

## Chapter 2

# Fundamentals

This chapter is devoted to reviewing optimization concepts and the related computational methods that will be used in the remaining chapters. In particular, we deal with three optimization concepts incorporating fuzziness and ambiguity of human judgments, uncertainty of events characterizing decision making problems, and multiplicity of evaluation criteria. Fuzzy programming which is developed in order to take into account fuzziness and ambiguity of human judgments, and this method is first presented together with the basic concepts in fuzzy set theory. Fundamentals of stochastic programming are also provided for decision problems under probabilistic uncertainty. Specifically, two-stage programming and chance constraint programming are covered. To meet expectations of diversified evaluations, multiobjective programming has been investigated. After presenting interactive methods for multiobjective linear programming problems, we give some techniques to deal with not only multiple objectives but also fuzzy goals for the objectives. In an organization with a hierarchical structure, we often find decision making with two or more decision makers (DMs) attempting to optimize their objective functions. To model such a decision making problem mathematically, a two-level programming problem is formulated, and cooperative and noncooperative solution methods are presented for two-level programming. Genetic algorithms are considered to be one of the most practical and proven meta heuristics for difficult classes of optimization problems, and finally basic concepts of genetic algorithms are provided.

### 2.1 Fuzzy programming

#### 2.1.1 Fuzzy sets

Throughout this book, the concepts of fuzzy set theory are used together with those of stochastic theory. Before discussing fuzzy programming, we provide the fundamentals of fuzzy set theory.

A fuzzy set initiated by Zadeh (1965) is defined as follows:

**Definition 2.1 (Fuzzy set).** Let  $X$  denote a universal set. Then, a fuzzy set  $\tilde{A}$  is defined by its membership function

$$\mu_{\tilde{A}} : X \rightarrow [0, 1]. \quad (2.1)$$

We deal with mathematical programming in this book, and therefore the set  $X$  in Definition 2.1 means the real line  $\mathbb{R}$  generally. The membership function  $\mu_{\tilde{A}}$  assigns to each element  $x \in X$  a real number  $\mu_{\tilde{A}}(x)$  in the interval  $[0, 1]$ , and the value of  $\mu_{\tilde{A}}(x)$  represents the grade of membership of  $x$  in  $\tilde{A}$ . A fuzzy set  $\tilde{A}$  is represented by a pair of an element  $x$  and its grade  $\mu_{\tilde{A}}(x)$ , and thus it is often written as

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) \mid x \in X\}. \quad (2.2)$$

For a given ordinary set  $A$ , the characteristic function

$$c_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A \end{cases} \quad (2.3)$$

defines the set  $A$  which is expressed by

$$A = \{x \in X \mid c_A(x) = 1\}. \quad (2.4)$$

From these definitions, one finds that a fuzzy set  $\tilde{A}$  is a natural extension of an ordinal set  $A$ .

The concept of  $\alpha$ -level sets is very important to serve a transfer relation between a fuzzy set and an ordinary set.

**Definition 2.2 ( $\alpha$ -level set).** For a given  $\alpha \in [0, 1]$ , the  $\alpha$ -level set of a fuzzy set  $\tilde{A}$  is defined as an ordinary set  $A_\alpha$  of elements  $x$  such that the membership function value  $\mu_{\tilde{A}}(x)$  of  $x$  exceeds  $\alpha$ , i.e.,

$$A_\alpha = \{x \mid \mu_{\tilde{A}}(x) \geq \alpha\}. \quad (2.5)$$

The extension principle introduced by Zadeh (1965) is to provide a general way for extending nonfuzzy mathematical concepts to the fuzzy framework.

**Definition 2.3 (Extension principle).** Let  $f : X \rightarrow Y$  be a mapping from a set  $X$  to a set  $Y$ . Then, the extension principle allows us to define the fuzzy set  $\tilde{B}$  in  $Y$  induced by the fuzzy set  $\tilde{A}$  in  $X$  through  $f$  as follows:

$$\tilde{B} = \{(y, \mu_{\tilde{B}}(y)) \mid y = f(x), x \in X\} \quad (2.6)$$

with

$$\mu_{\tilde{B}}(y) \triangleq \mu_{f(\tilde{A})}(y) = \begin{cases} \sup_{y=f(x)} \mu_{\tilde{A}}(x) & \text{if } f^{-1}(y) \neq \emptyset \\ 0 & \text{if } f^{-1}(y) = \emptyset, \end{cases} \quad (2.7)$$

where  $f^{-1}(y)$  is the inverse image of  $y$ , and  $\emptyset$  means the empty set.

Because we deal with mathematical programming under uncertainty, among fuzzy sets, fuzzy numbers which are linguistically-expressed such as “approximately  $m$ ” or “about  $n$ ” play important roles. Before defining fuzzy numbers, we give the definitions of convex and normalized fuzzy sets. A fuzzy set  $\tilde{A}$  is said to be convex if any  $\alpha$ -level set  $A_\alpha$  of  $\tilde{A}$  is convex, and a fuzzy set  $\tilde{A}$  is said to be normal if there is  $x$  such that  $\mu_{\tilde{A}}(x) = 1$ .

**Definition 2.4 (Fuzzy number).** A fuzzy number is a convex normalized fuzzy set of the real line  $\mathbb{R}$  whose membership function is piecewise continuous.

By using the extension principle by Zadeh, the binary operation “ $*$ ” in  $\mathbb{R}$  can be extended to the binary operation “ $\otimes$ ” of fuzzy numbers  $\tilde{M}$  and  $\tilde{N}$  as

$$\mu_{\tilde{M} \otimes \tilde{N}}(z) = \sup_{z=x*y} \min\{\mu_{\tilde{M}}(x), \mu_{\tilde{N}}(y)\}. \quad (2.8)$$

For example, consider an extension of addition “ $+$ ” of two numbers. By using (2.8), we can define the extended addition “ $\oplus$ ” of two fuzzy numbers as follows:

$$\begin{aligned} \mu_{\tilde{M} \oplus \tilde{N}}(z) &= \sup_{z=x+y} \min\{\mu_{\tilde{M}}(x), \mu_{\tilde{N}}(y)\} \\ &= \sup_{x \in \mathbb{R}} \min\{\mu_{\tilde{M}}(x), \mu_{\tilde{N}}(z-x)\}. \end{aligned}$$

To provide easy computation of fuzzy numbers, Dubois and Prade (1978) introduce the concept of  $L$ - $R$  fuzzy numbers.

**Definition 2.5 ( $L$ - $R$  fuzzy number).** A fuzzy number  $\tilde{M}$  is said to be an  $L$ - $R$  fuzzy number if

$$\mu_{\tilde{M}}(x) = \begin{cases} L\left(\frac{m-x}{\alpha}\right) & \text{if } x \leq m \\ R\left(\frac{x-m}{\beta}\right) & \text{if } x \geq m, \end{cases} \quad (2.9)$$

where  $m$  is the mean value of  $\tilde{M}$ , and  $\alpha$  and  $\beta$  are positive numbers which represent left and right spreads of the fuzzy number, respectively; a function  $L$  is a left shape function satisfying (i)  $L(x) = L(-x)$ , (ii)  $L(0) = 1$ , (iii)  $L(x)$  is nonincreasing on  $[0, \infty)$ ; and a right shape function  $R$  is similarly defined as  $L$ .

By using its mean, left and right spreads, and shape functions, an  $L$ - $R$  fuzzy number  $\tilde{M}$  is symbolically written as

$$\tilde{M} = (m, \alpha, \beta)_{LR}. \quad (2.10)$$

For two  $L$ - $R$  fuzzy numbers  $\tilde{M} = (m, \alpha, \beta)_{LR}$  and  $\tilde{N} = (n, \gamma, \delta)_{LR}$ , the extended addition  $\tilde{M} \oplus \tilde{N}$  is calculated as

$$(m, \alpha, \beta)_{LR} \oplus (n, \gamma, \delta)_{LR} = (m+n, \alpha+\gamma, \beta+\delta)_{LR}, \quad (2.11)$$

and scalar multiplication for an  $L$ - $R$  fuzzy numbers  $\tilde{M} = (m, \alpha, \beta)_{LR}$  and a scalar value  $\lambda$  is also given as

$$\lambda \otimes (m, \alpha, \beta)_{LR} = \begin{cases} (\lambda m, \lambda \alpha, \lambda \beta)_{LR} & \text{if } \lambda > 0 \\ (\lambda m, -\lambda \alpha, -\lambda \beta)_{LR} & \text{if } \lambda < 0, \end{cases} \quad (2.12)$$

where  $\otimes$  means the extended multiplication. The other operations are similarly given, and for further information on this issue, refer to Dubois and Prade (1978).

### 2.1.2 Fuzzy goals and Fuzzy constraints

Zimmermann (1976) introduces the concept of fuzzy set theory into linear programming. Assuming that the membership functions for fuzzy sets are linear, he shows that, by employing the principle of the fuzzy decision by Bellman and Zadeh (1970), a linear programming problem with a fuzzy goal and fuzzy constraints can be solved by using standard linear programming techniques.

A linear programming problem is represented as

$$\left. \begin{array}{l} \text{minimize } z(x_1, \dots, x_n) = c_1 x_1 + \dots + c_n x_n \\ \text{subject to } a_{11} x_1 + \dots + a_{1n} x_n \leq b_1 \\ \dots\dots\dots \\ a_{m1} x_1 + \dots + a_{mn} x_n \leq b_m \\ x_j \geq 0, j = 1, \dots, n, \end{array} \right\} \quad (2.13)$$

where  $x_j$  is a decision variable, and  $c_j$ ,  $a_{ij}$  and  $b_i$  are given coefficients of the objective function and the constraints. Let  $\mathbf{x} = (x_1, \dots, x_n)^T$  denote a column vector of the decision variables, and let  $\mathbf{c} = (c_1, \dots, c_n)$ ,  $A = [a_{ij}]$ ,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$  and  $\mathbf{b} = (b_1, \dots, b_m)^T$  denote an  $n$  dimensional row vector of the coefficients of the objective function, an  $m \times n$  matrix of the coefficients of the left-hand side of the constraints, and an  $m$  dimensional column vector of the coefficients of the right-hand side of the constraints, respectively; the superscript  $T$  means transposition of a vector or a matrix. Then, (2.13) is simply rewritten in the following vector and matrix representation:

$$\left. \begin{array}{l} \text{minimize } z(\mathbf{x}) = \mathbf{c}\mathbf{x} \\ \text{subject to } A\mathbf{x} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.14)$$

To a standard linear programming problem (2.14), taking into account the imprecision or fuzziness with respect to the judgment of a decision maker (DM), Zimmermann formulates the following linear programming problem with a fuzzy goal and fuzzy constraints:

$$cx \lesssim z_0 \quad (2.15)$$

$$Ax \lesssim b \quad (2.16)$$

$$x \geq 0, \quad (2.17)$$

where the symbol  $\lesssim$  denotes a relaxed or fuzzy version of the ordinary inequality  $\leq$ . From the DM's preference, the fuzzy goal (2.15) and the fuzzy constraints (2.16) mean that the objective function  $cx$  should be essentially smaller than or equal to a certain level  $z_0$ , and that the values of the constraints  $Ax$  should be substantially smaller than or equal to  $b$ , respectively. Assuming that the fuzzy goal and the fuzzy constraints are equally important, he employs the following unified formulation:

$$\left. \begin{array}{l} Bx \lesssim b' \\ x \geq 0, \end{array} \right\} \quad (2.18)$$

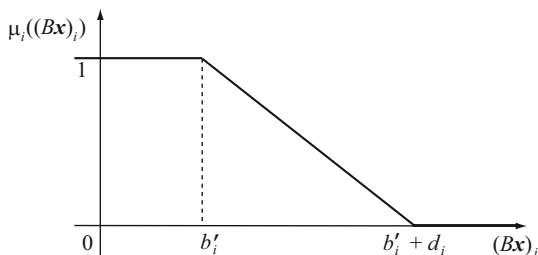
where

$$B = \begin{bmatrix} c \\ A \end{bmatrix}, \quad b' = \begin{bmatrix} z_0 \\ b \end{bmatrix}. \quad (2.19)$$

To express the imprecision or fuzziness of the DM's judgment, the  $i$ th fuzzy constraint  $(Bx)_i \lesssim b'_i$  is interpreted as the following linear membership function:

$$\mu_i((Bx)_i) = \begin{cases} 1 & \text{if } (Bx)_i \leq b'_i \\ 1 - \frac{(Bx)_i - b'_i}{d_i} & \text{if } b'_i < (Bx)_i \leq b'_i + d_i \\ 0 & \text{if } (Bx)_i > b'_i + d_i, \end{cases} \quad (2.20)$$

where  $d_i$  is a subjectively specified constant expressing the limit of the admissible violation of the  $i$ th constraint, and it is depicted in Fig. 2.1.



