Proceedings of the 2007 IEEE Symposium on Computational
Intelligence in Multicriteria Decision Making (MCDM 2007)

Interactive fuzzy programming based on a probability maximization model using genetic algorithms for two-level integer programming problems involving random variable coefficients

Kosuke Kato
Graduate School of Engineering,
Hiroshima University
1-4-1, Kagamiyama, Higashi-hiroshima,
739-8527 Japan
Email: kato@msl.sys.hiroshima-u.ac.jp

Masatoshi Sakawa
Graduate School of Engineering,
Hiroshima University
1-4-1, Kagamiyama, Higashi-hiroshima,
739-8527 Japan
Email: sakawa@msl.sys.hiroshima-u.ac.jp

***Abstract*— In this paper, we focus on two-level integer programming problems with random variable coefficients in objective functions and/or constraints. Using chance constrained programming approaches in stochastic programming, the stochastic two-level integer programming problems are transformed into deterministic two-level integer programming problems. After introducing fuzzy goals for objective functions, we consider the application of the interactive fuzzy programming technique to derive a satisfactory solution for decision makers. Since several integer programming problems have to be solved in the interactive fuzzy programming technique, we incorporate a genetic algorithm designed for integer programming problems into it. An illustrative numerical example is provided to demonstrate the feasibility of the proposed method.**

I. Introduction

In this paper, we consider decision making situations in hierarchical systems, which are formulated as two-level integer programming problems. In these problems, there exist a decision maker with integer decision variables at the upper level and another decision maker with integer decision variables at the lower level.

For two-level programming problems, a number of approaches are proposed according to the relationship between these decision makers. Under the assumption that they do not have motivation to cooperate mutually, the Stackelberg solution [18] is adopted as a reasonable solution for the situation. On the other hand, in the case of a project selection problem in an administrative office and an autonomous divisions of a company, it seems natural that the decision makers cooperate with each other. For such cooperative situations, solutions that both decision makers can be satisfied with seems reasonable. In order to obtain such the satisfactory solution for the decision makers, Sakawa et al. have proposed interactive fuzzy programming techniques for two-level or multi-level linear programming problems [15], [16].

In actual decision making situations, we must often make a decision on the basis of vague information or uncertain data. For such decision making problems involving uncertainty, there exist two typical approaches: probability-theoretic approach and fuzzy-theoretic one. Stochastic programming, as an optimization method based on probability theory, have been developed in various ways [7], including two stage problem considered by G.B. Dantzig [3], chance constrained programming proposed by A. Charnes and W.W. Cooper [2]. For multiobjective stochastic linear programming problems, I.M. Stancu-Minasian [7] discussed the minimum risk approach, while J.P. Leclercq [5] and J. Teghem Jr. et al. [20] proposed interactive decision making methods.

On the other hand, fuzzy mathematical programming representing the vagueness in decision making situations by fuzzy concepts have been studied by many researchers [8], [9]. Fuzzy multiobjective linear programming, first proposed by H.-J. Zimmermann [21], have been developed rapidly developed by numerous researchers, and an increasing number of successful applications has been appearing [17], [6], [19].

As a hybrid of the stochastic approach and the fuzzy one, Sakawa et al. [11], [12] presented an interactive fuzzy satisficing method to derive a satisficing solution for the decision maker after reformulating a multiobjective stochastic linear programming problem using several models for chance constrained programming. Furthermore, they also proposed decision making methods based on interactive fuzzy programming for two-level stochastic linear programming problems. However, there has never been reported the application of these methods to two-level stochastic programming problems with discrete decision variables.

Under these circumstances, in this paper, we deal with two-level integer programming problems with random variable coefficients in objective functions and/or constraints. First, the two-level stochastic integer programming problem is transformed into deterministic ones based on a probability maximization model. Then, we attempt to obtain a satisfactory solution for decision makers through interactive fuzzy programming [15], [16]. In this derivation of a satisfactory solution through interactive fuzzy programming, several deterministic linear and nonlinear integer programming problems have to be solved. Since it is generally difficult to strictly solve linear and nonlinear integer programming problems with considerably many decision variables and/or constraints, we use a genetic algorithm designed for integer programming problems.

II. Two-level stochastic integer programming problems

Two-level stochastic integer programming problems involving random variable coefficients are formualted as follows.

$$\left.\begin{array}{ll}\underset{\text{DM1 (upper level)}}{\text{minimize}} & z_1(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{11}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{12}(\omega)\boldsymbol{x}_2\\ \underset{\text{DM2 (lower level)}}{\text{minimize}} & z_2(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{21}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{22}(\omega)\boldsymbol{x}_2\\ \text{subject to} & A_1\boldsymbol{x}_1+A_2\boldsymbol{x}_2\le\boldsymbol{b}(\omega)\\ & x_{1j_1}\in\{0,1,\dots,\nu_{1j_1}\},\ j_1=1,\dots,n_1\\ & x_{2j_2}\in\{0,1,\dots,\nu_{2j_2}\},\ j_2=1,\dots,n_2\end{array}\right\}\tag{1}$$

where $\boldsymbol{x}_1$ is the n_1 dimensional integer decision variable column vector for the upper level decision maker DM1, $\boldsymbol{x}_2$ is the n_2 dimensional integer decision variable column vector for the lower level decision maker DM2, A_1 is the $m\times n_1$ coefficient matrix, A_2 is the $m\times n_2$ coefficient matrix. It should be noted that $\boldsymbol{c}_{lj}(\omega)$, $l=1,2$, $j=1,2$ are Gaussian random variable row vectors with finite mean $\bar{\boldsymbol{c}}_{lj}$ and finite covariance matrix $V_{lpq}=\left(v^l_{h_ph_q}\right)=\left(\mathrm{Cov}[c_{lh_p}(\omega),c_{lh_q}(\omega)]\right)$, $p=1,2$, $q=1,2$, $h_p=1,\dots,n_p$, $h_q=1,\dots,n_q$, and $b_i(\omega)$, $i=1,\dots,m$ are random variables with finite mean $\bar{b}_i$ which are independent of each other, and the distribution function of each of them is also assumed to be continuous and increasing. Furthermore, ν_{lj_l}, $l=1,2$, $j_l=1,\dots,n_l$ are positive integer values.

