_l^1 of the objective function for which the degree of satisfaction is 1. For the value undesired (larger) than f_l^0, it is defined that $\mu_l(f_l(\boldsymbol{x})) = 0$, and for the value desired (smaller) than f_l^1, it is defined that $\mu_l(f_l(\boldsymbol{x})) = 1$. Here a linear membership function in Fig. 1 is considered, which characterizes the fuzzy goal of the DM at each level. The corresponding linear membership function $\mu_l(f_l(\boldsymbol{x}))$ is defined as:

$$\mu_l(f_l(\boldsymbol{x})) = \begin{cases} 1 & f_l(\boldsymbol{x}) < f_l^1 \\ \dfrac{f_l(\boldsymbol{x}) - f_l^0}{f_l^1 - f_l^0} & f_l^1 \leq f_l(\boldsymbol{x}) < f_l^0 \\ 0 & f_l(\boldsymbol{x}) \geq f_l^0 \end{cases} \tag{4}$$

It is assumed that the DMs subjectively specify f_l^0 and f_l^1.

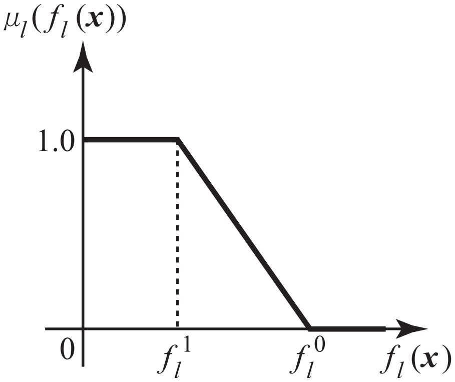


Figure 1: Linear Membership Function

Zimmermann [18] proposed a method for determining the parameters f_l^0 and f_l^1 of the linear membership function in the following way. That is, using the individual minimum, they are defined as

$$f_l^1 = f_l^{\min} = f_l(\boldsymbol{x}^{l0}) = \min_{\boldsymbol{x} \in X} \ f_l(\boldsymbol{x}), \ l = 1, \ldots, t \tag{5}$$

together with

$$f_l^0 = \max \left(f_l(\boldsymbol{x}^{10}), \ldots, f_l(\boldsymbol{x}^{(l-1)0}), f_l(\boldsymbol{x}^{(l+1)0}), \ldots, f_l(\boldsymbol{x}^{t0}) \right). \tag{6}$$

Having elicited membership functions $\mu_l(f_l(\boldsymbol{x}))$ for $f_l(\boldsymbol{x})$, $l = 1, \ldots, t$ by the DM at each level, then the original multi-level nonlinear integer programming problems (1) can be interpreted as a multi-level membership maximization problem defined by:

$$\begin{array}{rl} \underset{\text{DM1}}{\text{maximize}} & \mu_1(f_1(\boldsymbol{x})) \\ \underset{\text{DM2}}{\text{maximize}} & \mu_2(f_2(\boldsymbol{x})) \\ \cdots\ \cdots & \cdots\ \cdots \\ \underset{\text{DM}l}{\text{maximize}} & \mu_l(f_l(\boldsymbol{x})) \\ \cdots\ \cdots & \cdots\ \cdots \\ \underset{\text{DM}t}{\text{maximize}} & \mu_t(f_t(\boldsymbol{x})) \\ \text{subject to} & g_i(\boldsymbol{x}) \leq 0, \ i = 1, \ldots, m \\ & x_{lj} \in \{0, 1, \ldots, \nu_{lj}\}, \ l = 1, \ldots, t, \ j = 1, \ldots, n_l. \end{array} \tag{7}$$

Since (7) is a multi-level membership maximization problem, in general, a complete optimal solution that simultaneously maximizes all the DMs' degree of satisfaction of their objective functions does not always exist when the objective functions conflict with each other. Thus, a satisfactory solution is expected to be obtained from among M-Pareto optimal solution set which is defined for multiobjective programming problems [10, 11].