**Fig. 2.1** Linear membership function for the  $i$ th fuzzy constraint.

On the basis of the principle of the fuzzy decision by Bellman and Zadeh (1970), a mathematical programming problem for finding the maximum decision is represented as

$$\left. \begin{array}{l} \text{maximize } \min_{0 \leq i \leq m+1} \mu_i((Bx)_i) \\ \text{subject to } x \geq 0. \end{array} \right\} \quad (2.21)$$

With the variable transformation  $b_i'' = b_i'/d_i$  and  $(B'\mathbf{x})_i = (B\mathbf{x})_i/d_i$ , and an auxiliary variable  $\lambda$ , (2.21) can be transformed into the following conventional linear programming problem:

$$\left. \begin{array}{ll} \text{maximize} & \lambda \\ \text{subject to} & \lambda \leq 1 + b_i'' - (B'\mathbf{x})_i, \quad i = 0, \dots, m+1 \\ & \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.22)$$

Because the fuzzy decision is represented as  $\min_{0 \leq i \leq m+1} \mu_i((B\mathbf{x})_i)$  in (2.21), it is often called the minimum operator.

### 2.1.3 Linear programming problems with fuzzy parameters

To describe a real-world decision situation mathematically, the formulation of linear programming is often employed due to its simplicity and ease in computation, and naturally, possible values of the parameters in a linear programming problem may be assigned by experts for the decision situations. In most situations, however, it is observed that the experts may know the possible values of the parameters only imprecisely or ambiguously. From this reason, to express the imprecise judgments of the experts, the following linear programming problem involving fuzzy parameters is formulated:

$$\left. \begin{array}{ll} \text{minimize} & \tilde{C}_1 x_1 + \dots + \tilde{C}_n x_n \\ \text{subject to} & \tilde{A}_{11} x_1 + \dots + \tilde{A}_{1n} x_n \leq \tilde{B}_1 \\ & \dots\dots\dots \\ & \tilde{A}_{m1} x_1 + \dots + \tilde{A}_{mn} x_n \leq \tilde{B}_m \\ & x_j \geq 0, \quad j = 1, \dots, n, \end{array} \right\} \quad (2.23)$$

where  $\tilde{C}_j$ ,  $\tilde{A}_{ij}$ , and  $\tilde{B}_i$ ,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$  are fuzzy parameters represented by fuzzy numbers. Let the membership function of the fuzzy parameters  $\tilde{C}_j$ ,  $\tilde{A}_{ij}$ , and  $\tilde{B}_i$ ,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$  be denoted by  $\mu_{\tilde{C}_j}$ ,  $\mu_{\tilde{A}_{ij}}$ , and  $\mu_{\tilde{B}_i}$ ,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$ , respectively.

Here, we give two approaches for solving the linear programming problem with fuzzy numbers (2.23): one is an approach based on the possibility-based model (Dubois and Prade, 1980) and the other is the level set-based model (Sakawa, 1993).

#### 2.1.3.1 Possibility-based model

To deal with binary relations between a pair of fuzzy numbers  $\tilde{M}$  and  $\tilde{N}$  from the viewpoint of possibility, Dubois and Prade (1978, 1980) give the following index:

$$\text{Pos}(\tilde{M} \geq \tilde{N}) = \Pi_{\tilde{M}}([\tilde{N}, \infty)) = \sup\{\min\{\mu_{\tilde{M}}(u), \mu_{\tilde{N}}(v)\} \mid u \geq v\}. \quad (2.24)$$

The index (2.24) can be interpreted as the degree of possibility that  $\tilde{M}$  is larger than or equal to  $\tilde{N}$ .

For a given degree  $\alpha$ , the  $\alpha$ -level sets  $M_\alpha = \{u \in \mathbb{R} \mid \mu_{\tilde{M}}(u) \geq \alpha\}$  and  $N_\alpha = \{v \in \mathbb{R} \mid \mu_{\tilde{N}}(v) \geq \alpha\}$  of the fuzzy numbers  $\tilde{M}$  and  $\tilde{N}$  are represented by closed intervals  $M_\alpha = [m_\alpha^L, m_\alpha^R]$  and  $N_\alpha = [n_\alpha^L, n_\alpha^R]$ , respectively. Then, it is known that

$$\text{Pos}(\tilde{M} \geq \tilde{N}) \geq \alpha \text{ if and only if } m_\alpha^R \geq n_\alpha^L. \quad (2.25)$$

Using this binary relation from the viewpoint of possibility, if the DM specifies a certain degree  $\alpha$  of possibility, the constraints of (2.23) can be interpreted as

$$\mathbf{x} \in X_{\text{pos}}(\alpha) \triangleq \{\mathbf{x} \geq \mathbf{0} \mid \text{Pos}(\tilde{A}_{i1}x_1 + \dots + \tilde{A}_{in}x_n \leq \tilde{B}_i) \geq \alpha, i = 1, \dots, m\}. \quad (2.26)$$

The  $\alpha$ -level set of the fuzzy number  $\tilde{B}_i$  for the right-hand side constant in the  $i$ th constraint is represented by

$$B_{i\alpha} = \{u \in \mathbb{R} \mid \mu_{\tilde{B}_i}(u) \geq \alpha\} = [b_{i\alpha}^L, b_{i\alpha}^R]. \quad (2.27)$$

Similarly, the  $\alpha$ -level set of the vector of fuzzy numbers  $\tilde{A}_i = (\tilde{a}_{i1}, \dots, \tilde{a}_{in})$  in the left-hand side coefficients of the  $i$ th constraint is represented by

$$A_{i\alpha} = \{(a_{i1}, \dots, a_{in}) \mid a_{ij} \in [a_{ij\alpha}^L, a_{ij\alpha}^R], j = 1, \dots, n\}, \quad (2.28)$$

and, (2.28) is simply denoted by

$$A_{i\alpha} = [a_{i\alpha}^L, a_{i\alpha}^R]. \quad (2.29)$$

From the proposition (2.25), (2.26) can be represented as

$$\mathbf{x} \in X_{\text{pos}}(\alpha) = \{\mathbf{x} \geq \mathbf{0} \mid a_{i\alpha}^L \mathbf{x} \leq b_{i\alpha}^R, i = 1, \dots, m\}. \quad (2.30)$$

For the sake of simplicity, assume that all the fuzzy parameters in (2.23) are represented by  $L$ - $R$  fuzzy numbers:  $\tilde{C}_j = (c_j, \beta_j, \gamma_j)_{LR}$ ,  $j = 1, \dots, n$ ,  $\tilde{A}_{ij} = (a_{ij}, \delta_{ij}, \epsilon_{ij})_{LR}$ ,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$ ,  $\tilde{B}_i = (b_i, \zeta_i, \eta_i)_{LR}$ ,  $i = 1, \dots, m$ ; and  $L(y) = R(y) = \max\{0, 1 - |y|\}$ .

Then, (2.30) can be rewritten by ordinary linear inequalities

$$\mathbf{x} \in X_{\text{pos}}(\alpha) = \left\{ \mathbf{x} \geq \mathbf{0} \mid \sum_{j=1}^n \{a_{ij} - (1 - \alpha)\delta_{ij}\}x_j \leq b_i + (1 - \alpha)\eta_i, i = 1, \dots, m \right\}. \quad (2.31)$$

As for the objective function of (2.23), we introduce the following linear membership function of the fuzzy goal:

$$\mu_{\tilde{G}}(y) = \begin{cases} 1 & \text{if } y < z^1 \\ 1 - \frac{y - z^0}{z^1 - z^0} & \text{if } z^1 \leq y \leq z^0 \\ 0 & \text{if } y \geq z^0, \end{cases} \quad (2.32)$$

where  $z^0$  and  $z^1$  denote the values of the objective function such that the degrees of the membership function  $\mu_{\tilde{G}}$  are 0 and 1, respectively.

Because from the assumption of the fuzzy parameters the objective function  $\tilde{C}_1x_1 + \dots + \tilde{C}_nx_n$  of (2.23) can be represented by  $(\sum_{j=1}^n c_jx_j, \sum_{j=1}^n \beta_jx_j, \sum_{j=1}^n \gamma_jx_j)_{LR}$ , with appropriate values of the parameters  $z^0$  and  $z^1$  of the fuzzy goal (2.32), the degree of possibility with respect to the objective function is expressed as

$$\Pi_{\tilde{C}_x}(\tilde{G}) = \sup \min\{\mu_{\tilde{C}_x}(y), \mu_{\tilde{G}}(y)\} = \frac{\sum_{j=1}^n (\beta_j - c_j)x_j + z^0}{\sum_{j=1}^n \beta_jx_j - z^1 + z^0}, \quad (2.33)$$

where  $\mu_{\tilde{C}_x}$  is a membership function of the objective function for a given vector  $x$  of the decision variables.

Under the assumption of the fuzzy parameters, a linear programming problem with fuzzy parameters in the possibility-based model is formulated as

$$\left. \begin{array}{l} \text{maximize} \quad \frac{\sum_{j=1}^n (\beta_j - c_j)x_j + z^0}{\sum_{j=1}^n \beta_jx_j - z^1 + z^0} \\ \text{subject to} \quad \sum_{j=1}^n \{a_{1j} - (1 - \alpha)\delta_{1j}\}x_j \leq b_1 + (1 - \alpha)\eta_1 \\ \quad \dots\dots\dots \\ \sum_{j=1}^n \{a_{mj} - (1 - \alpha)\delta_{mj}\}x_j \leq b_m + (1 - \alpha)\eta_m \\ x_j \geq 0, \quad j = 1, \dots, n. \end{array} \right\} \quad (2.34)$$

Because (2.34) is a linear fractional programming problem, it can be solved by using the Charnes and Cooper method (1962) or the Bitran and Novaes method (1973).

### 2.1.3.2 Level set-based model

Consider the level set-based model for solving a linear programming problem involving fuzzy parameters (Sakawa, 1993). At the beginning, we give the formulation (2.23) of a linear programming problem involving fuzzy parameters once again.

$$\begin{array}{ll} \text{minimize} & \tilde{C}_1x_1 + \dots + \tilde{C}_nx_n \\ \text{subject to} & \tilde{A}_{11}x_1 + \dots + \tilde{A}_{1n}x_n \leq \tilde{B}_1 \\ & \dots\dots\dots \\ & \tilde{A}_{m1}x_1 + \dots + \tilde{A}_{mn}x_n \leq \tilde{B}_m \\ & x_j \geq 0, \quad j = 1, \dots, n. \end{array}$$

For a certain degree  $\alpha$  specified by the DM, the  $\alpha$ -level set of all of the fuzzy numbers  $\tilde{C}_j$ ,  $\tilde{A}_{ij}$ , and  $\tilde{B}_i$ ,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$  in (2.23) is defined as the ordinary set  $(C, A, B)_\alpha$  for which their membership function values exceed the degree  $\alpha$ , i.e.,

$$(C, A, B)_\alpha = \{(\mathbf{c}, \mathbf{a}, \mathbf{b}) \mid \mu_{\tilde{C}_j} \geq \alpha, \mu_{\tilde{A}_{ij}} \geq \alpha, \mu_{\tilde{B}_i} \geq \alpha, i = 1, \dots, m, j = 1, \dots, n\}, \quad (2.35)$$

where  $\mathbf{c}$ ,  $\mathbf{a}$ , and  $\mathbf{b}$  are an  $n$  dimensional row constant coefficient vector for the objective function, an  $m \times n$  constant coefficient matrix for the left-hand side of the constraints, and an  $m$  dimensional column right-hand side constant vector for the constraints, respectively. It should be noted here that the  $\alpha$ -level sets have the following property:

$$\alpha^1 \leq \alpha^2 \Leftrightarrow (C, A, B)_{\alpha^1} \supseteq (C, A, B)_{\alpha^2}. \quad (2.36)$$