Since (1) contains random variable coefficients, definitions and solution methods for ordinary deteministic mathematical programming problems cannot be directly applied. Consequently, we deal with the constraints in (1) as chance constrained conditions [2] which mean that the constraints need to be satisfied not always but with a certain probability (satisficing level) and over. Namely, replacing the constraints in (1) by chance constrained conditions with satisficing levels β_i, $i=1,\dots,m$, (1) can be converted as:

$$\left.\begin{array}{ll}\underset{\text{DM1 (upper level)}}{\text{minimize}} & z_1(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{11}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{12}(\omega)\boldsymbol{x}_2\\ \underset{\text{DM2 (lower level)}}{\text{minimize}} & z_2(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{21}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{22}(\omega)\boldsymbol{x}_2\\ \text{subject to} & \Pr\{\boldsymbol{a}_{i1}\boldsymbol{x}_1+\boldsymbol{a}_{i2}\boldsymbol{x}_2\le b_i(\omega)\}\ge\beta_i\\ & \qquad\qquad i=1,\dots,m\\ & x_{1j_1}\in\{0,1,\dots,\nu_{1j_1}\},\ j_1=1,\dots,n_1\\ & x_{2j_2}\in\{0,1,\dots,\nu_{2j_2}\},\ j_2=1,\dots,n_2\end{array}\right\}\tag{2}$$

where $\boldsymbol{a}_i$ is the ith row vector of A and $b_i(\omega)$ is the ith element of $\boldsymbol{b}(\omega)$.

Because distribution functions $F_i(r)=\Pr\{b_i(\omega)\le r\}$ of random variables $b_i(\omega)$, $i=1,\dots,m$ are continuous and increasing, the i th constraint in (2) can be rewritten as:

$$\Pr\{\boldsymbol{a}_{i1}\boldsymbol{x}_1+\boldsymbol{a}_{i2}\boldsymbol{x}_2\le b_i(\omega)\}\ge\beta_i\Leftrightarrow\boldsymbol{a}_{i1}\boldsymbol{x}_1+\boldsymbol{a}_{i2}\boldsymbol{x}_2\le\hat{b}_i$$

where $\hat{b}_i=F_i^{-1}(1-\beta_i)$.

Thereby, (2) can be transformed into the following equivalent problem:

$$\left.\begin{array}{ll}\underset{\text{DM1 (upper level)}}{\text{minimize}} & z_1(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{11}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{12}(\omega)\boldsymbol{x}_2\\ \underset{\text{DM2 (lower level)}}{\text{minimize}} & z_2(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{21}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{22}(\omega)\boldsymbol{x}_2\\ \text{subject to} & A_1\boldsymbol{x}_1+A_2\boldsymbol{x}_2\le\hat{\boldsymbol{b}}\\ & x_{1j_1}\in\{0,1,\dots,\nu_{1j_1}\},\ j_1=1,\dots,n_1\\ & x_{2j_2}\in\{0,1,\dots,\nu_{2j_2}\},\ j_2=1,\dots,n_2\end{array}\right\}\tag{3}$$

where $\hat{\boldsymbol{b}}=(\hat{b}_1,\dots,\hat{b}_m)^T$. In the following, for notational convenience, the feasible region of (3) is denoted by X.

For the two-level chance constrained programming problem (3), several models such as an expectation optimization model, a variance minimization model, a probability maximization model, a fractile criterion optimization model, have been proposed depending on the concern of the decision maker.

In this paper, we study the probability maximization model, which aims to maximize the probability that each of objective functions is less than or equal to a certain permissible level.

III. Probability maximization model

In (3), substituting the maximization of the probability that each of the objective functions $z_l(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)$ is less than or equal to a certain permissible level f_l for the minimization of the objective functions $z_l(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{l1}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{l2}(\omega)\boldsymbol{x}_2$, $l=1,2$, the problem can be converted as follows.

$$\left.\begin{array}{ll}\underset{\text{DM1 (upper level)}}{\text{maximize}} & p_1(\boldsymbol{x}_1,\boldsymbol{x}_2)=\Pr\{z_1(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)\le f_1\}\\ \underset{\text{DM2 (lower level)}}{\text{maximize}} & p_2(\boldsymbol{x}_1,\boldsymbol{x}_2)=\Pr\{z_2(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)\le f_2\}\\ \text{subject to} & A_1\boldsymbol{x}_1+A_2\boldsymbol{x}_2\le\hat{\boldsymbol{b}}\\ & x_{1j_1}\in\{0,1,\dots,\nu_{1j_1}\},\ j_1=1,\dots,n_1\\ & x_{2j_2}\in\{0,1,\dots,\nu_{2j_2}\},\ j_2=1,\dots,n_2\end{array}\right\}\tag{4}$$

The objective functions $\Pr\{z_l(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)\le f_l\}$, $l=1,2$ are rewritten as:

$$\begin{aligned}&\Pr\{z_l(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)\le f_l\}\\ &\quad=\ \Pr\{\boldsymbol{c}_{l1}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{l2}(\omega)\boldsymbol{x}_2\le f_l\}\\ &\quad=\ \Pr\Bigg\{\frac{\boldsymbol{c}_{l1}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{l2}(\omega)\boldsymbol{x}_2-(\bar{c}_{l1}\boldsymbol{x}_1+\bar{c}_{l2}\boldsymbol{x}_2)}{\sqrt{(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)V_l(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)^T}}\\ &\qquad\qquad\le\frac{f_l-(\bar{c}_{l1}\boldsymbol{x}_1+\bar{c}_{l2}\boldsymbol{x}_2)}{\sqrt{(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)V_l(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)^T}}\Bigg\}\\ &\quad=\ \Phi_l\left(\frac{f_l-(\bar{c}_{l1}\boldsymbol{x}_1+\bar{c}_{l2}\boldsymbol{x}_2)}{\sqrt{(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)V_l(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)^T}}\right)\end{aligned}$$

where $\Phi_l(\cdot)$ is the distribution function of a standard Gaussian random variable.

Then, (4) can be transformed into the following equivalent problem:

$$\left.\begin{array}{rl}\underset{\text{DM1 (upper level)}}{\text{maximize}} & \Phi_1\left(\dfrac{f_1-(\bar{c}_{11}\boldsymbol{x}_1+\bar{c}_{12}\boldsymbol{x}_2)}{\sqrt{(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)V_1(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)^T}}\right)\\ \underset{\text{DM2 (lower level)}}{\text{maximize}} & \Phi_2\left(\dfrac{f_2-(\bar{c}_{21}\boldsymbol{x}_1+\bar{c}_{22}\boldsymbol{x}_2)}{\sqrt{(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)V_2(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)^T}}\right)\\ \text{subject to} & A_1\boldsymbol{x}_1+A_2\boldsymbol{x}_2\le\hat{\boldsymbol{b}}\\ & x_{1j_1}\in\{0,1,\ldots,\nu_{1j_1}\}, j_1=1,\ldots,n_1\\ & x_{2j_2}\in\{0,1,\ldots,\nu_{2j_2}\}, j_2=1,\ldots,n_2\end{array}\right\}\tag{5}$$