For deriving an overall satisfactory solution to the formulated problem (7), first the maximizing decision of the fuzzy decision proposed by Bellman and Zadeh [3] is found. Namely, the following problem is solved for obtaining a solution which maximizes the smallest degree of satisfaction among the t DMs:

$$\begin{array}{lll} & \text{maximize} & \min_{l=1,\dots,t} \mu_l(f_l(\boldsymbol{x})) \\ (8) & \text{subject to} & g_i(\boldsymbol{x}) \le 0,\ i=1,\dots,m \\ & & x_{lj} \in \{0,1,\dots,\nu_{lj}\},\ l=1,\dots,t,\ j=1,\dots,n_l. \end{array}$$

In order to guarantee the M-Pareto optimality of the obtained solution, the following augmented maximin problem (9) instead of the maximin problem (8) is solved.

$$\text{(9)} \qquad \underset{\boldsymbol{x}\in X}{\text{maximize}} \ \min_{l=1,\dots,t} \left\{ \mu_l(f_l(\boldsymbol{x})) + \rho \sum_{q=1}^{t} \mu_q(f_q(\boldsymbol{x})) \right\}.$$

The term augmented is adopted here because the term $\rho \sum_{q=1}^{t} \mu_q(f_q(\boldsymbol{x}))$ is added to the standard maximin problem (8), where ρ is a sufficiently small positive number. By solving problem (9), we can obtain a solution which maximizes the smallest degree of satisfaction among all of the t DMs. This problem can also be solved by GADSCRRSU. Let us denote an optimal solution to the problem (9) by $\boldsymbol{x}^*$. Then, we define the satisfactory degree of the t DMs under the constraints as

$$\text{(10)} \qquad \lambda = \min\{\mu_1(f_1(\boldsymbol{x}^*)), \dots, \mu_t(f_t(\boldsymbol{x}^*))\}$$

and the ratio of satisfactory degrees between adjacent two levels as

$$\text{(11)} \qquad \Delta_l = \frac{\mu_{l+1}(f_{l+1}(\boldsymbol{x}))}{\mu_l(f_l(\boldsymbol{x}))},\ l=1,\dots,t-1.$$

If the optimal solution $\boldsymbol{x}^*$ to problem (9) cannot satisfy the DMs, problems concerning a part of the DMs are solved one after another from the last two levels in order to coordinate the satisfactory degrees of the DMs and finally obtain a satisfactory solution of all the DMs.

At an interaction k, let $\mu_l(f_l^k)$ and λ^k denote a satisfactory degree of DMl, $l=1,\dots,t$ and a satisfactory degree of all the levels, respectively, and let $\Delta_l^k = \mu_{l+1}(f_{l+1}^k)/\mu_l(f_l^k)$ denote a ratio of satisfactory degrees of the lth and the $(l+1)$th levels. Let $\boldsymbol{x}^k$ denote a solution at the interaction k. For all $l=1,\dots,t$, DMl is proposed a solution by DM$(l+1)$. Then the DMs at all the levels except for the tth level obtain the satisfactory solutions and the interactive process terminates if the following two conditions are satisfied.

Termination conditions of the interactive process:

Condition 1: For all $l=1,\dots,t-1$, DMl's satisfactory degree is larger than or equal to the minimal satisfactory level $\hat{\delta}_l$ specified by DMl, i.e., $\mu_l(f_l^k) \ge \hat{\delta}_l$, $l=1,\dots,t-1$.

Condition 2: For all $l=1,\dots,t-1$ the ratio of Δ_l^k of satisfactory degrees lies in the interval between the lower and the upper bounds specified by DMl, i.e., $\Delta_l^k \in [\Delta_l^{\min}, \Delta_l^{\max}]$.

Condition 1 is DMl's required condition for solutions proposed by DM$(l+1)$, and condition 2 is provided in order to keep overall satisfactory balance among all the levels.

Unless the conditions are satisfied simultaneously, some of DMl, $l=1,\dots,t-1$ needs to update his minimal satisfactory level $\hat{\delta}_l$. Suppose that the DMs from at the $(p+1)$th level to at the $(t-1)$th level, i.e., DM$(p+1)$, DM$(p+2), \ldots,$ DM$(t-1)$ are satisfied with the proposed solution but DMp is not satisfied with it. Then DMp, DM$(p+1), \ldots,$ and DM$(t-1)$ need to update their minimal satisfactory levels $\hat{\delta}_l$, $l = p, p+1, \ldots, t-1$. For any two levels adjacent to each other, giving the DM at an upper level serious consideration, the DM at a lower level should update his minamal satisfatory level.

Procedure for updating minimal satisfactory level $\hat{\delta}_l$:

Case 1: If the condition 1 with respect to a DM at a level is not satisfied, then the DM decreases the minimal satisfactory level $\hat{\delta}_l$.

Case 2: If the ratio Δ_l^k exceeds its upper bound, then DMl increases the minimal satisfactory level $\hat{\delta}_l$. Conversely, if Δ_l^k is less than its lower bound, then DMl decreases the minimal satisfactory level $\hat{\delta}_l$.