Suppose the DM thinks that the degree of all of the membership functions of the fuzzy numbers involved in (2.23) should be larger than or equal to a certain degree  $\alpha$ . Then, for such a degree  $\alpha$ , (2.23) can be interpreted as the following nonfuzzy linear programming problem which depends on the coefficient vector  $(\mathbf{c}, \mathbf{a}, \mathbf{b}) \in (C, A, B)_\alpha$ :

$$\left. \begin{array}{ll} \text{minimize} & c_1x_1 + \dots + c_nx_n \\ \text{subject to} & a_{11}x_1 + \dots + a_{1n}x_n \leq b_1 \\ & \dots\dots\dots \\ & a_{m1}x_1 + \dots + a_{mn}x_n \leq b_m \\ & x_j \geq 0, \quad j = 1, \dots, n. \end{array} \right\} \quad (2.37)$$

Observe that there exist an infinite number of linear programming problems such as (2.37) depending on the coefficient vector  $(\mathbf{c}, \mathbf{a}, \mathbf{b}) \in (C, A, B)_\alpha$ , and the values of  $(\mathbf{c}, \mathbf{a}, \mathbf{b})$  are arbitrary for any  $(\mathbf{c}, \mathbf{a}, \mathbf{b}) \in (C, A, B)_\alpha$  in the sense that the degree of all of the membership functions for fuzzy numbers in the linear programming problem involving fuzzy parameters (2.23) exceeds the degree  $\alpha$  specified by the DM. However, if possible, it would be desirable for the DM to choose  $(\mathbf{c}, \mathbf{a}, \mathbf{b}) \in (C, A, B)_\alpha$  in (2.37) so as to minimize the objective function under the constraints. From such a point of view, for the given degree  $\alpha$ , it seems to be quite natural to have the linear programming problem involving fuzzy parameters (2.23) as the following nonfuzzy  $\alpha$ -linear programming problem:

$$\left. \begin{array}{ll} \text{minimize} & c_1x_1 + \dots + c_nx_n \\ \text{subject to} & a_{11}x_1 + \dots + a_{1n}x_n \leq b_1 \\ & \dots\dots\dots \\ & a_{m1}x_1 + \dots + a_{mn}x_n \leq b_m \\ & x_j \geq 0, \quad j = 1, \dots, n \\ & (\mathbf{c}, \mathbf{a}, \mathbf{b}) \in (C, A, B)_\alpha. \end{array} \right\} \quad (2.38)$$

It should be emphasized here that in the nonfuzzy  $\alpha$ -linear programming problem (2.38), the parameters  $(\mathbf{c}, \mathbf{a}, \mathbf{b})$  are treated as decision variables rather than constants of the coefficients.

From the property of the  $\alpha$ -level set for the fuzzy numbers  $\tilde{C}_j$ ,  $\tilde{A}_{ij}$ , and  $\tilde{B}_i$ ,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$ , the feasible regions for  $c_j$ ,  $a_{ij}$ , and  $b_i$  can be denoted by the closed intervals  $[c_{j\alpha}^L, c_{j\alpha}^R]$ ,  $[a_{ij\alpha}^L, a_{ij\alpha}^R]$ , and  $[b_{i\alpha}^L, b_{i\alpha}^R]$ , respectively. Therefore, we can obtain an optimal solution to the nonfuzzy  $\alpha$ -linear programming problem (2.38) by solving the following linear programming problem:

$$\left. \begin{array}{ll} \text{minimize} & c_1^L x_1 + \dots + c_n^L x_n \\ \text{subject to} & a_{11}^L x_1 + \dots + a_{1n}^L x_n \leq b_1^R \\ & \dots\dots\dots \\ & a_{m1}^L x_1 + \dots + a_{mn}^L x_n \leq b_m^R \\ & x_j \geq 0, \quad j = 1, \dots, n. \end{array} \right\} \quad (2.39)$$

It is important to realize here that (2.39) is no more nonlinear but an ordinary linear programming problem, and consequently it is easy to solve it by using linear programming techniques such as the simplex method.

## 2.2 Stochastic programming

### 2.2.1 Random variables

An outcome of a random trial such as coin tossing is called a sample point, which is denoted by  $\omega$ , and the set of all of the possible outcomes of the random trial is called a sample space, which is denoted by  $\Omega$ . An event is a subset of the sample space.

For any events  $A$  and  $B$ , the union and the intersection are defined as follows:

(i) Union of  $A$  and  $B$ :

$$A \cup B = \{\omega \mid \omega \in A \text{ or } \omega \in B\}.$$

(ii) Intersection of  $A$  and  $B$ :

$$A \cap B = \{\omega \mid \omega \in A \text{ and } \omega \in B\}.$$

Furthermore, for any event  $A$ , the compliment is defined as

(iii) Compliment of  $A$ :

$$A^c = \{\omega \mid \omega \notin A\}.$$

A family  $\mathfrak{B}$  of subsets of the sample space  $\Omega$  which has the following properties is called a  $\sigma$ -field:

- (i)  $\Omega \in \mathfrak{B}$ ;
- (ii)  $A \in \mathfrak{B} \Rightarrow A^c \in \mathfrak{B}$ ;
- (iii)  $A_1, A_2, \dots \in \mathfrak{B} \Rightarrow \bigcup_{k=1}^{\infty} A_k \in \mathfrak{B}$ .

Let  $\mathfrak{S}$  be a family of subsets of the sample space  $\Omega$ . The intersection of all  $\sigma$ -fields containing  $\mathfrak{S}$  is called the  $\sigma$ -field generated by  $\mathfrak{S}$ , and it is denoted by  $\sigma[\mathfrak{S}]$ .

Let  $E = (E, d)$  be a metric space, and let  $\mathfrak{D}$  be a family of all open subsets of  $E$ . Then, a  $\sigma$ -field  $\sigma[\mathfrak{D}]$  generated by  $\mathfrak{D}$  is called a family of Borel sets on  $E$ . Consider the  $d$ -dimensional Euclidean space  $\mathbb{R}^d$  as a special case of  $E$ . A family of all open subsets of  $\mathbb{R}^d$  is defined as

$$\mathfrak{J}_d \triangleq \{(a_1, b_1] \times \cdots \times (a_d, b_d] \subset \mathbb{R}^d \mid -\infty \leq a_k \leq b_k \leq +\infty, k = 1, \dots, d\}. \quad (2.40)$$

Then,  $\sigma[\mathfrak{J}_d]$  is a  $\sigma$ -field in  $\mathbb{R}^d$ , and it is called a family of  $d$ -dimensional Borel sets on  $\mathbb{R}^d$ .

Let  $\Omega$  be a sample space, and let  $\mathfrak{B}$  be a  $\sigma$ -field in the sample space  $\Omega$ . A real-valued function  $P$  defined on the  $\sigma$ -field  $\mathfrak{B}$  in  $\Omega$  satisfying the following conditions is called a probability measure:

- (i)  $0 \leq P(A) \leq 1$  for  $A \in \mathfrak{B}$ ;
- (ii)  $P(\Omega) = 1$ ;
- (iii) If  $A_1, A_2, \dots \in \mathfrak{B}$  and  $A_1, A_2, \dots$  is a disjoint sequence, then

$$P\left(\bigcup_{k=1}^{\infty} A_k\right) = \sum_{k=1}^{\infty} P(A_k).$$

If  $\mathfrak{B}$  be a  $\sigma$ -field in a sample space  $\Omega$  and  $P$  is a probability measure on  $\mathfrak{B}$ , the triple  $(\Omega, \mathfrak{B}, P)$  is called a probability measure space or simply a probability space. For an arbitrary probability space  $(\Omega, \mathfrak{B}, P)$ , let  $X(\omega)$  be a real-valued function on  $\Omega$ . Then,  $X(\omega)$  is a random variable if

$$\{\omega \mid X(\omega) \leq x\} \in \mathfrak{B} \quad (2.41)$$

holds for each real value  $x$ . For the random variable  $X(\omega)$ , the function

$$F(x) = P(\{\omega \mid X(\omega) \leq x\}) \quad (2.42)$$

is called a distribution function, where  $P(\{\omega \mid X(\omega) \leq x\})$  is often simply denoted by  $P(X \leq x)$ . The distribution function has the following properties:

- (i)  $\lim_{x \rightarrow -\infty} F(x) = 0$  and  $\lim_{x \rightarrow +\infty} F(x) = 1$ ;
- (ii) If  $x < y$ , then  $F(x) \leq F(y)$ .

In this book, henceforth to discriminate a random variable from the other variables or parameters, we attach a bar “ $\bar{\cdot}$ ” to a character like  $\bar{d}$ .

### 2.2.2 Two-stage programming

In the real-world decision making problems, some stochastic events may influence elements characterizing decision making problems such as demands of products,

the amount of available resources and so forth. When such a decision making problem under uncertainty is formulated as a linear programming problem, it may be difficult that the constraints of the problem always hold completely. Then, a shortage or an excess comes from the violation of the constraints, and the corresponding penalties are imposed as the occasion demands. From this point of view, two-stage programming had been investigated from the beginning of the development of linear programming (Beale, 1955; Dantzig, 1955; Wets, 1966; Everitt and Ziemba, 1978).

To understand the framework of two-stage programming, consider a decision problem of a manufacturing company. Let  $\mathbf{x} = (x_1, \dots, x_n)^T$  denote activity levels in a production plant of the company, and then  $\mathbf{w} = T\mathbf{x}$  denotes the amount of products, where  $T$  is an  $m \times n$  matrix transforming the  $n$  kinds of activity levels into the  $m$  types of products. Let  $\bar{d}_i$  be the demand for the  $i$ th product which is only known in probability, and assume that the random variable  $\bar{d}_i$  of the demand is indicated by the probability distribution function  $F_i(d_i) \triangleq P(\bar{d}_i \leq d_i)$ . Suppose that the DM selects the activity levels, say  $\mathbf{x} = \hat{\mathbf{x}}$ , and after the occurrence of the random event the demands  $\bar{\mathbf{d}} = (\bar{d}_1, \dots, \bar{d}_m)^T$  are fixed at  $\bar{\mathbf{d}}$ . Then, if  $\hat{y}_i^+ \triangleq \bar{d}_i - \hat{w}_i \geq 0$ , the shortage of the  $i$ th product is  $\hat{y}_i^+$ , and if  $\hat{y}_i^- \triangleq \hat{w}_i - \bar{d}_i \geq 0$ , the excess of the  $i$ th product is  $\hat{y}_i^-$ . This situation can be formulated as

$$T\mathbf{x} + I\mathbf{y}^+ - I\mathbf{y}^- = \bar{\mathbf{d}},$$

where  $\mathbf{y}^+$  and  $\mathbf{y}^-$  represent the errors for estimating the demands, and  $I$  is the  $m$  dimensional identity matrix. Let  $\mathbf{q}^+$  and  $\mathbf{q}^-$  denote the penalty costs for making these errors, and let  $\mathbf{c}$  be the original costs for the activities in the production plant. Then, the objective function to be minimized may be the expectation of

$$\mathbf{c}\mathbf{x} + \mathbf{q}^+\mathbf{y}^+ + \mathbf{q}^-\mathbf{y}^-.$$

Adding the constraints

$$A\mathbf{x} \leq \mathbf{b}$$

for the activity levels such as the capacity, budget, technology, etc., we can formulate the standard form of the two-stage programming problem as

$$\left. \begin{array}{ll} \text{minimize} & \mathbf{c}\mathbf{x} + E[\mathbf{q}^+\mathbf{y}^+ + \mathbf{q}^-\mathbf{y}^-] \\ \text{subject to} & T\mathbf{x} + I\mathbf{y}^+ - I\mathbf{y}^- = \bar{\mathbf{d}} \\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \right\} \quad (2.43)$$

where  $E$  means the function of expectation. In particular, a two-stage programming problem such that the coefficient vector of  $\mathbf{y}^+$  and  $\mathbf{y}^-$  is the identity matrix such as (2.43) is called a simple recourse problem.

For the selected  $\mathbf{x}$  and realized values  $\bar{\mathbf{d}}$ , the following solution minimizes the objective function of (2.43) unless  $q_i^+ + q_i^-$  is negative, and therefore we assume that  $q_i^+ + q_i^- \geq 0$ ,  $i = 1, \dots, m$ :

$$\left. \begin{aligned} y_i^+ &= \hat{d}_i - \sum_{j=1}^n t_{ij}x_j, & y_i^- &= 0 & \text{if } \hat{d}_i &\geq \sum_{j=1}^n t_{ij}x_j \\ y_i^+ &= 0, & y_i^- &= \sum_{j=1}^n t_{ij}x_j - \hat{d}_i & \text{if } \hat{d}_i < \sum_{j=1}^n t_{ij}x_j, \end{aligned} \right\} \quad (2.44)$$

where  $t_{ij}$  is the  $ij$ -element of  $T$ .