IV. Interactive fuzzy programming

In general, it seems natural that there exists the ambiguity or fuzziness in the evaluation of each objective function by the decision maker. In order to consider the imprecise nature of the decision maker's judgement for each objective function in (5), we introduce the fuzzy goals such as "$p_l(\boldsymbol{x}_1,\boldsymbol{x}_2)$ should be substantially greater than or equal to a certain value". Then, (5) can be rewritten as:

$$\left.\begin{array}{rl}\underset{\text{DM1 (upper level)}}{\text{maximize}} & \mu_1(p_1(\boldsymbol{x}_1,\boldsymbol{x}_2))\\ \underset{\text{DM2 (lower level)}}{\text{maximize}} & \mu_2(p_2(\boldsymbol{x}_1,\boldsymbol{x}_2))\\ \text{subject to} & A_1\boldsymbol{x}_1+A_2\boldsymbol{x}_2\le\hat{\boldsymbol{b}}\\ & x_{1j_1}\in\{0,1,\ldots,\nu_{1j_1}\}, j_1=1,\ldots,n_1\\ & x_{2j_2}\in\{0,1,\ldots,\nu_{2j_2}\}, j_2=1,\ldots,n_2\end{array}\right\}\tag{6}$$

where $\mu_l(\cdot)$ is a membership function to quantify a fuzzy goal for the lth objective function in (5), as shown in Fig. 1.

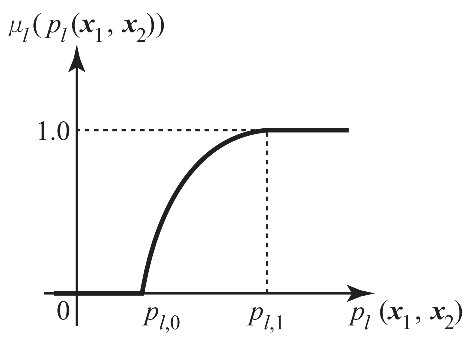

Fig. 1. Membership function

We attempt to derive a satisfactory solution by the interactive fuzzy programming technique using fuzzy goals to consider the ambiguity of the decision makers' judgement and a ratio of the satisfactory degree for DM2 (μ_2) to that for DM1 (μ_1).

Computational procedure of interactive fuzzy programming

Step 1: Determine the satisficing levels $\beta_i,\ i=1,\ldots,m$ for constraints in (2).

Step 2: Solve problems (7), (8) to obtain the individual minimum $\bar{z}_{l,\min}$ and maximum $\bar{z}_{l,\max}$ of $\mathrm{E}\{z_l(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)\}=\bar{\boldsymbol{c}}_{l1}\boldsymbol{x}_1+\bar{\boldsymbol{c}}_{l2}\boldsymbol{x}_2$, $l=1,2$ under the chance constrained conditions with satisficing levels $\beta_i,\ i=1,\ldots,m$.

$$\underset{\boldsymbol{x}\in X}{\text{minimize}}\quad \bar{z}_l(\boldsymbol{x}_1,\boldsymbol{x}_2),\ l=1,2\tag{7}$$

$$\underset{\boldsymbol{x}\in X}{\text{maximize}}\quad \bar{z}_l(\boldsymbol{x}_1,\boldsymbol{x}_2),\ l=1,2\tag{8}$$

Since these problems are deterministic integer programming problems, in order to find (approximate) optimal solutions to them, we apply Genetic Algorithm with Double Strings using Continuous Relaxation based on Reference Solution Updating (GADSCRRSU) designed for integer programming problems [10]. Then, specify permissible levels $f_l,\ l=1,2$ for the objective functions in consideration of $\bar{z}_{l,\min}$ and $\bar{z}_{l,\max}$.

Step 3: For the purpose of obtaining the individual minimum $p_{l,\min}$ and maximum $p_{l,\max}$ of $p_l(\boldsymbol{x}_1,\boldsymbol{x}_2)$, $l=1,2$ in (5), solve the following problems.

$$\underset{\boldsymbol{x}\in X}{\text{minimize}}\quad \frac{f_l-(\bar{c}_{l1}\boldsymbol{x}_1+\bar{c}_{l2}\boldsymbol{x}_2)}{\sqrt{(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)V_l(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)^T}}\tag{9}$$

$$\underset{\boldsymbol{x}\in X}{\text{maximize}}\quad \frac{f_l-(\bar{c}_{l1}\boldsymbol{x}_1+\bar{c}_{l2}\boldsymbol{x}_2)}{\sqrt{(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)V_l(\boldsymbol{x}_1^T,\boldsymbol{x}_2^T)^T}}\tag{10}$$

We also apply GADSCRRSU [10] to solve these problems. Then, specify membership functions for objective functions in (5), $\mu_l(p_l(\boldsymbol{x}_1,\boldsymbol{x}_2))$, $l=1,2$, and set the upper bound $\Delta_{\max}$ and the lower bound $\Delta_{\min}$ of a ratio of the satisfactory degree for DM2 to that for DM1, $\Delta=\mu_2(p_2(\boldsymbol{x}_1,\boldsymbol{x}_2))/\mu_1(p_1(\boldsymbol{x}_1,\boldsymbol{x}_2))$.

Step 4: Based on the maximizing decision of Bellman and Zadeh [1], solve the following maximin problem through GADSCRRSU [10].

$$\left.\begin{array}{rl}\text{maximize} & \min\limits_{l=1,2}\{\mu_l(p_l(\boldsymbol{x}_1,\boldsymbol{x}_2))\}\\ \text{subject to} & \boldsymbol{x}\in X\end{array}\right\}\tag{11}$$

If DM1 is satisfied with the optimal solution to (11), terminate the interaction procedure. Otherwise, taking into account a ratio of satisfactory degrees Δ, DM1 subjectively specifies the minimal satisfactory level $\hat{\delta}$ for $\mu_1(p_1(\boldsymbol{x}_1,\boldsymbol{x}_2))$.

Step 5: Solve the following problem for $\hat{\delta}$ using GADSCRRSU [10].

$$\left.\begin{array}{rl}\text{maximize} & \mu_2(p_2(\boldsymbol{x}_1,\boldsymbol{x}_2))\\ \text{subject to} & \mu_1(p_1(\boldsymbol{x}_1,\boldsymbol{x}_2))\ge\hat{\delta}\\ & \boldsymbol{x}\in X\end{array}\right\}\tag{12}$$

Then, calculate the value of Δ corresponding to the optimal solution $(\boldsymbol{x}_1^*,\boldsymbol{x}_2^*)$ to (12).