Case 3: Although conditions 1 and 2 are satisfied, if a DM at a level is not satisfied with the obtained solution and judges that it is desirable to increase the satisfactory degree of the DM at the expense of the satisfactory degree of the DM at his lower level, then the first DM increases the minimal satisfactory level $\hat{\delta}_l$ and vice versa.

Let $\hat{\delta}'_l$, $l = p, \ldots, t-1$, denote the updated minimal satisfactory level, then the following problem (12) is solved which maximizes the smallest degree of satisfaction among the DMs from 1 to $p-1$ and tth levels.

$$\text{(12)} \qquad \begin{array}{ll} \text{maximize} & \displaystyle\min_{l=1,\ldots,p,t} \left\{ \mu_l(f_l(\boldsymbol{x})) + \rho \sum_{q=1,\ldots,p,\,t} \mu_q(f_q(\boldsymbol{x})) \right\} \\ \text{subject to} & g_i(\boldsymbol{x}) \le 0,\ i = 1, \ldots, m \\ & \mu_l(f_l(\boldsymbol{x})) \ge \hat{\delta}'_l,\ l = p, \ldots, t-1 \\ & x_{lj} \in \{0, 1, \ldots, \nu_{lj}\},\ l = 1, \ldots, t,\ j = 1, \ldots, n_l \end{array}$$

It should be noted that GADSCRRSU is applicable to (12).

3.2 Algorithm of the Interactive Fuzzy Programming through GADSCRRSU

We are now ready to present an interactive algorithm for deriving an overall satisfactory solution to multi-level nonlinear integer programming problems (1) through genetic algorithms with double strings using continuous relaxation based on reference solution updating (GADSCRRSU), which is summarized in the following.

Step 1: Solve (5) through GADSCRRSU for individual minimum and by using equation (6) calculate f_l^0 for each objective function of all the DMs and ask the DMs to identify their membership functions $\mu_l(f_\cdot)$, $l = 1, \ldots, t$ of the fuzzy goals for their own objective functions. Also ask the DMl, $l = 1, \ldots, t-1$ to specify subjectively the lower and the upper bounds of the ratio of satisfactory degrees Δ_l.

Step 2: Set interaction $k := 1$ and solve the problem (9) through GADSCRSSU, in which the smallest degree among all the DMs is maximized. The solution is proposed to all the DMs except the tth level.

Step 3: If the solution proposed to DMs at all the levels except for the tth level satisfies the termination conditions, they conclude the solution as a satisfactory one and the algorithm stops. Otherwise, let $k := k+1$ and go to step 4.

Step 4: If the DMs from at the $(p+1)$th level to at the $(t-1)$th level, i.e., DM$(p+1)$, DM$(p+2), \ldots,$ and DM$(t-1)$ is satisfied with the proposed solution but DMp is not satisfied with it, DMl, $l = p, \ldots, t-1$ update the minimal satisfactory levels $\hat{\delta}_l$, $l = p, \ldots, t-1$ in according to the procedure of updating minimal satisfactory level.

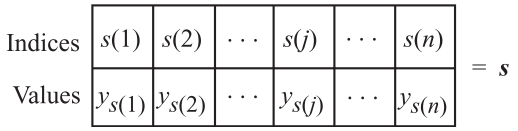
Step 5: Solve (12) through GADSCRRSU, in which smallest degree of satisfaction among the DMs from 1 to $p-1$ and tth levels is maximized and propose the solution to the DMs from 1 to $p-1$ levels. Return to step 3.

4 Genetic Algorithms with Double Strings using Continuous Relaxation based on Reference Solution Updating (GADSCRRSU) In this section, we mention GADSCRRSU proposed as a general solution method for nonlinear integer programming problems defined as

$$\begin{array}{rl} \text{minimize} & f(\boldsymbol{x}) \\ \text{subject to} & g_i(\boldsymbol{x}) \leq 0, \;\; i = 1, \ldots, m \\ & x_j \in \{0, 1, \ldots, \nu_j\}, \; j = 1, \ldots, n. \end{array} \tag{13}$$

In (13), $\boldsymbol{x}$ is an n dimensional integer decision variable vector, $f(\boldsymbol{x})$, $g_i(\boldsymbol{x})$, $i = 1, \ldots, m$ are nonlinear functions and ν_j, $j = 1, \ldots, n$ is the upper bound of each decision variable.