Assume that the random variables  $\bar{d}_i$ ,  $i = 1, \dots, m$  are independent each other. From the independence of the random variables and (2.44), we can calculate the second term of the objective function of (2.43) as follows:

$$\begin{aligned} E[q^+y^+ + q^-y^-] &= \sum_{i=1}^m q_i^+ \int_{\sum_{j=1}^n t_{ij}x_j}^{\infty} \left( d_i - \sum_{j=1}^n t_{ij}x_j \right) dF_i(d_i) + \sum_{i=1}^m q_i^- \int_{-\infty}^{\sum_{j=1}^n t_{ij}x_j} \left( \sum_{j=1}^n t_{ij}x_j - d_i \right) dF_i(d_i) \\ &= \sum_{i=1}^m q_i^+ \left( E[\bar{d}_i] - \sum_{j=1}^n t_{ij}x_j \right) + \sum_{i=1}^m (q_i^+ + q_i^-) \left( \sum_{j=1}^n t_{ij}x_j \right) F_i \left( \sum_{j=1}^n t_{ij}x_j \right) \\ &\quad - \sum_{i=1}^m (q_i^+ + q_i^-) \int_{-\infty}^{\sum_{j=1}^n t_{ij}x_j} d_i dF_i(d_i). \end{aligned} \quad (2.45)$$

Thus, (2.43) can be transformed into the problem

$$\left. \begin{aligned} \text{minimize } & \mathbf{c}\mathbf{x} + \sum_{i=1}^m q_i^+ \left( E[\bar{d}_i] - \sum_{j=1}^n t_{ij}x_j \right) + \sum_{i=1}^m (q_i^+ + q_i^-) \\ & \left\{ \left( \sum_{j=1}^n t_{ij}x_j \right) F_i \left( \sum_{j=1}^n t_{ij}x_j \right) - \int_{-\infty}^{\sum_{j=1}^n t_{ij}x_j} d_i dF_i(d_i) \right\} \\ \text{subject to } & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \right\} \quad (2.46)$$

Let  $z_i$  be the expression in the brace of the third term of the objective function of (2.46), i.e.,

$$z_i \triangleq \left( \sum_{j=1}^n t_{ij}x_j \right) F_i \left( \sum_{j=1}^n t_{ij}x_j \right) - \int_{-\infty}^{\sum_{j=1}^n t_{ij}x_j} d_i dF_i(d_i), \quad (2.47)$$

and let  $f_i$  denote the probability density function for  $F_i$ . Then, the partial differential of  $z_i$  for  $x_j$  and  $x_k$  can be calculated as

$$\frac{\partial z_i}{\partial x_j \partial x_k} = t_{ij} t_{ik} f_i(t_{ij}x_j). \quad (2.48)$$

The Hessian matrix for  $z_i$  can be written as

$$f_i(t_{ij}x_j) \begin{pmatrix} t_{i1}^2 & \cdots & t_{i1}t_{in} \\ \vdots & \ddots & \vdots \\ t_{in}t_{i1} & \cdots & t_{in}^2 \end{pmatrix}, \quad (2.49)$$

and because it is positive semidefinite, one finds that  $z_i$  is convex. From this fact and linearity of the first and the second terms of the objective function of (2.46), (2.46) is a convex programming problem, and then it can be solved by a conventional convex programming techniques such as the sequential quadratic programming method (Fletcher, 1980; Gill, Murray and Wright, 1981; Powell, 1983).

### 2.2.3 Chance constraint programming

As shown in the previous subsection, in two-stage programming, some penalties are imposed for constraint violations. For decision problems under probabilistic uncertainty, from a different viewpoint, Charnes and Cooper (1963) propose chance constraint programming which admits random data variations and permits constraint violations up to specified probability limits.

Let  $\bar{A}$  and  $\bar{b}$  denote an  $m \times n$  coefficient matrix of the left-hand side of the constraints and an  $m$  dimensional column vector of the right-hand side of the constraints, respectively; and suppose that some or all of the elements of  $\bar{A}$  and  $\bar{b}$  are random variables. Then, for a given vector  $\alpha$  of probabilities, the chance constraint formulation for the constraint  $\bar{A}x \leq \bar{b}$  of a linear programming problem is represented as

$$P(\bar{A}x \leq \bar{b}) \geq \alpha, \quad (2.50)$$

where  $P$  means a probability measure, and the vector  $\alpha = (\alpha_1, \dots, \alpha_m)^T$  are probabilities of the extents to which constraint violations are admitted. Then, the element  $\alpha_i$  is associated with the  $i$ th constraint  $\sum_{j=1}^n a_{ij}x_j \leq b_i$ , and the  $i$ th constraint is interpreted as

$$P\left(\sum_{j=1}^n \bar{a}_{ij}x_j \leq \bar{b}_i\right) \geq \alpha_i. \quad (2.51)$$

The inequality (2.51) means that the  $i$ th constraint may be violated, but at most  $\beta_i = 1 - \alpha_i$  proportion of the time.

First, assume that only  $\bar{b}_i$  in the right-hand side of the chance constraint condition (2.51) is a random variable and  $\bar{a}_{ij}$  is a constant. Therefore, we use the notation:  $\bar{a}_{ij} = a_{ij}$ . Let  $F_i(\tau)$  denote the probability distribution function of  $\bar{b}_i$ . From the fact that

$$P\left(\sum_{j=1}^n a_{ij}x_j \leq \bar{b}_i\right) = 1 - F_i\left(\sum_{j=1}^n a_{ij}x_j\right),$$

the chance constraint condition (2.51) can be rewritten as

$$F_i\left(\sum_{j=1}^n a_{ij}x_j\right) \leq 1 - \alpha_i. \quad (2.52)$$

Let  $K_{1-\alpha_i}$  denote the maximum of  $\tau$  such that  $\tau = F_i^{-1}(1 - \alpha_i)$ , and then the inequality (2.52) can be simply expressed as

$$\sum_{j=1}^n a_{ij}x_j \leq K_{1-\alpha_i}. \quad (2.53)$$

Second, consider a more general case where not only  $\bar{b}_i$  but also  $\bar{a}_{ij}$  in the left-hand side of (2.51) are random variables, and specifically we assume that  $\bar{b}_i$  and  $\bar{a}_{ij}$  are normal random variables. Let  $m_{\bar{b}_i}$  and  $\sigma_{\bar{b}_i}^2$  be the mean and the variance of  $\bar{b}_i$ , respectively, and let  $m_{\bar{a}_{ij}}$  and  $V$  be the mean and the variance-covariance matrix of  $\bar{a}_{ij}$ , respectively. Moreover, assume that  $\bar{b}_i$  and  $\bar{a}_{ij}$  are independent. Then, the chance constraint condition (2.51) can be transformed into

$$\sum_{j=1}^n m_{\bar{a}_{ij}}x_j - \Phi^{-1}(1 - \alpha_i)\sqrt{\sigma_{\bar{b}_i}^2 + \mathbf{x}^T V \mathbf{x}} \leq m_{\bar{b}_i}, \quad (2.54)$$

where  $\Phi$  is the standardized normal distribution function with parameter  $(0, 1)$ .

Charnes and Cooper (1963) also consider three types of decision rules for optimizing objective functions with random variables: (i) the minimum or maximum expected value model, (ii) the minimum variance model, and (iii) the maximum probability model, which are referred to as the E-model, the V-model, and the P-model, respectively. Moreover, Kataoka (1963) and Geoffrion (1967) individually propose the fractile criterion model.

Let  $\bar{\mathbf{c}} = (\bar{c}_1, \dots, \bar{c}_n)$  denote an  $n$  dimensional coefficient row vector of the objective function, and suppose that some or all of coefficients  $\bar{c}_j$ ,  $j = 1, \dots, n$  are random variables. Then, the objective function in the E-model is represented as

$$E[\bar{\mathbf{c}}\mathbf{x}] = E\left[\sum_{j=1}^n \bar{c}_j x_j\right], \quad (2.55)$$

where  $E$  means the function of expectation. Let  $m_i$  denote the mean value of  $\bar{c}_j$ , and then the objective function of the E-model is simply written as

$$E\left[\sum_{j=1}^n \bar{c}_j x_j\right] = \sum_{j=1}^n m_j x_j. \quad (2.56)$$

The realization value of the objective function may vary quite widely even if the expected value of the objective function is minimized. In such a case, it may be suspicious if a plan based on the solution of the E-model would work well because uncertainty is large. Some DMs would prefer to plan with lower uncertainty. To meet this demand, the objective function may be formulated in the V-model as

$$Var[\bar{\mathbf{c}}\mathbf{x}] = Var\left[\sum_{j=1}^n \bar{c}_j x_j\right], \quad (2.57)$$

where  $Var$  means the function of variance. Let  $V$  denote an  $n \times n$  variance-covariance matrix for the vector of the random variables  $\bar{\mathbf{c}}$ , then the objective function of the V-model can be calculated as

$$Var \left[ \sum_{j=1}^n \bar{c}_j x_j \right] = \mathbf{x}^T V \mathbf{x}. \quad (2.58)$$

In the P-model, the probability that the objective function value is smaller than a certain target value is maximized, and then the objective function of the P-model is represented as

$$P(\bar{\mathbf{c}}\mathbf{x} \leq f_0), \quad (2.59)$$

where  $f_0$  is a given target value for the objective function.

The fractile criterion model is considered as complementary to the P-model; a target variable to the objective function is minimized, provided that the probability that the objective function value is smaller than the target variable is guaranteed to be larger than a given assured level. Then, the objective function of the fractile criterion model is represented as

$$f \text{ subject to } P(\bar{\mathbf{c}}\mathbf{x} \leq f) \geq \alpha, \quad (2.60)$$

where  $f$  and  $\alpha$  are the target variable to the objective function and the given assured level for the probability that the objective function value is smaller than the target variable.

## 2.3 Multiobjective programming

### 2.3.1 Multiobjective programming problem

A mathematical programming problem to optimize multiple conflicting linear objective functions simultaneously under given linear constraints is called a multiobjective linear programming problem. Let  $\mathbf{c}_i = (c_{i1}, \dots, c_{in})$ ,  $i = 1, \dots, k$  denote an  $n$  dimensional coefficient row vector of the  $i$ th objective function. Then, the multiobjective linear programming problem is represented as

$$\left. \begin{array}{l} \text{minimize } z_1(\mathbf{x}) = c_{11}x_1 + \dots + c_{1n}x_n \\ \dots\dots\dots \\ \text{minimize } z_k(\mathbf{x}) = c_{k1}x_1 + \dots + c_{kn}x_n \\ \text{subject to } a_{11}x_1 + \dots + a_{1n}x_n \leq b_1 \\ \dots\dots\dots \\ a_{m1}x_1 + \dots + a_{mn}x_n \leq b_m \\ x_j \geq 0, \quad j = 1, \dots, n. \end{array} \right\}$$

Alternatively, it is expressed by

$$\left. \begin{array}{l} \text{minimize } \mathbf{z}(\mathbf{x}) = \mathbf{C}\mathbf{x} \\ \text{subject to } \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0}, \end{array} \right\} \quad (2.61)$$

where  $\mathbf{C}$  denote a  $k \times n$  coefficient matrix of the objective functions.

First, we give the notion of optimality in a multiobjective linear programming problem. Because there does not always exist a solution minimizing all of the objective functions simultaneously, the solution concept of Pareto optimality plays an important role in multiobjective optimization, and it is defined as follows. Let  $X$  denote the nonempty set of all feasible solutions of (2.61), i.e.,  $X \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$ .

**Definition 2.6 (Pareto optimal solution).** A point  $\mathbf{x}^* \in X$  is said to be a *Pareto optimal solution* if and only if there does not exist another  $\mathbf{x} \in X$  such that  $z_i(\mathbf{x}) \leq z_i(\mathbf{x}^*)$  for all  $i \in \{1, \dots, k\}$  and  $z_j(\mathbf{x}) < z_j(\mathbf{x}^*)$  for at least one  $j \in \{1, \dots, k\}$ .

By substituting the strict inequality  $<$  for the inequality  $\leq$  in Definition 2.6, weak Pareto optimality is defined as a slightly weaker solution concept.