Step 6: If DM1 is satisfied with $\mu_l(p_l(\boldsymbol{x}_1^*,\boldsymbol{x}_2^*))$, $l=1,2$ and $\Delta\in[\Delta_{\min},\Delta_{\max}]$, stop. Otherwise, ask DM1 to update the minimal satisfactory level $\hat{\delta}$. To be more specific, if $\Delta\le\Delta_{\min}$, i.e., $\mu_1(p_1(\boldsymbol{x}_1^*,\boldsymbol{x}_2^*))$ is much greater than $\mu_2(p_2(\boldsymbol{x}_1^*,\boldsymbol{x}_2^*))$, DM1 should decrease the value of $\hat{\delta}$. If $\Delta\ge\Delta_{\max}$, i.e., $\mu_1(p_1(\boldsymbol{x}_1^*,\boldsymbol{x}_2^*))$ is much less than $\mu_2(p_2(\boldsymbol{x}_1^*,\boldsymbol{x}_2^*))$, DM1 should increase the value of $\hat{\delta}$. Go to step 5.

V. Genetic algorithm with double strings using continuous relaxation based on reference solution updating (GADSCRRSU)

In this section, we explain GADSCRRSU [10] proposed as a general solution method for linear integer programming problems defined as (13).

$$\left.\begin{array}{rl} \text{minimize} & \boldsymbol{c}\boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} \le \boldsymbol{b} \\ & x_j \in \{0, 1, \ldots, \nu_j\},\ j = 1, \ldots, n \end{array}\right\} \quad (13)$$

where $A = [\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n]$ is an $m \times n$ coefficient matrix, $\boldsymbol{b} = (b_1, \ldots, b_m)^T$ is an m dimensional column vector and $\boldsymbol{c} = (c_1, \ldots, c_n)$ is an n dimensional row vector.

A. Individual Representation

In GADSCRRSU, double strings shown in Fig. 2 is used as the individual representation.

Individual $\mathbf{S}$:

$s(1)$	$s(2)$	$\cdots$	$s(n)$
$g_{s(1)}$	$g_{s(2)}$	$\cdots$	$g_{s(n)}$

Fig. 2. Double string

In the figure, each of $s(j)$, $j = 1, \ldots, n$ is the index of an element in a solution vector and each of $g_{s(j)} \in \{0, 1, \ldots, \nu_{s(j)}\}$, $j = 1, \ldots, n$ is the value of the element, respectively. For example, the first column of the double string in Fig. 2 means that the candidate of the value of $x_{s(1)}$ is $g_{s(1)}$.

B. Decoding Algorithm

Since a solution corresponding to a double string, i.e., $\boldsymbol{x} = (g_1, g_2, \ldots, g_n)^T$ does not always safisfy the constraint $A\boldsymbol{x} \le \boldsymbol{b}$ in (13), the decoding procedure is needed to repair infeasible solutions. In [10], a decoding algorithm of double strings for linear integer programming problems is constructed as follows. In the algorithm, a feasible solution $\boldsymbol{x}^*$, called a reference solution, is used as the origin of decoding.

Decoding algorithm using a reference solution

In this algorithm, it is assumed that a feasible solution $\boldsymbol{x}^*$ to (13) is obtained in advance. Let n and N be the number of variables and the number of individuals in the population, respectively. Also, $\boldsymbol{b}^+$ means a column vector of positive right-hand side constants, and the corresponding coefficient matrix is denoted by $A^+ = (\boldsymbol{p}_1^+, \ldots, \boldsymbol{p}_n^+)$.

Step 1: Let $j := 1$ and $\mathbf{psum} := \mathbf{0}$.
Step 2: If $g_{s(j)} = 0$, set $q_{s(j)} := 0$ and $j := j + 1$, and go to step 4. If $g_{s(j)} \ne 0$, go to step 3.
Step 3: If $\mathbf{psum} + \boldsymbol{p}^+_{s(j)} \cdot g_{s(j)} \le \boldsymbol{b}^+$, set $q_{s(j)} := g_{s(j)}$, $\mathbf{psum} := \mathbf{psum} + \boldsymbol{p}^+_{s(j)} \cdot g_{s(j)}$ and $j := j + 1$, and go to step 4. Otherwise, set $q_{s(j)} := 0$ and $j := j+1$, and go to step 4.
Step 4: If $j > n$, go to step 5. If $j \le n$, go to step 2.
Step 5: Let $j := 1$, $l := 0$ and $\mathbf{sum} := \mathbf{0}$.
Step 6: If $g_{s(j)} = 0$, set $j := j + 1$ and go to step 8. If $g_{s(j)} \ne 0$, set $\mathbf{sum} := \mathbf{sum} + \boldsymbol{p}_{s(j)} \cdot g_{s(j)}$ and go to step 7.
Step 7: If $\mathbf{sum} \le \boldsymbol{b}$, set $l := j$, $j := j + 1$, and go to step 8. Otherwise, set $j := j + 1$ and go to step 8.
Step 8: If $j > n$, go to step 9. If $j \le n$, go to step 6.
Step 9: If $l > 0$, go to step 10. If not, go to step 11.
Step 10: For $x_{s(j)}$ satisfying $1 \le j \le l$, let $x_{s(j)} := g_{s(j)}$. For $x_{s(j)}$ satisfying $l + 1 \le j \le n$, let $x_{s(j)} := 0$, and stop.
Step 11: Let $\mathbf{sum} := \sum_{k=1}^{n} \boldsymbol{p}_{s(k)} \cdot x^*_{s(k)}$ and $j := 1$.
Step 12: If $g_{s(j)} = x^*_{s(j)}$, let $x_{s(j)} := g_{s(j)}$ and $j := j + 1$, and go to step 16. Otherwise, go to step 13.
Step 13: If $\mathbf{sum} - \boldsymbol{p}_{s(j)} \cdot x^*_{s(j)} + \boldsymbol{p}_{s(j)} \cdot g_{s(j)} \le \boldsymbol{b}$, set $\mathbf{sum} := \mathbf{sum} - \boldsymbol{p}_{s(j)} \cdot x^*_{s(j)} + \boldsymbol{p}_{s(j)} \cdot g_{s(j)}$ and $x_{s(j)} := g_{s(j)}$, and go to step 16. Otherwise, go to step 14.
Step 14: Let $t_{s(j)} := \lfloor 0.5 \cdot (x^*_{s(j)} + g_{s(j)}) \rfloor$ and go to step 15.
Step 15: If $\mathbf{sum} - \boldsymbol{p}_{s(j)} \cdot x^*_{s(j)} + \boldsymbol{p}_{s(j)} \cdot t_{s(j)} \le \boldsymbol{b}$, set $\mathbf{sum} := \mathbf{sum} - \boldsymbol{p}_{s(j)} \cdot x^*_{s(j)} + \boldsymbol{p}_{s(j)} \cdot t_{s(j)}$, $g_{s(j)} := t_{s(j)}$ and $x_{s(j)} := t_{s(j)}$, and go to step 16. Otherwise, set $x_{s(j)} := x^*_{s(j)}$ and go to step 16.
Step 16: If $j > n$, stop. Otherwise, return to step 12.