4.1 Individual Representation The individual representation [10, 11] by double strings shown in Figure 2 is adopted in GADSCRRSU. In the figure, each of $s(j)$, $j = 1, \ldots, n$ is the index of an element in a solution vector and each of $y_{s(j)} \in \{0, 1, \ldots, \nu_j\}$, $j = 1, \ldots, n$ is the value of the element, respectively.



Indices	$s(1)$	$s(2)$	$\cdots$	$s(j)$	$\cdots$	$s(n)$	$= \boldsymbol{s}$
Values	$y_{s(1)}$	$y_{s(2)}$	$\cdots$	$y_{s(j)}$	$\cdots$	$y_{s(n)}$	

Figure 2: Double Strings Representation

4.2 Decoding Algorithm Let N be the total number of population (*pop_size*). The individuals $\boldsymbol{s}$ with the dimensions n are generated randomly. Unfortunately, however, the direct mapping of the individual can generate infeasible solutions [10, 11]. To eliminate such solutions, a decoding algorithm of double strings for nonlinear integer programming problems (13) using a reference solution $\boldsymbol{x}^0$, which is a feasible solution and used as the origin of decoding, is constructed as follows.

Decoding algorithm using reference solution:

In the algorithm, it is assumed that a feasible solution $\boldsymbol{x}^0$ is obtained in advance. Let n, and N be the number of variables and number of individuals in the population, respectively.

Step 1: If the index of an individual to be decoded is in $\{1, \ldots, \lfloor N/2 \rfloor\}$, go to step 2. Otherwise, go to step 8.

Step 2: Let $j := 1$, $\boldsymbol{x} := \{0, \ldots, 0\}$, $l := 1$.

Step 3: Let $x_{s(j)} := y_{s(j)}$.

Step 4: If $g_i(\boldsymbol{x}) \leq 0$, $i = 1, \ldots, m$, let $l := j$, $j := j+1$, and go to step 5. Otherwise, let $j := j+1$, and go to step 5.

Step 5: If $j \leq n$, go to step 3. Otherwise, go to step 6.

Step 6: If $l > 0$, go to step 7. Otherwise, go to step 8.

Step 7: By substituting $x_{s(j)} := y_{s(j)}$, $1 \leq j \leq l$ and $x_{s(j)} := 0$, $l < j \leq n$, we obtain a feasible solution $\boldsymbol{x}$ corresponding to the individual $\boldsymbol{s}$ and stop.

Step 8: Let $j := 1$, $\boldsymbol{x} := \boldsymbol{x}^0$.

Step 9: Let $x_{s(j)} := y_{s(j)}$. If $y_{s(j)} = x^0_{s(j)}$, let $j := j+1$, and go to step 11. If $y_{s(j)} \neq x^0_{s(j)}$, go to step 10.

Step 10: If $g_i(\boldsymbol{x}) \leq 0$, $i = 1, \ldots, m$, let $j := j+1$, and go to step 11. Otherwise, let $\boldsymbol{x}_{s(j)} := \boldsymbol{x}^0_{s(j)}$, $j := j+1$, and go to step 11.

Step 11: If $j \leq n$, go to step 9. Otherwise, we obtain a feasible solution $\boldsymbol{x}$ corresponding to the individual $\boldsymbol{s}$ and stop.

This decoding algorithm enables us to decode each of the individuals represented by the double strings to the corresponding feasible solution. However, the diversity of the solution $\boldsymbol{x}$ greatly depends on the reference solution, because solutions obtained by the decoding algorithm using reference solution tend to concentrate around the reference solution. To overcome such situations, the reference solution updating procedure [10, 11] is adopted here.

4.3 Fitness Nature obeys the principle of Darwinian "survival of the fittest", the individuals with high fitness values will, on average, reproduce more often than those low fitness values. For obtaining satisfactory solution for all the DMs to multi-level nonlinear integer programming problems (1) through GADSCRRSU, the objective function value is used as the fitness value f of an individual $\boldsymbol{s}$. When the variance of fitness in a population is small, it is often observed that the ordinary roulette wheel selection does not work well because there is little difference between the probability of a good individual surviving and that of a bad one surviving [10, 11]. In order to overcome this problem, the linear scaling [10, 11] is adopted here. The new fitness $f'_l(\boldsymbol{s})$, $l = 1, \ldots, t$ of the DMl is obtained by using the following linear scaling

$$f'_l(\boldsymbol{s}) := a_l f_l(\boldsymbol{s}) + b_l \tag{14}$$

where $f_l(\boldsymbol{s})$, $l = 1, \ldots, t$ are the fitness values of the DMs at all levels with respect to each decoded individual $\boldsymbol{s}$.