### 2.3.2 Interactive multiobjective programming

As seen from Definition 2.6, in general there exist an infinite number of Pareto optimal solutions if the feasible region  $X$  is not empty. In real-world decision making problems, to make a reasonable decision or implement a desirable scheme, the DM should select one point from among the set of Pareto optimal solutions. To meet this demand, several interactive multiobjective programming methods were developed from the 1970s to the 1980s, and it is known that the reference point method developed by Wierzbicki (1980) is relatively practical.

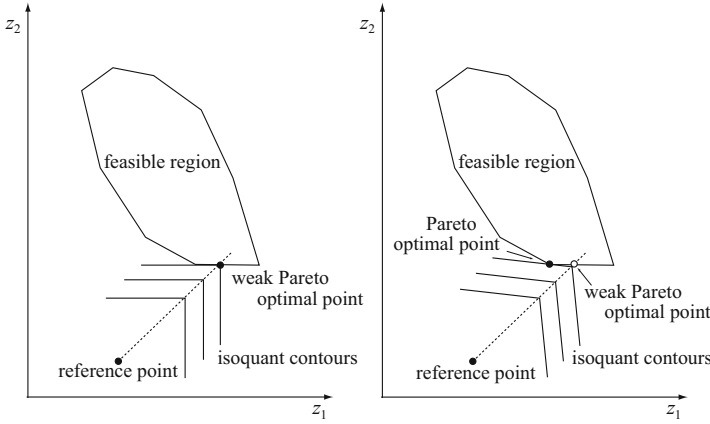
For each of the objective functions  $\mathbf{z}(\mathbf{x}) = (z_1(\mathbf{x}), \dots, z_k(\mathbf{x}))^T$  in (2.61), the DM specifies a reference point  $\hat{\mathbf{z}} = (\hat{z}_1, \dots, \hat{z}_k)^T$  which reflects the desired values of the objective functions, and it is thought that by changing the reference points in the interactive solution procedure, the DM can perceive, understand and learn the DM's own preference.

After the reference point  $\hat{\mathbf{z}}$  is specified, the following minimax problem is solved:

$$\left. \begin{array}{l} \text{minimize } \max_{1 \leq i \leq k} \{z_i(\mathbf{x}) - \hat{z}_i\} \\ \text{subject to } \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.62)$$

An optimal solution to (2.62) is a Pareto optimal solution closest to the reference point in the  $L_\infty$  norm; the  $L_\infty$  norm is also called the Tchebyshev norm or the Manhattan distance. Introducing an auxiliary variable  $v$ , (2.62) is equivalently expressed as

$$\left. \begin{array}{l} \text{minimize } v \\ \text{subject to } z_i(\mathbf{x}) - \hat{z}_i \leq v, i = 1, \dots, k \\ \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.63)$$



**Fig. 2.2** Reference point method.

Because the isoquant contour of the objective function  $\max_{1 \leq i \leq k} \{z_i(\mathbf{x}) - \hat{z}_i\}$  is orthogonal, there is a possibility that an optimal solution to (2.63) is not a Pareto optimal solution but a weak Pareto optimal solution due to a location of the reference point and the shape of the feasible region as seen in the left-hand side graph of Fig. 2.2. Let  $\rho$  be a given small positive number. By adding the augmented term  $\rho \sum_{i=1}^k (z_i(\mathbf{x}) - \hat{z}_i)$  to the objective function, the isoquant contour of the revised objective function has an obtuse angle as seen in the right-hand side graph of Fig. 2.2. From this fact, one finds that a Pareto optimal solution to the multiobjective linear programming problem (2.61), which is closest to the reference point  $\hat{\mathbf{z}}$ , can be obtained by solving the following revised problem:

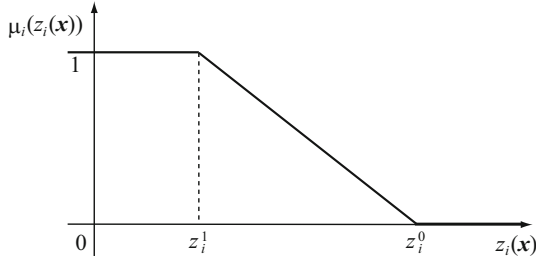
$$\left. \begin{array}{ll} \text{minimize} & v + \rho \sum_{i=1}^k (z_i(\mathbf{x}) - \hat{z}_i) \\ \text{subject to} & z_i(\mathbf{x}) - \hat{z}_i \leq v, \quad i = 1, \dots, k \\ & A\mathbf{x} \leq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.64)$$

### 2.3.3 Fuzzy multiobjective programming

To multiobjective linear programming problem (2.61), Zimmermann (1978) extends fuzzy programming given in the subsection 2.1.2 by introducing fuzzy goals for all the objective functions. Assuming that the DM has a fuzzy goal for each of the objective functions, the corresponding linear membership function is defined as

$$\mu_i(z_i(\mathbf{x})) = \begin{cases} 1 & \text{if } z_i(\mathbf{x}) \leq z_i^1 \\ \frac{z_i(\mathbf{x}) - z_i^0}{z_i^1 - z_i^0} & \text{if } z_i^1 < z_i(\mathbf{x}) \leq z_i^0 \\ 0 & \text{if } z_i(\mathbf{x}) > z_i^0, \end{cases} \quad (2.65)$$

where  $z_i^0$  and  $z_i^1$  denote the values of the  $i$ th objective function  $z_i(\mathbf{x})$  such that the degrees of the membership function are 0 and 1, respectively, and it is depicted in Fig. 2.3.



**Fig. 2.3** Linear membership function for the  $i$ th fuzzy goal.

Zimmermann (1978) also suggests a method for assessing these parameters  $z_i^0$  and  $z_i^1$  of the membership function. In his method, the parameter  $z_i^1$  is determined as

$$z_i^1 = \min_{\mathbf{x} \in X} z_i(\mathbf{x}), \quad (2.66)$$

and the parameter  $z_i^0$  is specified as

$$z_i^0 = \max_{j \neq i} z_i(\mathbf{x}^{jo}), \quad (2.67)$$

where  $\mathbf{x}^{jo}$  is a feasible solution minimizing  $z_j(\mathbf{x})$ , i.e.,  $\mathbf{x}^{jo} \in \arg \min_{\mathbf{x} \in X} z_j(\mathbf{x})$ . In this setting,  $z_i^1$  is set at the minimum of the  $i$ th objective function, and  $z_i^0$  is set at the maximum among the values of the  $i$ th objective function with respect to solutions minimizing  $z_j(\mathbf{x})$ ,  $j \neq i$ . By setting the parameters as described above, the linear membership functions (2.65) can be identified.

Following the principle of the fuzzy decision by Bellman and Zadeh (1970), the multiobjective linear programming problem (2.61) can be reformulated as the following maximin problem:

$$\begin{aligned} & \text{maximize } \min_{1 \leq i \leq k} \{ \mu_i(z_i(\mathbf{x})) \} \\ & \text{subject to } A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}. \end{aligned} \quad (2.68)$$

By introducing an auxiliary variable  $\lambda$ , the maximin problem (2.68) is equivalently rewritten as a standard linear programming problem

$$\left. \begin{array}{l} \text{maximize } \lambda \\ \text{subject to } \mu_i(z_i(\mathbf{x})) \geq \lambda, \quad i = 1, \dots, k \\ \quad \quad \quad \mathbf{Ax} \leq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.69)$$

In the formulation by Zimmermann, it is implicitly assumed that the DM feels that the fuzzy decision is appropriate for combining the fuzzy goals. However, in real-world decision situations, such an assumption is not always well-suited, and then an interactive method provides an alternative approach. Sakawa, Yano and Yumine (1987) propose an interactive method for a multiobjective linear programming problem with fuzzy goals. Because they incorporate not only “fuzzy min  $z_i(\mathbf{x})$ ” or “fuzzy max  $z_i(\mathbf{x})$ ” which is a fuzzy goal of the DM such as “ $z_i(\mathbf{x})$  should be substantially less than or equal to  $p_i$ , or greater than or equal to  $q_i$ ” but also “fuzzy equal  $z_i(\mathbf{x})$ ” which is a fuzzy goal such as “ $z_i(\mathbf{x})$  should be in the vicinity of  $r_i$ ,” and thus the concept of Pareto optimality cannot be applied, they introduce the following concept of M-Pareto optimal solutions defined in terms of membership functions instead of objective functions (Sakawa, 1993):

**Definition 2.7 (M-Pareto optimal solution).** A point  $\mathbf{x}^* \in X$  is said to be an *M-Pareto optimal solution* if and only if there does not exist another  $\mathbf{x} \in X$  such that  $\mu_i(z_i(\mathbf{x})) \geq \mu_i(z_i(\mathbf{x}^*))$  for all  $i \in \{1, \dots, k\}$  and  $\mu_j(z_j(\mathbf{x})) \neq \mu_j(z_j(\mathbf{x}^*))$  for at least one  $j \in \{1, \dots, k\}$ .

After identifying the membership functions  $\mu_i(z_i(\mathbf{x}))$ ,  $i = 1, \dots, k$  for the fuzzy goals of the objective functions  $z_i(\mathbf{x})$ ,  $i = 1, \dots, k$ , the DM is asked to specify the reference membership values which are the aspiration levels of achievement for the values of the membership functions. It follows that a vector of the reference membership values is a natural extension of the reference point in the reference point method by Wierzbicki (1980).

Let  $\hat{\boldsymbol{\mu}} = (\hat{\mu}_1, \dots, \hat{\mu}_k)^T$  denote the reference membership values for the membership functions  $\boldsymbol{\mu}(\mathbf{z}(\mathbf{x})) = (\mu_1(z_1(\mathbf{x})), \dots, \mu_k(z_k(\mathbf{x})))^T$ . Then, by solving the minimax problem

$$\left. \begin{array}{l} \text{minimize } \max_{1 \leq i \leq k} \{\hat{\mu}_i - \mu_i(z_i(\mathbf{x}))\} \\ \text{subject to } \mathbf{Ax} \leq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}, \end{array} \right\} \quad (2.70)$$

an M-Pareto optimal solution closest to the vector of the reference membership values in the  $L_\infty$  norm can be obtained.

The maximin problem (2.70) is equivalently expressed as

$$\left. \begin{array}{l} \text{minimize } v \\ \text{subject to } \hat{\mu}_i - \mu_i(z_i(\mathbf{x})) \leq v, \quad i = 1, \dots, k \\ \quad \quad \quad \mathbf{Ax} \leq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.71)$$

As we discussed linear programming with fuzzy parameters in the subsection 2.1.3, also in multiobjective situations, the formulation involving fuzzy parameters is still important. Sakawa and Yano (1990) extend the concept of Pareto optimality in order to deal with multiobjective linear programming problems with fuzzy parameters characterized by fuzzy numbers in the level set-based model. For given

parameters  $A$  and  $\mathbf{b}$  in the constraints, let  $X(\mathbf{a}, \mathbf{b})$  denote the corresponding feasible region, i.e.,  $X(\mathbf{a}, \mathbf{b}) \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$ .

**Definition 2.8 (M- $\alpha$ -Pareto optimal solution).** For a certain degree  $\alpha$  specified by the DM, a point  $\mathbf{x}^* \in X(\mathbf{a}^*, \mathbf{b}^*)$  is said to be an *M- $\alpha$ -Pareto optimal solution* if and only if there does not exist another  $\mathbf{x} \in X(\mathbf{a}, \mathbf{b})$ ,  $(\mathbf{c}, \mathbf{a}, \mathbf{b}) \in (C, A, B)_\alpha$  such that  $\mu_i(z_i(\mathbf{x})) \geq \mu_i(z_i(\mathbf{x}^*))$  for all  $i \in \{1, \dots, k\}$  and  $\mu_j(z_j(\mathbf{x})) \neq \mu_j(z_j(\mathbf{x}^*))$  for at least one  $j \in \{1, \dots, k\}$ , where the corresponding values of the parameters  $(\mathbf{c}^*, \mathbf{a}^*, \mathbf{b}^*) \in (C, A, B)_\alpha$  are called the  $\alpha$ -level optimal parameters.