Because solutions obtained the decoding algorithm using a reference solution tend to concentrate around the reference solution, the reference solution updating procedure is adopted.

C. Reference solution updating

The diversity of solutions $\boldsymbol{x}$ greatly depends on the reference solution used in the above decoding algorithm. In order to widen the search region, we propose the following reference solution updating procedure such that the current reference solution is updating by another feasible solution if the diversity of solutions seems to be lost. To do so, for every generation, check the dependence on the reference solution through the calculation of the mean of the Hamming distance between all solutions corresponding to individuals and the reference solution, and when the dependence on the reference solution is strong, replace the reference solution by the solution corresponding to an individual having maximum Hamming distance.

Let N, $\boldsymbol{x}^*$, η (< 1.0) and $\boldsymbol{x}^r$ respectively denote the number of individuals, the reference solution, a parameter for reference solution updating and a feasible solution decoded by the r th individual, then the reference solution updating procedure can be described as follows.

Reference solution updating procedure

Step 1: Set $r := 1$, $r_{\max} := 1$, $d_{\max} := 0$ and $d_{\text{sum}} := 0$.
Step 2: Calculate $d_r = \sum_{j=1}^{n} |x^r_{s(j)} - x^*_{s(j)}|$ and let $d_{\text{sum}} := d_{\text{sum}} + d_r$. If $d_r > d_{\max}$ and $\boldsymbol{c}\boldsymbol{x}^r < \boldsymbol{c}\boldsymbol{x}^0$, let $d_{\max} := d_r$, $r_{\max} := r$ and $r := r + 1$, and go to step 3. Otherwise, let $r := r + 1$ and go to step 3.
Step 3: If $r > n$, go to step 4. Otherwise, return to step 2.
Step 4: If $d_{\text{sum}} / (N \cdot \sum_{j=1}^{n} \nu_j) < \eta$, then update the reference solution as $\boldsymbol{x}^* := \boldsymbol{x}^{r_{\max}}$, and stop. Otherwise,

TABLE I

RELATIONS BETWEEN OPTIMAL SOLUTIONS TO INTEGER KNAPSACK PROBLEMS AND THOSE TO CONTINUOUS RELAXATION PROBLEMS.

	$x_j^* = \hat{x}_j$	$x_j^* \neq \hat{x}_j$
$x_j^* = 0$	606	4
$x_j^* \neq 0$	279	111

stop without updating the reference solution.

It should be observed here that when the constraints of the problem are strict, there exist a possibility that all of the individuals are decoded in the neighborhood of the reference solution. To avoid a such possibility, in addition to the reference solution updating procedure, after every P generations, the reference solution is replaced by another feasible solution.

D. Usage of Continuous Relaxation

It is expected that an optimal solution to the continous relaxation problem becomes a good approximate optimal solution of the original integer programming problem. With this observation in mind, after generating 20 single-objective integer programming problems involving 50 variables and 10 constraints at random, they and their linear programming relaxation problems are solved. To be more explicit, c_j, $j = 1, \ldots, n$ and a_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$ are determined by uniform integer random numbers in $[-999, 0]$ and $[0, 999]$, respectively, while b_i, $i = 1, \ldots, m$ are defined as

$$b_i = \gamma \cdot \sum_{j=1}^{n} a_{ij}, \;\; i = 1, \ldots, m \tag{14}$$

where a positive constant γ is a parameter to control the degree of strictness of the constraints, determined by a uniform real random number ranging from 5 to 10. In addition, upper bounds ν_j of x_j, $j = 1, \ldots, n$ are set at 20 for all j.

Table 1 shows relations between the optimal solution to integer knapsack problems x_j^* and that to corresponding continuous relaxation problems $\hat{x}_j$, while Fig. 3 shows the frequency distribution of differences between the values of an optimal solution x_i of integer programming problems and an optimal solution $\hat{x}_i$ of linear programming relaxation problems.

As a result, it is recognized that each variable x_i takes exactly or approximately the same value that $\hat{x}_i$ does, especially, such variables x_i as $\hat{x}_i = 0$ are very likely to be equal to 0.

Based on the fact, the information about the optimal solution to the continuous relaxation problem is used in the generation of the initial population and the mutation [10].

E. Reproduction

As a reproduction operator, elitist expected value selection, which is the combination of expected value selection and elitist preserving selection, is adopted. In [14], elitist expected value selection is defined as a combination of elitism and expected value selection as mentioned below.

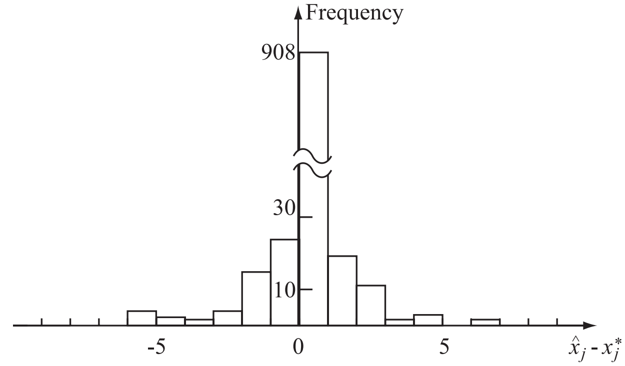

Fig. 3. Frequency distribution of $x_j - \hat{x}_j$.

Elitism: If the fitness of a string in the past populations is larger than that of every string in the current population, preserve this string into the current generation.

Expected value selection: For a population consisting of N strings, the expected value of the number of the i th string $\mathbf{s}_i$ in the next population

$$N_i = \frac{f(\mathbf{s}_i)}{\sum_{i=1}^{N} f(\mathbf{s}_i)} \times N$$

is calculated. Then, the integral part of N_i denotes the deterministic number of the string $\mathbf{s}_i$ preserved in the next population. While, the decimal part of N_i is regarded as probability for one of the string $\mathbf{s}_i$ to survive, i.e., $N - \sum_{i=1}^{N} N_i$ strings are determined on the basis of this probability.

F. Crossover

If a single-point crossover or multi-point crossover is directly applied to individuals of double string type, the kth element of an offspring may take the same number that the k'th element takes. Similar violation occurs in solving traveling salesman problems or scheduling problems through genetic algorithms as well. In order to avoid this violation, a crossover method called partially matched crossover (PMX) was proposed [4] and was modified so as to be suitable for double strings [14].