4.4 Genetic Operators For obtaining satisfactory solution for all the DMs to multi-level nonlinear integer programming problems (1) through GADSCRRSU, four genetic operators such as reproduction, partially matched crossover (PMX), bit reverse mutation and inversion [10, 11] are adopted here.

4.5 Usage of Continuous Relaxation In order to find an approximate optimal solution with high accuracy in reasonable time, we need some schemes such as the restriction of the search space to a promising region, the generation of individuals near the optimal solution and so forth. From the point of view, the information about an optimal solution to the corresponding continuous relaxation problem

$$\begin{array}{rl} \text{minimize} & f(\boldsymbol{x}) \\ \text{subject to} & g_i(\boldsymbol{x}) \leq 0,\ i = 1, \ldots, m \\ & 0 \leq x_j \leq \nu_j,\ j = 1, \ldots, n \end{array} \tag{15}$$

is used in the generation of the initial population and the bit reverse mutation. When this problem is convex, we can obtain a global optimal solution by some convex programming technique, e.g., the sequential quadratic programming. Otherwise, i.e., when it is nonconvex, because it is difficult to find a global optimal solution, we search an approximate optimal solution by some approximate solution method such as genetic algorithms or simulated annealing. Here GENOCOP V [5] is used to find the solution of corresponding continuous relaxation problem (15).

4.6 Computational Procedures of GADSCRRSU Now the genetic algorithms with double strings using continuous relaxation based on reference solution updating (GADSCRRSU) for solving nonlinear integer programming problems (13) are summarized in the following.

Step 0: Determine values of the parameters used in GADSCRRSU: the population size N, the minimal search generation $I_{\min}$, the maximal search generation $I_{\max} > I_{\min}$, the convergence criterion ϵ, the degree of use of information about solutions to nonlinear programming relaxation problem R, the parameter for feasible solution θ, the parameter for reference solution updating η, the parameter for augmented maxmin problem ρ, the upper bound of each decision variable ν, the scaling constant c_{mult}, the probability of crossover p_c, the generation gap G, the probability of mutation p_m, the probability of inversion p_i and set the generation counter r at 0.

Step 1: Generate the initial population consisting of N individuals based on the information of a solution to the continuous relaxation problem (15).

Step 2: Decode each individual (genotype) in the current population and calculate its fitness based on the corresponding solution (phenotype).

Step 3: If the termination condition is fulfilled, stop. Otherwise, let $r := r + 1$ and go to step 4.

Step 4: Apply the reproduction operator based on the elitist expected value selection, after carrying out linear scaling.

Step 5: Apply the crossover operator, called PMX (Partially Matched Crossover) for double strings.

Step 6: Apply the mutation operator based on the information of an optimal solution to the continuous relaxation problem (15).

Step 7: Apply the inversion operator. Go to step 2.

5 Numerical Example Here, we consider the following multiobjective two-level nonlinear integer programming problem in order to test the proposed algorithm.

$$
\begin{aligned}
&\underset{\text{DM1}}{\text{maximize}} && f_1(\boldsymbol{x}) = \prod_{j=1}^{n} \left[1-(1-r_j)^{x_j}\right] \\
&\underset{\text{DM2}}{\text{minimize}} && f_2(\boldsymbol{x}) = \sum_{j=1}^{n} c_j \left[x_j + \exp\left(\frac{x_j}{4}\right)\right] \\
&\underset{\text{DM3}}{\text{minimize}} && f_3(\boldsymbol{x}) = \sum_{j=1}^{n} q_j x_j \, \exp\left(\frac{x_j}{4}\right) \\
&\text{subject to} && g_1(\boldsymbol{x}) = \sum_{j=1}^{n} p_j x_j^2 - P \leq 0 \\
& && g_2(\boldsymbol{x}) = \sum_{j=1}^{n} w_j x_j \, \exp\left(\frac{x_j}{4}\right) - W \leq 0 \\
& && x_j \in \{1,2,\ldots,10\},\ j=1,\ldots,n
\end{aligned}
\tag{16}
$$

The numerical experiments were performed on a personal computer (processor: Intel 1 GHz, memory: 512 MB, OS: Windows 2000) using Visual C/C++ compiler (version 6.0). The parameter values used in GADSCRRSU for solving (16) were set as follows: $N = 100$, $I_{\min} = 100$, $I_{\max} = 1000$, $\epsilon = 0.02$, $R = 0.9$, $\theta = 5.0$, $\eta = 0.1$, $\sigma = 2.0$, $\tau = 3.0$, $\rho = 0.005$, $\nu = 10$, $c_{\text{mult}} = 1.8$, $p_c = 0.9$, $G = 0.9$, $p_m = 0.05$, $p_i = 0.03$, and $P = 200$. Several problems with different numbers of variables were considered to test the proposed algorithm for solving the problem (16). The data were generated randomly. In the following section, the result has been discussed briefly when $n = 15$ and the data are shown in Table 1.