To derive a satisficing solution to a multiobjective linear programming problem with fuzzy parameters for a certain degree  $\alpha$  specified by the DM, the minimax problem utilized in the interactive method can be formulated as

$$\left. \begin{array}{l} \text{minimize } v \\ \text{subject to } \hat{\mu}_i - \mu_i(z_i(\mathbf{x})) \leq v, \quad i = 1, \dots, k \\ A\mathbf{x} \leq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0} \\ (\mathbf{c}, \mathbf{a}, \mathbf{b}) \in (C, A, B)_\alpha. \end{array} \right\} \quad (2.72)$$

Let  $I_1$ ,  $I_2$ , and  $I_3$  be the index sets of the fuzzy goals for “fuzzy min  $z_i(\mathbf{x})$ ,” “fuzzy max  $z_i(\mathbf{x})$ ,” and “fuzzy equal  $z_i(\mathbf{x})$ ,” respectively. For notational convenience, denote the strictly monotone decreasing functions of  $\mu_i$ ,  $i \in I_1$  and the right-hand side functions of  $\mu_i$ ,  $i \in I_3$  by  $d_{iR}$ ,  $i \in I_1 \cup I_3$ , and the strictly monotone increasing functions of  $\mu_i$ ,  $i \in I_2$  and the left-hand side functions of  $\mu_i$ ,  $i \in I_3$  by  $d_{iL}$ ,  $i \in I_2 \cup I_3$ .

From the properties of the  $\alpha$ -level sets, (2.72) can be transformed into

$$\left. \begin{array}{l} \text{minimize } v \\ \text{subject to } c_{i\alpha}^L \mathbf{x} \leq d_{iR}^{-1}(\hat{\mu}_i - v), \quad i \in I_1 \cup I_3 \\ c_{i\alpha}^R \mathbf{x} \geq d_{iL}^{-1}(\hat{\mu}_i - v), \quad i \in I_2 \cup I_3 \\ a_{i1}^L x_1 + \dots + a_{in}^L x_n \leq b_i^R, \quad i = 1, \dots, m, \quad \mathbf{x} \geq \mathbf{0}, \end{array} \right\} \quad (2.73)$$

where  $d_{iR}^{-1}$  and  $d_{iL}^{-1}$  are pseudo-inverse functions defined by

$$d_{iR}^{-1}(h) = \sup\{y \mid d_{iR}(y) \geq h\}, \quad (2.74)$$

$$d_{iL}^{-1}(h) = \inf\{y \mid d_{iL}(y) \geq h\}. \quad (2.75)$$

It should be noted that (2.73) can be solved by using the combined use of the two-phase simplex method and the bisection method. For readers interested in this issue, refer to Sakawa (1993).

## 2.4 Two-level programming

### 2.4.1 Fuzzy programming for two-level programming

In the real world, we often encounter situations where there are two or more DMs in an organization with a hierarchical structure, and they make decisions in turn or at the same time so as to optimize their objective functions. In this section, we consider a case where there are two DMs; one of the DMs first makes a decision, and then after acknowledging the decision of the first DM, the other DM chooses a decision. Such a situation is formulated as a two-level programming problem.

A linear programming problem with two DMs is formulated as follows. For the sake of simplicity, we call the two DMs DM1 and DM2 in this subsection. Let  $\mathbf{x}_1$  and  $\mathbf{x}_2$  denote the column vectors of the decision variables of DM1 and DM2, respectively, and let  $z_1(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{c}_{11}\mathbf{x}_1 + \mathbf{c}_{12}\mathbf{x}_2$  and  $z_2(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{c}_{21}\mathbf{x}_1 + \mathbf{c}_{22}\mathbf{x}_2$  denote the objective functions of DM1 and DM2, respectively, where  $\mathbf{c}_{i1}$ ,  $i = 1, 2$  are  $n_1$  dimensional coefficient row vector, and  $\mathbf{c}_{i2}$ ,  $i = 1, 2$  are  $n_2$  dimensional coefficient row vector. Assume that  $A_1\mathbf{x}_1 + A_2\mathbf{x}_2 \leq \mathbf{b}$  are the common constraints of DM1 and DM2, where  $A_1$  is an  $m \times n_1$  coefficient matrix,  $A_2$  is an  $m \times n_2$  coefficient matrix,  $\mathbf{b}$  is an  $m$  dimensional constant column vector. Then, for a given  $\mathbf{x}_2$ , DM1 deals with the following linear programming problem:

$$\left. \begin{array}{ll} \underset{\mathbf{x}_1}{\text{minimize}} & z_1(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{c}_{11}\mathbf{x}_1 + \mathbf{c}_{12}\mathbf{x}_2 \\ \text{subject to} & A_1\mathbf{x}_1 \leq \mathbf{b} - A_2\mathbf{x}_2 \\ & \mathbf{x}_1 \geq \mathbf{0}. \end{array} \right\} \quad (2.76)$$

Similarly, for a given  $\mathbf{x}_1$ , the linear programming problem for DM2 is formulated as

$$\left. \begin{array}{ll} \underset{\mathbf{x}_2}{\text{minimize}} & z_2(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{c}_{21}\mathbf{x}_1 + \mathbf{c}_{22}\mathbf{x}_2 \\ \text{subject to} & A_2\mathbf{x}_2 \leq \mathbf{b} - A_1\mathbf{x}_1 \\ & \mathbf{x}_2 \geq \mathbf{0}. \end{array} \right\} \quad (2.77)$$

Combining the two problems (2.76) and (2.77) into one problem, we formulate the following linear programming problem with DM1 and DM2:

$$\left. \begin{array}{ll} \underset{\text{for DM1}}{\text{minimize}} & z_1(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{c}_{11}\mathbf{x}_1 + \mathbf{c}_{12}\mathbf{x}_2 \\ \underset{\text{for DM2}}{\text{minimize}} & z_2(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{c}_{21}\mathbf{x}_1 + \mathbf{c}_{22}\mathbf{x}_2 \\ \text{subject to} & A_1\mathbf{x}_1 + A_2\mathbf{x}_2 \leq \mathbf{b} \\ & \mathbf{x}_2 \geq \mathbf{0}, \mathbf{x}_1 \geq \mathbf{0}, \end{array} \right\} \quad (2.78)$$

where “minimize”<sub>for DM1</sub> and “minimize”<sub>for DM2</sub> mean that DM1 and DM2 are minimizers for their objective functions.

Assume that DM1 first makes a decision and then DM2 chooses a decision later, and to be better off each other, they can coordinate their decisions. Namely, we as-

sume that DM1 and DM2 make a binding agreement to select actions cooperatively, and then it is predicted that the selected actions should be in the set of Pareto optimal solutions. It should be noted here that Pareto optimality in two-level programming is defined for the two objective functions:  $z_1$  of DM1 and  $z_2$  of DM2.

For two-level linear programming problems with cooperative DMs, fuzzy programming approaches have been developed (Lai, 1996; Shih, Lai and Lee, 1996; Sakawa, Nishizaki and Uemura, 1998; Sakawa and Nishizaki, 2009). In fuzzy programming for two-level linear programming by Sakawa et al. (Sakawa, Nishizaki and Uemura, 1998; Sakawa and Nishizaki, 2009), for each of the objective functions  $z_i(\mathbf{x})$ ,  $i = 1, 2$  of (2.78), it is assumed that the DMs have fuzzy goals such as “the objective function  $z_i(\mathbf{x})$  should be substantially less than or equal to some specific value  $p_i$ .” Although the membership function does not always need to be linear, for the sake of simplicity, assume that a membership function  $\mu_i(z_i)$  which characterizes the fuzzy goal of each DM is linear and it is specified as

$$\mu_i(z_i(\mathbf{x})) = \begin{cases} 1 & \text{if } z_i(\mathbf{x}) \leq z_i^1 \\ \frac{z_i(\mathbf{x}) - z_i^0}{z_i^1 - z_i^0} & \text{if } z_i^1 < z_i(\mathbf{x}) \leq z_i^0 \\ 0 & \text{if } z_i(\mathbf{x}) > z_i^0. \end{cases} \quad (2.79)$$

By identifying the membership functions  $\mu_1(z_1(\mathbf{x}))$  and  $\mu_2(z_2(\mathbf{x}))$  for the objective functions  $z_1(\mathbf{x})$  and  $z_2(\mathbf{x})$ , the original two-level linear programming problem (2.78) can be interpreted as the membership function maximization problem defined by

$$\left. \begin{array}{l} \text{minimize}_{\text{for DM1}} \mu_1(z_1(\mathbf{x})) \\ \text{minimize}_{\text{for DM2}} \mu_2(z_2(\mathbf{x})) \\ \text{subject to } \mathbf{Ax} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.80)$$

In (2.80),  $\mathbf{x} \in \mathbb{R}^n$  is an  $n$  dimensional decision variable vector, and it is divided into two vectors  $\mathbf{x}_1$  and  $\mathbf{x}_2$  which are  $n_1$  and  $n_2$  dimensional decision variable vectors of DM1 and DM2, respectively, i.e.,  $n = n_1 + n_2$ . Because the two DMs make decisions cooperatively, the decision variable vector is represented simply by  $\mathbf{x}$  without partition.

To derive an overall satisfactory solution to the membership function maximization problem (2.80), we first find the maximizing decision of the fuzzy decision proposed by Bellman and Zadeh (1970). Namely, the following problem is solved for obtaining a solution which maximizes the smaller degree of satisfaction between those of the two DMs:

$$\left. \begin{array}{l} \text{maximize } \min\{\mu_1(z_1(\mathbf{x})), \mu_2(z_2(\mathbf{x}))\} \\ \text{subject to } \mathbf{Ax} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.81)$$

By introducing an auxiliary variable  $\lambda$ , this problem can be transformed into the following equivalent problem:

$$\left. \begin{array}{l} \text{maximize } \lambda \\ \text{subject to } \mu_1(z_1(\mathbf{x})) \geq \lambda \\ \mu_2(z_2(\mathbf{x})) \geq \lambda \\ A\mathbf{x} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0}. \end{array} \right\} \quad (2.82)$$

Solving (2.82), we can obtain a solution which maximizes the smaller satisfactory degree between those of both DMs. It should be noted that if the membership functions  $\mu_i(z_i(\mathbf{x}))$ ,  $i = 1, 2$  are linear membership functions such as (2.79), (2.82) becomes a linear programming problem. Let  $\mathbf{x}^*$  denote an optimal solution to problem (2.82). Then, we define the satisfactory degree of both DMs under the constraints as

$$\lambda^* = \min\{\mu_1(z_1(\mathbf{x}^*)), \mu_2(z_2(\mathbf{x}^*))\}. \quad (2.83)$$

If DM1 is satisfied with the optimal solution  $\mathbf{x}^*$ , it follows that the optimal solution  $\mathbf{x}^*$  becomes a satisfactory solution; however, DM1 is not always satisfied with the solution  $\mathbf{x}^*$ . If DM1 is not satisfied with the solution  $\mathbf{x}^*$ , it is quite natural to assume that DM1 specifies the minimal satisfactory level  $\delta \in [0, 1]$  for the membership function  $\mu_1(z_1(\mathbf{x}))$  subjectively. Then, the following problem is formulated:

$$\left. \begin{array}{l} \text{maximize } \mu_2(z_2(\mathbf{x})) \\ \text{subject to } \mu_1(z_1(\mathbf{x})) \geq \delta \\ A\mathbf{x} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0}, \end{array} \right\} \quad (2.84)$$

where DM2's membership function is maximized under the condition that DM1's membership function  $\mu_1(z_1(\mathbf{x}))$  is larger than or equal to the minimal satisfactory level  $\delta$  specified by DM1. It should be also noted that if the membership functions  $\mu_i(z_i(\mathbf{x}))$ ,  $i = 1, 2$  are linear membership functions such as (2.79), then (2.84) also becomes a linear programming problem.

If there exists an optimal solution to (2.84), it follows that DM1 obtains a satisfactory solution having a satisfactory degree larger than or equal to the minimal satisfactory level specified by DM1's self. However, the larger the minimal satisfactory level  $\delta$  is assessed, the smaller the DM2's satisfactory degree becomes when the objective functions of DM1 and DM2 conflict with each other. Consequently, a relative difference between the satisfactory degrees of DM1 and DM2 becomes larger, and it follows that the overall satisfactory balance between both DMs is not appropriate.

In order to take account of the overall satisfactory balance between both DMs, DM1 needs to compromise with DM2 on DM1's own minimal satisfactory level. To do so, the following ratio of the satisfactory degree of DM2 to that of DM1 is helpful:

$$\Delta = \frac{\mu_2(z_2(\mathbf{x}))}{\mu_1(z_1(\mathbf{x}))}, \quad (2.85)$$

which is originally introduced by Lai (1996).