PMX for double string

Step 0: Select two individuals X, Y from the population as parent individuals and prepare copies X' and Y' of X and Y, respectively.

Step 1: Choose two crossover points at random on these strings, say, h and k $(h < k)$.

Step 2: (a) Set $j = h$.
(b) Find j' such that $s_{X'}(j') = s_Y(j)$. Then, interchange $(s_{X'}(j),\ g_{s_{X'}(j)})^T$ with $(s_{X'}(j'),\ g_{s_{X'}(j')})^T$ and set $j = j + 1$.
(c) If $j > k$, stop. Otherwise, return to (b).

Step 3: Replace the part from h to k of X' with that of Y and let X' be the offspring of X.

An illustrative example of crossover is shown in Fig. 4.

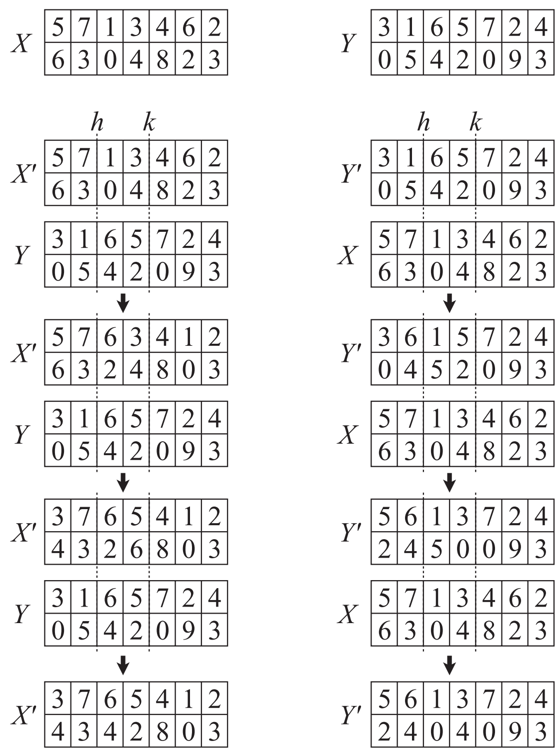

Fig. 4. Partially matched crossover (PMX) for double strings.

G. Mutation

It is considered that mutation plays the role of local random search in genetic algorithms. In this paper, two mutation operators (bit-reverse type and inversion) are used. A direct extension of mutation for 0-1 programming problems is to change the value of $g_{s(j)}$ at random in $[0, \nu_{s(j)}]$ uniformly, when mutation occurs at $g_{s(j)}$. The mutation operator is further refined by using the information about the solution of the linear programming relaxation problem $\hat{\boldsymbol{x}}$. To be more explicit, the following algorithm is carried out.

Mutation of bit reverse type for double strings

Step 0: Set $r := 1$.
Step 1: Set $j := 1$.
Step 2: Generate a random number R_τ according to the corresponding Gaussian distribution with mean $g_{s(j)}$ and variance τ^2. For a given mutation rate p_m, if $\texttt{rand}() \leq p_m$, then go to step 3. Otherwise, go to step 4.
Step 3: Let

$$g_{s(j)} := \begin{cases} 0 & , \quad R_\tau < 0.5 \\ \lfloor R_\tau + 0.5 \rfloor & , \quad 0.5 \leq R_\tau < \nu_j - 0.5 \\ \nu_j & , \quad R_\tau \geq \nu_j - 0.5 \end{cases}$$

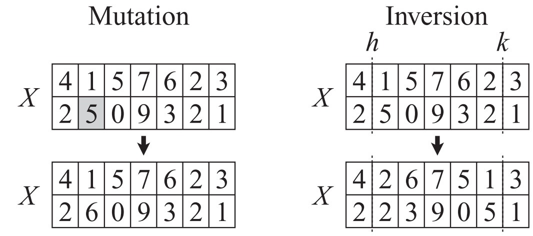

Fig. 5. Mutation and inversion for double strings.

and go to step 4.
Step 4: If $j < n$, set $j := j + 1$ and return to step 2. Otherwise, go to step 5.
Step 5: If $r < N$, set $r := r + 1$ and return to step 1. Otherwise, stop.

Inversion operator

Step 1: After determining the two inversion points h and k $(h < k)$, pick out a part of the upper row of a double string from h to k.
Step 2: Arrange the substring in the reverse order.
Step 3: Put the arranged substring back in the double string.

An illustrative example of mutation and inversion is shown in Fig. 5.

H. Computational procedures of GADSCRRSU

Step 0: Determine values of the parameters used in the genetic algorithm: the population size N, the generation gap G, the probability of crossover p_c, the probability of mutation p_m, the probability of inversion p_i, the minimal search generation $I_{\min}$, the maximal search generation $I_{\max}(> I_{\min})$, the scaling constant c_{mult}, the convergence criterion ε, the degree of use of information about solutions to linear programming relaxation problems R, and set the generation counter t at 0.
Step 1: Generate the initial population consisting of N individuals based on the information of the optimal solution to the continuous relaxation problem.
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 $t := t + 1$ and go to step 4.
Step 4: Apply reproduction operator using elitist expected value selection after linear scaling.
Step 5: Apply crossover operator, called PMX (Partially Matched Crossover) for double string.
Step 6: Apply mutation based on the information of a solution to the continuous relaxation problem.
Step 7: Apply inversion operator. Go to step 2.

VI. Numerical Example

To demonstrate the feasibility of the proposed method, consider the following two-level integer programming problem

TABLE II

MEANS OF COEFFICIENTS OF OBJECITVE FUNCTIONS.

$\bar{c}_1$	3	−8	3	−6	3	−6	2	−5	3	−3
	8	−2	6	−4	1	−4	2	−10	7	−5
	9	−7	1	−6	1	−2	2	−4	10	−7
$\bar{c}_2$	3	4	1	−6	10	10	−8	−4	6	3
	−9	2	9	1	−8	9	2	4	−1	3
	9	−7	2	10	7	10	5	7	−8	−5

involving random variable coefficients.