Table 1: Data for problem (16) with 15 variables

Co-efficients	Values														
$\boldsymbol{r}$	0.737	0.882	0.844	0.930	0.613	0.630	0.777	0.631	0.986	0.778	0.892	0.978	0.990	0.718	0.751
$\boldsymbol{c}$	0.030	0.700	0.900	0.200	0.600	1.100	0.020	0.130	0.200	1.000	0.030	0.500	10.000	0.090	0.060
$\boldsymbol{q}$	2.310	12.590	4.600	4.750	1.450	1.060	4.060	2.460	3.060	2.000	3.850	4.290	3.460	0.950	3.910
$\boldsymbol{p}$	3.000	3.000	4.000	1.000	3.000	4.000	3.000	5.000	4.000	5.000	4.000	4.000	2.000	3.000	2.000
$\boldsymbol{w}$	7.000	0.800	0.300	0.200	1.000	5.000	0.900	1.000	8.000	0.800	0.300	1.000	0.500	9.000	2.000
P	905.000														
W	293.195														

5.1 Result and Discussion First, in step 1, the individual minimum and maximum of each objective function of all the DMs are calculated by using Zimmermann method and are shown in Table 2. After calculating individual minimum and maximum of each objective function of all the DMs, the corresponding membership functions are specified subjectively by the DMs.

Since DM1's objective function is a maximization type, the linear membership function

Table 2: Calculated individual minimum and maximum

Objective	$f_l^{\min}$	$f_l^{\max}$	Time (Sec.)
$f_1(\boldsymbol{x})$	0.0345	0.9195	29.87
$f_2(\boldsymbol{x})$	35.4394	63.1294	
$f_3(\boldsymbol{x})$	70.3646	546.8730	

in equation (17) in Figure 3 is used to specify the fuzzy goal of the DM1.

$$\mu_1(f_1(\boldsymbol{x})) = \begin{cases} 0 & f_1(\boldsymbol{x}) < f_1^0 \\ \dfrac{f_1(\boldsymbol{x}) - f_1^0}{f_1^1 - f_1^0} & f_1^0 \leq f_1(\boldsymbol{x}) < f_1^1 \\ 1 & f_1(\boldsymbol{x}) \geq f_1^1 \end{cases} \tag{17}$$

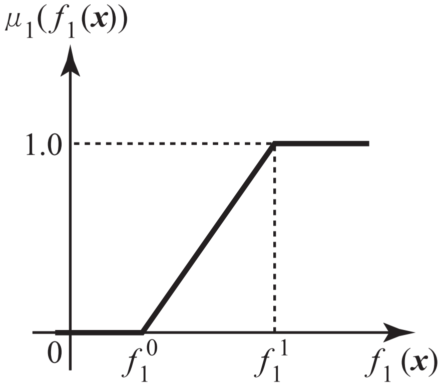


Figure 3: Linear Membership Function for DM1

On the other hand, since DM2's and DM3's objective function are minimization type, the linear membership function in equation (4) in Figure 1 is used to specify the fuzzy goal of the DM2 and DM3. The corresponding linear membership functions of all the DMs are shown in Table 3.

Table 3: DMs' linear membership functions

Decision Maker	f_l^0	f_l^1
DM1	0.25	0.80
DM2	80.00	40.00
DM3	600.00	100.00

After specifying the linear membership functions, in order to obtain a satisfactory solution to the multi-level nonlinear integer programming problem (16), we apply the proposed interactive algorithm to it. The process of interaction is summarized in Table 4. In the numerical experiment, at the third interaction, termination conditions were satisfied for (16) with 15 variables.