DM1 is guaranteed to have a satisfactory degree larger than or equal to the minimal satisfactory level for the fuzzy goal because the corresponding constraint  $\mu_1(z_1(\mathbf{x})) \geq \delta$  is involved in (2.84). To take into account the overall satisfactory balance between both DMs, DM1 specifies the lower bound  $\Delta_{\min}$  and the upper bound  $\Delta_{\max}$  of the ratio, and then it is verified whether the ratio  $\Delta$  is in the interval  $[\Delta_{\min}, \Delta_{\max}]$  or not. The condition that the overall satisfactory balance is appropriate is represented by

$$\Delta \in [\Delta_{\min}, \Delta_{\max}]. \quad (2.86)$$

At the iteration  $l$ , let  $\mu_1(z_1^l)$ ,  $\mu_2(z_2^l)$ ,  $\lambda^l$  and  $\Delta^l = \mu_2(z_2^l)/\mu_1(z_1^l)$  denote DM1's and DM2's satisfactory degrees, a satisfactory degree of both DMs, and the ratio of satisfactory degrees of the two DMs, respectively. Let the solution be  $\mathbf{x}^l$  at the iteration  $l$ . The interactive process terminates if the following two conditions are satisfied and if DM1 concludes the solution as an overall satisfactory solution.

### Termination conditions of the interactive process

Condition 1: DM1's satisfactory degree is larger than or equal to the minimal satisfactory level  $\delta$  specified by DM1's self, i.e.,  $\mu_1(z_1^l) \geq \delta$ .

Condition 2: The ratio  $\Delta^l$  of satisfactory degrees lies in the closed interval between the lower and the upper bounds specified by DM1, i.e.,  $\Delta^l \in [\Delta_{\min}, \Delta_{\max}]$ .

Condition 1 ensures the minimal satisfaction to DM1 in the sense of the attainment of the fuzzy goal, and condition 2 is provided in order to keep overall satisfactory balance between both DMs. If these two conditions are not satisfied simultaneously, DM1 needs to update the minimal satisfactory level  $\delta$ . The updating procedures are summarized as follows:

### Procedure for updating the minimal satisfactory level $\delta$

Case 1: If condition 1 is not satisfied, then DM1 decreases the minimal satisfactory level  $\delta$ .

Case 2: If the ratio  $\Delta^l$  exceeds its upper bound, then DM1 increases the minimal satisfactory level  $\delta$ . Conversely, if the ratio  $\Delta^l$  is below its lower bound, then DM1 decreases the minimal satisfactory level  $\delta$ .

Case 3: Although conditions 1 and 2 are satisfied, if DM1 is not satisfied with the obtained solution and judges that it is desirable to increase the satisfactory degree of DM1 at the expense of the satisfactory degree of DM2, then DM1 increases the minimal satisfactory level  $\delta$ . Conversely, if DM1 judges that it is desirable to increase the satisfactory degree of DM2 at the expense of the satisfactory degree of DM1, then DM1 decreases the minimal satisfactory level  $\delta$ .

In particular, if condition 1 is not satisfied, it follows that there does not exist any feasible solution for (2.84), and therefore DM1 has to moderate the minimal satisfactory level.

We are now ready to give a procedure of interactive fuzzy programming for deriving an overall satisfactory solution to (2.78), which is summarized in the following.

### Algorithm of interactive fuzzy programming

- Step 1: Ask DM1 to identify the membership function  $\mu_1(z_1)$  of the fuzzy goal for the objective function  $z_1(\mathbf{x})$ . Similarly, ask DM2 to identify the membership function  $\mu_2(z_2)$  of the fuzzy goal for the objective function  $z_2(\mathbf{x})$ .
- Step 2: Set  $l := 1$  and solve (2.82), in which a smaller satisfactory degree between those of DM1 and DM2 is maximized. If DM1 is satisfied with the obtained optimal solution, the solution becomes a satisfactory solution. Otherwise, ask DM1 to specify the minimal satisfactory level  $\delta$  together with the lower and the upper bounds  $[\Delta_{\min}, \Delta_{\max}]$  of the ratio of satisfactory degrees  $\Delta^l$  with the satisfactory degree  $\lambda^*$  of both DMs and the related information about the solution in mind.
- Step 3: Set  $l := l + 1$ . Solve (2.84), in which the satisfactory degree of DM2 is maximized under the condition that the satisfactory degree of DM1 is larger than or equal to the minimal satisfactory level  $\delta$ , and then an optimal solution  $\mathbf{x}^l$  to (2.84) is proposed to DM1 together with  $\lambda^l, \mu_1(z_1^l), \mu_2(z_2^l)$  and  $\Delta^l$ .
- Step 4: If the solution  $\mathbf{x}^l$  satisfies the termination conditions and DM1 accepts it, then the procedure stops, and the solution  $\mathbf{x}^l$  is determined to be a satisfactory solution.
- Step 5: Ask DM1 to revise the minimal satisfactory level  $\delta$  in accordance with the procedure of updating minimal satisfactory level. Return to step 3.

### 2.4.2 Stackelberg solution to two-level programming problem

In the previous subsection, we considered a two-level linear programming problem in which two DMs can coordinate their decisions. However, for example, if the two DMs are corporate managers in different companies and the companies have business with each other, it is natural to suppose that there should be conflict interests between them. If the two DMs have conflicting interests, they cannot always cooperate in their decisions. Then, it is difficult to employ a cooperative solution method such as the fuzzy programming approach in the previous subsection.

For noncooperative decision making in a two-level programming problem, a DM who first makes a decision is called the leader and the other DM is called the follower conventionally. The leader first specifies a decision and then the follower determines a decision so as to optimize the objective function of the follower with full knowledge of the decision of the leader. Anticipating this, the leader also makes a decision so as to optimize the objective function of self. The solution defined as the above mentioned procedure is a Stackelberg equilibrium solution, and we call it a Stackelberg solution shortly.

A two-level linear programming problem for obtaining the Stackelberg solution is formulated as

$$\left. \begin{array}{l}
 \underset{x}{\text{minimize}} \quad z_1(\mathbf{x}, \mathbf{y}) = \mathbf{c}_1 \mathbf{x} + \mathbf{d}_1 \mathbf{y} \\
 \text{where } \mathbf{y} \text{ solves} \\
 \underset{y}{\text{minimize}} \quad z_2(\mathbf{x}, \mathbf{y}) = \mathbf{c}_2 \mathbf{x} + \mathbf{d}_2 \mathbf{y} \\
 \text{subject to} \quad \mathbf{A} \mathbf{x} + \mathbf{B} \mathbf{y} \leq \mathbf{b} \\
 \mathbf{x} \geq \mathbf{0}, \mathbf{y} \geq \mathbf{0},
 \end{array} \right\} \quad (2.87)$$

where  $\mathbf{c}_i$ ,  $i = 1, 2$  are  $n_1$  dimensional coefficient row vector,  $\mathbf{d}_i$ ,  $i = 1, 2$  are  $n_2$  dimensional coefficient row vector,  $\mathbf{A}$  is an  $m \times n_1$  coefficient matrix,  $\mathbf{B}$  is an  $m \times n_2$  coefficient matrix,  $\mathbf{b}$  is an  $m$ -dimensional constant column vector. In the two-level linear programming problem (2.87),  $z_1(\mathbf{x}, \mathbf{y})$  and  $z_2(\mathbf{x}, \mathbf{y})$  represent the objective functions of the leader and the follower, respectively, and  $\mathbf{x}$  and  $\mathbf{y}$  represent the decision variables of the leader and the follower, respectively.

Each DM knows the objective function of the opponent as well as the objective function of self and the constraints. The leader first makes a decision, and then the follower makes a decision so as to minimize the objective function with full knowledge of the decision of the leader. Namely, after the leader chooses  $\mathbf{x}$ , the follower solves the following linear programming problem:

$$\left. \begin{array}{l}
 \underset{y}{\text{minimize}} \quad z_2(\mathbf{x}, \mathbf{y}) = \mathbf{d}_2 \mathbf{y} + \mathbf{c}_2 \mathbf{x} \\
 \text{subject to} \quad \mathbf{B} \mathbf{y} \leq \mathbf{b} - \mathbf{A} \mathbf{x} \\
 \mathbf{y} \geq \mathbf{0},
 \end{array} \right\} \quad (2.88)$$

and chooses an optimal solution  $\mathbf{y}(\mathbf{x})$  to (2.88) as a rational response. Assuming that the follower chooses the rational response, the leader also makes a decision such that the objective function  $z_1(\mathbf{x}, \mathbf{y}(\mathbf{x}))$  is minimized. Then, the solution defined as the above mentioned procedure is a Stackelberg solution.

Computational methods for obtaining Stackelberg solutions to two-level linear programming problems are classified roughly into three categories: the vertex enumeration approach (Bialas and Karwan, 1984), the Kuhn-Tucker approach (Bard and Falk, 1982; Bard and Moore, 1990; Bialas and Karwan, 1984; Hansen, Jaumard and Savard, 1992), and the penalty function approach (White and Anandalingam, 1993). The vertex enumeration approach takes advantage of the property that there exists a Stackelberg solution in a set of extreme points of the feasible region. In the Kuhn-Tucker approach, the leader's problem with constraints involving the optimality conditions of the follower's problem is solved. In the penalty function approach, a penalty term is appended to the objective function of the leader so as to satisfy the optimality of the follower's problem. It is well-known that a two-level linear programming problem is an NP-hard problem (Shimizu, Ishizuka and Bard, 1997). In the following, we outline a couple of conventional computational methods for obtaining Stackelberg solutions.

The  $k$ th best method proposed by Bialas and Karwan (1984) can be thought of as the vertex enumeration approach, and it is based on a very simple idea. The solution search procedure of the method starts from a point which is an optimal solution to the problem of the leader and checks whether it is also an optimal solution to

the problem of the follower or not. If the first point is not the Stackelberg solution, the procedure continues to examine the second best solution to the problem of the leader, and so forth.

At the beginning, the following linear programming problem is solved:

$$\left. \begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & z_1(\mathbf{x}, \mathbf{y}) = \mathbf{c}_1 \mathbf{x} + \mathbf{d}_1 \mathbf{y} \\ \text{subject to} & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \mathbf{y} \geq \mathbf{0}. \end{array} \right\} \quad (2.89)$$

Assuming that the feasible region of (2.89) is not empty and there are  $N$  extreme points, i.e.,  $N$  basic solutions of (2.89). Let  $(\hat{\mathbf{x}}^{[1]}, \hat{\mathbf{y}}^{[1]})$  denote an optimal solution to (2.89), and  $(\hat{\mathbf{x}}^{[2]}, \hat{\mathbf{y}}^{[2]})$ ,  $\dots$ ,  $(\hat{\mathbf{x}}^{[N]}, \hat{\mathbf{y}}^{[N]})$  be the rest of  $N - 1$  basic feasible solutions such that  $z_1(\hat{\mathbf{x}}^{[j]}, \hat{\mathbf{y}}^{[j]}) \leq z_1(\hat{\mathbf{x}}^{[j+1]}, \hat{\mathbf{y}}^{[j+1]})$ ,  $j = 1, \dots, N - 1$ . It is verified if the solution  $(\hat{\mathbf{x}}^{[j]}, \hat{\mathbf{y}}^{[j]})$  is optimal to (2.88) of the follower from  $j = 1$  to  $j = N$  in turn. Then, the first solution found to be optimal to (2.88) is the Stackelberg solution.

In the Kuhn-Tucker approach, the leader's problem with constraints involving the optimality conditions of the follower's problem (2.88) is solved. The Kuhn-Tucker conditions for (2.88) are shown as follows:

$$\left. \begin{array}{l} \mathbf{u}\mathbf{B} - \mathbf{v} = -\mathbf{d}_2 \\ \mathbf{u}(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{b}) - \mathbf{v}\mathbf{y} = 0 \\ \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \leq \mathbf{b} \\ \mathbf{y} \geq \mathbf{0}, \mathbf{u}^T \geq \mathbf{0}, \mathbf{v}^T \geq \mathbf{0}, \end{array} \right\} \quad (2.90)$$

where  $\mathbf{u}$  is an  $m$  dimensional row vector and  $\mathbf{v}$  is an  $n_2$  dimensional row vector.