$$\left.\begin{array}{rl}\underset{\text{DM1 (upper level)}}{\text{minimize}} & z_1(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{11}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{12}(\omega)\boldsymbol{x}_2 \\ \underset{\text{DM2 (lower level)}}{\text{minimize}} & z_2(\boldsymbol{x}_1,\boldsymbol{x}_2,\omega)=\boldsymbol{c}_{21}(\omega)\boldsymbol{x}_1+\boldsymbol{c}_{22}(\omega)\boldsymbol{x}_2 \\ \text{subject to} & \boldsymbol{a}_{11}\boldsymbol{x}_1+\boldsymbol{a}_{12}\boldsymbol{x}_2 \le b_1(\omega) \\ & \boldsymbol{a}_{21}\boldsymbol{x}_1+\boldsymbol{a}_{22}\boldsymbol{x}_2 \le b_2(\omega) \\ & \vdots \\ & \boldsymbol{a}_{101}\boldsymbol{x}_1+\boldsymbol{a}_{102}\boldsymbol{x}_2 \le b_{10}(\omega) \\ & x_{1j_1}\in\{0,1,\dots,30\},\ j_1=1,\dots,15 \\ & x_{2j_2}\in\{0,1,\dots,30\},\ j_2=1,\dots,15 \end{array}\right\} \tag{15}$$

where $\boldsymbol{x}_1 = (x_1,\dots,x_{15})^T$, $\boldsymbol{x}_2 = (x_{16},\dots,x_{30})^T$, each element of random variable vectors $\boldsymbol{c}_l(\omega) = (\boldsymbol{c}_{l1}(\omega), \boldsymbol{c}_{l2}(\omega))$, $l = 1, 2$ is a Gaussian random variable whose mean is given as Table II, and random variables $b_i(\omega)$, $i = 1,\dots,10$ are also Gaussian random variables, $N(6753, 20^2)$, $N(-3588, 50^2)$, $N(5534, 25^2)$, $N(10225, 80^2)$, $N(-2837, 15^2)$, $N(6122, 30^2)$, $N(3169, 70^2)$, $N(5811, 100^2)$, $N(7565, 120^2)$, $N(-1433, 30^2)$, respectively. All of ν_{lj_l}s are equal to 30.

Table III shows the result of the application of the proposed interactive fuzzy programming based on the probability maximization model.

Parameters of GADSCRRSU [10] are set as: population size $N = 100$, crossover rate $p_c = 0.9$, generation gap $G = 0.9$, mutation rate $p_m = 0.05$, inversion rate $p_i = 0.05$, parameter for reproduction $\lambda = 0.9$, minimal search generation number $I_{\min} = 500$, maximal search generation number $I_{\max} = 1000$, scaling constant $c_{mult} = 1.6$, parameter for reference solution updating $\eta = 0.2$, penalty constant $\theta = 5$.

First, following step 1 in the interactive fuzzy programming, DM1 specifies satisficing levels β_i, $i = 1,\dots,10$ as:

$$\begin{array}{rcl}(\beta_1,\dots,\beta_{10})^T & = & (0.95,\ 0.80,\ 0.85,\ 0.90,\ 0.90, \\ & & \quad 0.85,\ 0.85,\ 0.95,\ 0.80,\ 0.95).\end{array}$$

Then, the corresponding $\hat{\boldsymbol{b}}$ are calculated as:

$$\begin{array}{rl}\hat{\boldsymbol{b}} = & (6720.10,\ -3630.08,\ 5508.09,\ 10122.48,\ -2856.22, \\ & \quad 6090.91,\ 3096.45,\ 5646.51,\ 7464.01,\ -1482.35)^T.\end{array}$$

Next, following step 2, minimal values $\bar{z}_{l,\min}$ and maximal values $\bar{z}_{l,\max}$ of objective functions $E[\bar{z}_l(\boldsymbol{x}_1,\boldsymbol{x}_2)]$ under the chance constrained conditions corresponding to given satisficing levels are calculated using GADSCRRSU as

$$\begin{array}{rclcrcl}\bar{z}_{1,\min} & = & -2206 & , & \bar{z}_{2,\min} & = & -1278 \\ \bar{z}_{1,\max} & = & 1670 & , & \bar{z}_{2,\max} & = & 3323\end{array}$$

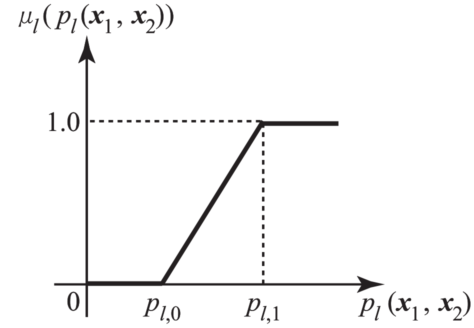

Fig. 6. Linear membership function

Based on these values, the permissible levels f_l, $l = 1, 2$ for objective functions are specified as $f_1 = -200$, $f_2 = 1000$.

Following step 3, after (9) and (10) are solved by GADSCRRSU, minimal values $p_{l,\min}$ and maximal values $p_{l,\max}$ are calculated as

$$\begin{array}{rclcrcl}p_{1,\min} & = & 0.003387 & , & p_{2,\min} & = & 0.183073 \\ p_{1,\max} & = & 0.908240 & , & p_{2,\max} & = & 0.999727\end{array}$$

In consideration of these values, the membership functions to quantify fuzzy goals for objective functions are subjectively determined. Here, the following linear membership function, as shown in Fig. 6, is adopted.

$$\begin{array}{l}\mu_l(p_l(\boldsymbol{x}_1,\boldsymbol{x}_2)) \\ \quad = \left\{\begin{array}{cl} 1 & ,\quad p_l(\boldsymbol{x}_1,\boldsymbol{x}_2) \ge p_{l,1} \\ \dfrac{p_l(\boldsymbol{x}_1,\boldsymbol{x}_2)-p_{l,0}}{p_{l,1}-p_{l,0}} & ,\ p_{l,0} < p_l(\boldsymbol{x}_1,\boldsymbol{x}_2) < p_{l,1} \\ 0 & ,\quad p_l(\boldsymbol{x}_1,\boldsymbol{x}_2) \le p_{l,0}\end{array}\right.\end{array}$$

In this paper, parameters $p_{l,1}$, $p_{l,0}$, $l = 1, 2$ which characterize membership functions $\mu_l(\cdot)$ are determined by Zimmermann's method [21].

$$\begin{array}{rcl}p_{1,1} = & p_1(\boldsymbol{x}_1^1,\boldsymbol{x}_2^1) & = & 0.908240 \\ p_{1,0} = & p_1(\boldsymbol{x}_1^2,\boldsymbol{x}_2^2) & = & 0.420420 \\ p_{2,1} = & p_2(\boldsymbol{x}_1^2,\boldsymbol{x}_2^2) & = & 0.999727 \\ p_{2,0} = & p_2(\boldsymbol{x}_1^1,\boldsymbol{x}_2^1) & = & 0.507786\end{array}$$

Furthermore, the upper bound and the lower bound of the ratio of satisfactory degrees Δ are set as $\Delta_{\max} = 0.85$ and $\Delta_{\min} = 0.75$.