Table 4: Interactive results of problem (16) with 15 variables

Int.	$\hat{\delta}_1$	$\hat{\delta}_2$	$\mu_1(f_1)$	$\mu_2(f_2)$	$\mu_3(f_3)$	$f_1(\boldsymbol{x})$	$f_2(\boldsymbol{x})$	$f_3(\boldsymbol{x})$	Δ_1	Δ_2	Time (S.)
1st	...	...	0.7706	0.7687	0.7637	0.6738	49.2528	218.1386	0.9976	0.9936	10.90
2nd	...	0.78	0.7589	0.7804	0.7545	0.6674	48.7853	222.7474	1.0282	0.9669	21.57
3rd	0.80	0.78	0.8072	0.7850	0.7299	0.6939	48.5992	235.0590	0.9726	0.9298	15.96

Suppose that DM1 and DM2 subjectively specify the lower and upper bounds of Δ_1 and Δ_2 as [0.6, 1.0]. In step 2, the problem (8) for the numerical example (16) can be fomulated as

$$\underset{\boldsymbol{x}\in X}{\text{maximize}} \quad \min\left\{\frac{f_1(\boldsymbol{x})-0.25}{0.80-0.25}, \frac{f_2(\boldsymbol{x})-80.00}{40.00-80.00}, \frac{f_3(\boldsymbol{x})-600.00}{100.00-600.00}\right\}, \tag{18}$$

where X denotes the feasible area of problem (16) and (18) is solved through GADSCRRSU. Result is shown in the second row of Table 4. At this interaction, the termination conditions of the interactive process are not satisfied. In step 3 and 4, it needs to update the minimal satisfactory levels of DM1 and DM2 one after another.

Suppose that DM2 specifies the minimal satisfactory level $\hat{\delta}_2 = 0.78$. Then a problem corresponding to (12) is formulated as

$$\begin{array}{ll} \text{maximize} & \min\left\{\dfrac{f_1(\boldsymbol{x})-0.25}{0.80-0.25}, \dfrac{f_3(\boldsymbol{x})-600.00}{100.00-600.00}\right\} \\ \text{subject to} & \boldsymbol{x}\in X \\ & \dfrac{f_2(\boldsymbol{x})-80.00}{40.00-80.00} \geq 0.78. \end{array} \tag{19}$$

In step 5, (19) is solved through GADSCRRSU. Result is shown in the third row of Table 4. At this interaction, the termination conditions of the interactive process for DM2 are satisfied but are not satisfied for DM1. Again in step 3 and 4, it needs to update the minimal satisfactory level of DM1.

Suppose that DM1 specifies the minimal satisfactory level $\hat{\delta}_1 = 0.80$. Then a problem corresponding to (12) is formulated as

$$\begin{array}{ll} \text{maximize} & \min\left\{\dfrac{f_3(\boldsymbol{x})-600.00}{100.00-600.00}\right\} \\ \text{subject to} & \boldsymbol{x}\in X \\ & \dfrac{f_2(\boldsymbol{x})-80.00}{40.00-80.00} \geq 0.78 \\ & \dfrac{f_1(\boldsymbol{x})-0.25}{0.80-0.25} \geq 0.80. \end{array} \tag{20}$$

In step 5, (20) is solved through GADSCRRSU. Result is shown in the fourth row of Table 4. At this interaction, the termination conditions of the interactive process for both the DM1 and DM2 are satisfied because the degree of satisfaction $\mu_1^3 = 0.8072$ of DM1 is greater than his minimal satisfactory level $\hat{\delta}_1 = 0.80$ and $\Delta_1^3 = 0.9726 \in [0.60,\ 1.00]$. Consequently, the degree of satisfaction $\mu_2^3 = 0.7850$ of DM2 is greater than his minimal satisfactory level $\hat{\delta}_2 = 0.78$ and $\Delta_2^3 = 0.9298 \in [0.60,\ 1.00]$. Therefore, the solution at the third interaction satisfies the termination conditions of the interactive process and then becomes a satisfactory solution for all the DMs.

Through the application of the proposed method to this test problem (16), we could find one of satifactory solutions. Furthermore, we applied the proposed method to (16) with different numbers of variables. In all applications, we could also find one of satifactory solutions. Figure 4 shows the relation between the number of variables and the total computational time. From the figure, it seems that the computational time increases polynomially as the number of variables increases. These results indicate the feasibility and efficiency of the proposed interactive fuzzy programming for MLNLIPPs.

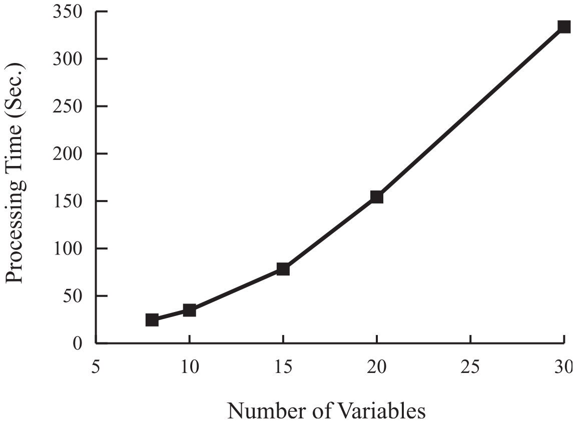


Figure 4: Total Computational Time

6 Conclusion In this paper, focusing on multi-level nonlinear integer programming problems, an interactive fuzzy programming procedure for them through genetic algorithms with double strings using continuous relaxation based on reference solution updating (GADSCRRSU) is presented. Furthermore, the feasibility of the proposed method is shown by applying it to illustrative numerical examples and checking the total computational time for various size problems.