Following step 4, the maxmin problem, (11), are solved by GADSCRRSU. For the obtained optimal solution $\boldsymbol{x}_1^*$, $\boldsymbol{x}_2^*$, corresponding objective function values and membership function values are calculated as $p_1(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*) = 0.7392$, $p_2(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*) = 0.8290$, $\mu_1(p_1(\boldsymbol{x}_1^*,\boldsymbol{x}_2^*)) = 0.6535$, $\mu_2(p_2(\boldsymbol{x}_1^*,\boldsymbol{x}_2^*)) = 0.6529$. Then, the ratio of satisfactory degrees Δ is equal to 0.9991. Since DM1 is not satisfied with this solution, DM1 sets the minimal satisfactory level $\hat{\delta}$ for $\mu_1(p_1(\boldsymbol{x}_1,\boldsymbol{x}_2))$ to 0.70.

In step 5, (12) for $\hat{\delta} = 0.70$ is solved by GADSCRRSU. Then, the ratio of satisfactory degrees for the optimal solution is calculated as $\Delta = 0.9033$.

Following step 6, DM1 judges if he is satisfied with the solution obtained in step 5. Since the ratio of satisfactory

TABLE III
INTERACTION PROCESS

Interaction		1st	2nd	3rd
$\hat{\delta}$		0.70	0.80	0.75
$\mu_1(p_1(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*))$	0.6535	0.7007	0.8004	0.7528
$\mu_2(p_2(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*))$	0.6529	0.6330	0.5375	0.6121
$p_1(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*)$	0.7392	0.7622	0.8109	0.7876
$p_2(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*)$	0.8290	0.8192	0.7722	0.8089
Δ	0.9991	0.9033	0.6715	0.8131

degrees Δ is greater than $\Delta_{\max} = 0.85$, DM1 can not be satisfied with it and he updates the minimal satisfactory level $\hat{\delta}$ from 0.70 to 0.80.

Again in step 5, (12) for $\hat{\delta} = 0.80$ is solved by GAD-SCRRSU. Then, the ratio of satisfactory degrees for the optimal solution is calculated as $\Delta = 0.6715$.

In step 6, since the ratio of satisfactory degrees Δ is less than $\Delta_{\min} = 0.75$, DM1 can not be satisfied with it and he updates the minimal satisfactory level $\hat{\delta}$ from 0.80 to 0.75.

Again in step 5, (12) for $\hat{\delta} = 0.75$ is solved by GAD-SCRRSU. Then, the ratio of satisfactory degrees for the optimal solution is calculated as $\Delta = 0.8131$.

In step 6, since the ratio of satisfactory degrees Δ exists in the interval $[0.75, 0.85]$ and DM1 is satisfied with it, the satisfactory solution is obtained and the interaction procedure is stopped.

VII. Conclusions

In this paper, focusing on two-level integer programming problems involving random variable coefficients, we presented interactive fuzzy programming based on a probability maximization model. Since several integer programming problems in the proposed interactive fuzzy programming approachand showed its feasibility have to be solved, we adopt the genetic algorithm with double strings using continuous relaxation based on reference solution updating (GADSCRRSU) [10]. Furthermore, we showed its feasibility for a simple numerical example. As future problems, we are going to consider other stochastic programming models such as the expectation optimization model, the variance minimization model and so forth, and two-level integer programming problems involving fuzzy random variable coefficients.

References

[1] R.E. Bellman and L.A. Zadeh, Decision making in a fuzzy environment, Management Science, Vol. 17, pp. 141–164, 1970.

[2] A. Charnes and W.W. Cooper, Chance constrained programming, Management Science, Vol. 6, pp. 73–79, 1959.

[3] G.B. Dantzig, Linear programming under uncertainty, Management Science, Vol. 1, pp. 3–4, 1955.

[4] D.E. Goldberg R. and Lingle, Alleles, loci, and the traveling salesman problem, in: *Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications*, Lawrence Erlbaum Associates, Publishers, New Jersey, pp. 154–159, 1985.

[5] J.-P. Leclercq, Stochastic programming: an interactive multicriteria approach, European Journal of Operational Research, Vol. 10, pp. 33–41, 1982.

[6] M.K. Luhandjula, Multiple objective programming problems with possibilistic coefficients, Fuzzy Sets and Systems, Vol. 21, pp. 135–145, 1987.

[7] I.M. Stancu-Minasian, Overview of different approaches for solving stochastic programming problems with multiple objective functions, R. Slowinski and J. Teghem (eds.): Stochastic Versus Fuzzy Approaches to Multiobjective Mathematical Programming Under Uncertainty, Kluwer Academic Publishers, Dordrecht/Boston/London, pp. 71–101, 1990.

[8] H. Rommelfanger, Fuzzy linear programming and applications, European Journal of Operational Research, Vol. 92, pp. 512–527, 1996.

[9] M. Sakawa, Fuzzy Sets and Interactive Multiobjective Optimization, Plenum Press, New York, 1993.

[10] M. Sakawa, Genetic Algorithms and Fuzzy Multiobjective Optimization, Kluwer Academic Publishers, 2001.

[11] M. Sakawa, K. Kato and H. Katagiri, An interactive fuzzy satisficing method through a variance minimization model for multiobjective linear programming problems involving random variables, KES2002, Part 2, pp. 1222–1226, 2002.

[12] M. Sakawa, K. Kato and I. Nishizaki, An interactive fuzzy satisficing method for multiobjective stochastic linear programming problems through an expectation model, European Journal of Operational Research, Vol. 145, pp. 665–672, 2003.

[13] M. Sakawa, K. Kato, I. Nishizaki and M. Yoshioka, Interactive decision making for fuzzy multiobjective linear programming problems involving random variable coefficients, Proceedings of The Fourth Asian Fuzzy Systems Symposium, Vol. 1, pp. 392–397, 2000.

[14] M. Sakawa, K. Kato, H. Sunada and T. Shibano, Fuzzy programming for multiobjective 0-1 programming problems through revised genetic algorithms, European Journal of Operational Research, Vol. 97, pp. 149–158, 1997.

[15] 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.

[16] 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.

[17] M. Sakawa, H. Yano and T. Yumine, An interactive fuzzy satisficing method for multiobjective linear-programming problems and its application, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-17, pp. 654–661, 1987.

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

[19] R. Slowinski (ed.), Fuzzy Sets in Decision Analysis, Operations Research and Statistics, Kluwer Academic Publishers, Dordrecht/Boston/London, 1998.

[20] J. Teghem Jr., D. Dufrane, M. Thauvoye, and P. Kunsch, STRANGE: an interactive method for multi-objective linear programming under uncertainty, European Journal of Operational Research, Vol. 26, pp. 65–82, 1986.

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