References

[1] M. A. K. Azad, M. Sakawa, K. Kato and H. Katagiri, Interactive fuzzy programming for two-level nonlinear integer programming problems through genetic algorithms, Asia Pacific Management Review: An International Journal, vol. 10, no. 1, pp. 70-77, 2005 (in press).

[2] J. F. Bard, An investigation of the linear three level programming problem, IEEE transaction on Systems, Man, and Cybernetics, vol. SMC-14, pp. 711-717, 1983.

[3] R. E. Bellman, L. A. Zadeh, Decision making in a fuzzy environment, Management Science, vol. 17, pp. 141-164, 1970.

[4] W. F. Bialas and M. H. Karwan, Two-Level Linear Programming, Management Science, vol. 30, no. 8, pp. 1004-1020, 1984.

[5] GENOCOP V, ftp://ftp.uncc.edu/coe/evol/zbyszek/genocop5.0.tar.Z

[6] R. Jan and M. Chern, Nonlinear Integer Bilevel Programming, European Journal of Operational Research, vol. 72, pp. 574-587, 1994.

[7] Y. J. Lai, Hierarchical optimization: a satisfactory solution, Fuzzy Sets and Systems, vol. 77, pp. 321-335, 1996.

[8] I. Nishizaki, M. Sakawa and K. Kato, Computational methods through Genetic Algorithms for obtaining Stackelberg solutions to Two-level Zero-One Programming Problems, Proceedings of 2000 IEEE International Conference on Industrial Electronics, Control and Instrumentation, pp. 2750-2755, 2000.

[9] I. Nishizaki and M. Sakawa, Computational Methods through Genetic Algorithms for obtaining Stackelberg solutions to Two-Level Mixed Zero-One Programming Problems, Cybernetics and Systems: An International Journal, Vol. 31, pp. 203-221, 2000.

[10] M. Sakawa, Genetic Algorithms and Fuzzy Multiobjective Optimization, Kluwer Academic Publishers, Boston, USA, 2002.

[11] M. Sakawa and K. Kato, Genetic algorithms with double strings for 0-1 programming problems, European Journal of Operational Research, vol. 144, pp. 581-597, 2003.

[12] M. Sakawa, I. Nishizaki and Y. Uemura, Interactive fuzzy programming for multi-level linear programming problems, Computers and Mathematics with Applications, vol. 36, pp. 71-86, 1998.

[13] M. Sakawa, I. Nishizaki and Y. Uemura, Interactive fuzzy programming for multi-level linear programming problems with fuzzy parameters, Fuzzy Sets and Systems, vol. 109, pp. 3-19, 2000.

[14] M. Sakawa, I. Nishizaki and M. Hitaka, Interactive fuzzy programming for multi-level 0-1 programming problems through genetic algorithms, European Journal of Operational Research, vol. 114, pp. 580-588, 1999.

[15] H. S. Shih, Y. J. Lai and E. S. Lee, Fuzzy approach for multi-level programming problems, Computers and Operations Research, vol. 23, pp. 73-91, 1996.

[16] K. Shimizu, Y. Ishizuka and J.F. Bard, Nondifferentiable and Two-Level Mathematical Programming, Kluwer Academic Publishers, Boston, USA, 1997.

[17] U. -P. Wen and W. F. Bialas, The hybrid algorithm for solving the three-level linear programming problem, Conputers and Operations Research, vol. 13, pp. 367-377, 1986.

[18] H. -J. Zimmermann, Fuzzy programming and linear programming with several objective functions, Fuzzy Sets and Systems, vol. 1, pp. 45-55, 1978.

Corresponding Author:
Md. Abul Kalam Azad
Department of Artificial Complex Systems Engineering
Graduate School of Engineering, Hiroshima University
1-4-1 Kagamiyama, Higashi Hiroshima 739-8527, JAPAN
Tel.+Fax: +81-(0)82-424-7694
E-mail: azad@msl.sys.hiroshima-u.ac.jp