Then, the follower's problem (2.88) for a two-level linear programming problem can be replaced by the above conditions (2.90), and (2.87) is rewritten as the following equivalent single-level mathematical programming problem:

$$\left. \begin{array}{ll} \text{minimize} & z_1(\mathbf{x}, \mathbf{y}) = \mathbf{c}_1 \mathbf{x} + \mathbf{d}_1 \mathbf{y} \\ \text{subject to} & \mathbf{u}\mathbf{B} - \mathbf{v} = -\mathbf{d}_2 \\ & \mathbf{u}(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{b}) - \mathbf{v}\mathbf{y} = 0 \\ & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \mathbf{y} \geq \mathbf{0}, \mathbf{u}^T \geq \mathbf{0}, \mathbf{v}^T \geq \mathbf{0}. \end{array} \right\} \quad (2.91)$$

From the equality constraint  $\mathbf{u}\mathbf{B} - \mathbf{v} = -\mathbf{d}_2$  of (2.91),  $\mathbf{v}$  is eliminated and the equality constraint  $\mathbf{u}(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{b}) - \mathbf{v}\mathbf{y} = 0$  is transformed into

$$\mathbf{u}(\mathbf{b} - \mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{y}) + (\mathbf{u}\mathbf{B} + \mathbf{d}_2)\mathbf{y} = 0. \quad (2.92)$$

Moreover, the complementarity condition (2.92) implies that  $\mathbf{b} - \mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{y} \geq \mathbf{0}$ ,  $\mathbf{u}^T \geq \mathbf{0}$ ,  $(\mathbf{u}\mathbf{B} + \mathbf{d}_2)^T \geq \mathbf{0}$ ,  $\mathbf{y} \geq \mathbf{0}$ . Let  $A_i$  and  $B_i$  be the  $i$ th row vector of the matrix  $A$  and the matrix  $B$ , respectively, and let  $B^j$  and  $d_{2j}$  be the  $j$ th column vector of the matrix  $B$  and the  $j$ th element of the vector  $\mathbf{d}_2$ , respectively. Then, the condition of either  $u_i = 0$  or  $b_i - A_i\mathbf{x} - B_i\mathbf{y} = 0$  for  $i = 1, \dots, m$  and the condition of either  $\mathbf{u}\mathbf{B}^j + d_{2j} = 0$  or  $y_j = 0$  for  $j = 1, \dots, n_2$  must be satisfied simultaneously. By introducing zero-one

vectors  $\mathbf{w}_1 = (w_{11}, \dots, w_{1m})$  and  $\mathbf{w}_2 = (w_{21}, \dots, w_{2n_2})$ , the equality constraint (2.92) can be expressed as follows (Fortuny-Amat and McCarl, 1981):

$$\left. \begin{aligned} \mathbf{u} &\leq M\mathbf{w}_1 \\ \mathbf{b} - A\mathbf{x} - B\mathbf{y} &\leq M(\mathbf{e} - \mathbf{w}_1^T) \\ \mathbf{u}B + \mathbf{d}_2 &\leq M\mathbf{w}_2 \\ \mathbf{y} &\leq M(\mathbf{e} - \mathbf{w}_2^T), \end{aligned} \right\} \quad (2.93)$$

where  $\mathbf{e}$  is an  $m$  dimensional vector of ones, and  $M$  is a large positive constant.

Therefore, the mathematical programming problem (2.91) is equivalent to the following mixed zero-one programming problem, and it can be solved by a zero-one mixed integer solver:

$$\left. \begin{aligned} \text{minimize } z_1(\mathbf{x}, \mathbf{y}) &= \mathbf{c}_1\mathbf{x} + \mathbf{d}_1\mathbf{y} \\ \text{subject to } \mathbf{0} &\leq \mathbf{u}^T \leq M\mathbf{w}_1^T \\ \mathbf{0} &\leq \mathbf{b} - A\mathbf{x} - B\mathbf{y} \leq M(\mathbf{e} - \mathbf{w}_1^T) \\ \mathbf{0} &\leq (\mathbf{u}B + \mathbf{d}_2)^T \leq M\mathbf{w}_2^T \\ \mathbf{0} &\leq \mathbf{y} \leq M(\mathbf{e} - \mathbf{w}_2^T) \\ \mathbf{x} &\geq \mathbf{0}. \end{aligned} \right\} \quad (2.94)$$

In the penalty function approach, the duality gap of the follower's problem (2.88) is appended to the objective function of the leader. The dual problem to (2.88) ignoring the constant term  $\mathbf{c}_2\mathbf{x}$  is written as

$$\left. \begin{aligned} \text{minimize } &\mathbf{u}(A\mathbf{x} - \mathbf{b}) \\ \text{subject to } &-\mathbf{u}B \leq \mathbf{d}_2 \\ &\mathbf{u}^T \geq \mathbf{0}, \end{aligned} \right\} \quad (2.95)$$

where  $\mathbf{u}$  is an  $m$ -dimensional row vector. Because the duality gap  $\mathbf{d}_2\mathbf{y} - \mathbf{u}(A\mathbf{x} - \mathbf{b})$  is zero if  $\mathbf{y}$  is a rational response of the follower with respect to a choice  $\mathbf{x}$  of the leader, the following mathematical programming problem is formulated:

$$\left. \begin{aligned} \text{minimize } &\mathbf{c}_1\mathbf{x} + \mathbf{d}_1\mathbf{y} + K\mathbf{u}(A\mathbf{x} - \mathbf{b}) \\ \text{subject to } &A\mathbf{x} + B\mathbf{y} \leq \mathbf{b} \\ &-\mathbf{u}B \leq \mathbf{d}_2 \\ &\mathbf{x} \geq \mathbf{0}, \mathbf{y} \geq \mathbf{0}, \mathbf{u}^T \geq \mathbf{0}, \end{aligned} \right\} \quad (2.96)$$

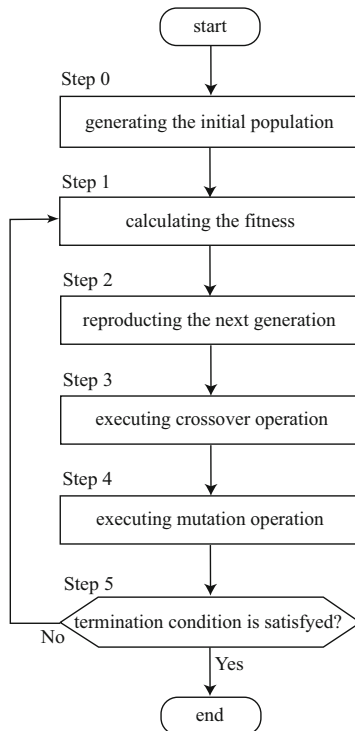
where  $K$  is a constant value. By repeatedly solving (2.96) for updated values of  $K$  and  $\mathbf{u}$ , (2.96) yields an optimal solution to (2.87), i.e., the Stackelberg solution.

## 2.5 Genetic algorithms

### 2.5.1 Fundamental elements in genetic algorithms

It is hard to obtain exact optimal solutions of difficult classes of optimization problems such as combinatorial problems and nonconvex nonlinear problems, and thus it is quite natural for DMs to require approximate optimal solutions instead. To meet this demand, recently several meta-heuristics have been developed and their effectiveness is demonstrated. Among them, genetic algorithms are known to be one of the most practical and proven methods, and in this section we present the basic concepts of the genetic algorithms.

A computational framework of genetic algorithms initiated by Holland (1975) has been attracted attention of many researchers with applicability in optimization, as well as in search and learning. Furthermore, publications of books by Goldberg (1989) and Michalewicz (1996) bring heightened and increasing interests in applications of genetic algorithms to complex function optimization.



**Fig. 2.4** Flowchart of genetic algorithms.

The fundamental procedure of genetic algorithms is shown as a flowchart in Fig. 2.4, and it is summarized as follows:

- Step 0: Initialization. Generate a given number of individuals randomly to form the initial population.
- Step 1: Evaluation. Calculate the fitness value of each individual in the population.
- Step 2: Reproduction. According to the fitness values and a reproduction rule specified in advance, select individuals from the current population to form the next population.
- Step 3: Crossover. Select two individuals randomly from the population, and exchange some part of the string of one individual for the corresponding part of the other individual with a given probability for crossover.
- Step 4: Mutation. Alter one or more genes in the string of an individual with a given probability of mutation.
- Step 5: Termination. Stop the procedure if the condition of termination is satisfied, and an individual with the maximum fitness value is determined as an approximate optimal solution. Otherwise, return to step 1.

2.5.1.1 Representation of individuals

When genetic algorithms are applied to optimization problems, a vector of decision variables corresponds to an individual in the population, which is represented by a string as in Fig. 2.5.

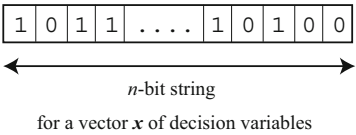


Fig. 2.5 Individual represented by a string.

As seen in Fig. 2.5, each element of the string is either 1 or 0 usually, but real numbers, integers, alphabets, or some other symbols can also be used to represent individuals.

Let  $s$  and  $x$  denote an individual represented by a string and a vector of the decision variables, respectively. The string  $s$  which means a chromosome in the context of biology is called the genotype of an individual, and the decision variables  $x$  is called the phenotype. The mapping from phenotypes to genotypes is called coding, and the reverse mapping is called decoding.

### 2.5.1.2 Fitness function and scaling

In an optimization problem where the objective function is minimized or maximized, a solution with the smallest objective function value or the largest objective function value is searched. When a genetic algorithm is applied to solving an optimization problem, a solution to the optimization problem is associated with an individual in the genetic algorithm, and the objective function value of the solution corresponds to the fitness of the individual. Thus, an individual with a larger fitness value has a higher probability of surviving in the next generation.

Let  $z(\mathbf{x})$  denote an objective function to be minimized in an optimization problem. The corresponding fitness function in genetic algorithms is commonly defined as (Goldberg, 1989)

$$f(s_i) = \begin{cases} C_{\max} - z(\mathbf{x}) & \text{if } z(\mathbf{x}) < C_{\max} \\ 0 & \text{otherwise,} \end{cases} \quad (2.97)$$

where  $s_i$  denotes the  $i$ th individual in the population, and  $C_{\max}$  is a given constant. For example, the value of  $C_{\max}$  is determined as the largest objective function value  $z(\mathbf{x})$  observed thus far, the largest value  $z(\mathbf{x})$  in the current population, or the largest value  $z(\mathbf{x})$  in the last  $t$  generations. Similarly, in maximization problems, to prevent the fitness value from being negative, the constant  $C_{\min}$  is introduced, and the following fitness function is often used:

$$f(s_i) = \begin{cases} z(\mathbf{x}) + C_{\min} & \text{if } z(\mathbf{x}) + C_{\min} > 0 \\ 0 & \text{otherwise.} \end{cases} \quad (2.98)$$

The absolute value of the smallest  $z(\mathbf{x})$  in the current population or in the last  $t$  generations is often used for the value of  $C_{\min}$ .

To properly distribute fitness values in the population, fitness scaling is employed. The linear scaling, which is a simple and useful procedure, is represented by

$$f'(s_i) = af(s_i) + b, \quad (2.99)$$

where  $f$  and  $f'$  are the raw fitness value and the scaled fitness value, respectively;  $a$  and  $b$  are coefficients. To perform the operation of reproduction appropriately, the coefficients  $a$  and  $b$  may be chosen in such a way that the average scaled fitness value  $f'_{\text{ave}}$  is equal to the average raw fitness value  $f_{\text{ave}}$ , and the maximum scaled fitness value is determined as  $f'_{\text{max}} = C_{\text{mult}}f_{\text{ave}}$ , where  $C_{\text{mult}}$  is a given constant.

### 2.5.1.3 Genetic operators

The three genetic operators, reproduction, crossover, and mutation, are outlined below. Individuals are copied into the next generation according to their fitness values by a reproduction operator. The roulette wheel selection is one of the most popular

reproduction operators, and in this method, each individual in the current population has a roulette wheel slot sized in proportion to its fitness value. Let  $pop\_size$  be the number of individuals in the population. The percentage of the roulette wheel given to an individual  $s_i$  is  $100f(s_i)/\sum_{l=1}^{pop\_size} f(s_l)\%$ . Namely, the individual  $s_i$  is reproduced with the probability  $p(s_i) = f(s_i)/\sum_{l=1}^{pop\_size} f(s_l)$  each spin of the roulette wheel. An example of the roulette wheel is given in Fig. 2.6; the numbers in the wheel are fitness values of individuals, and the decimal numbers outside of the wheel are the corresponding probabilities.

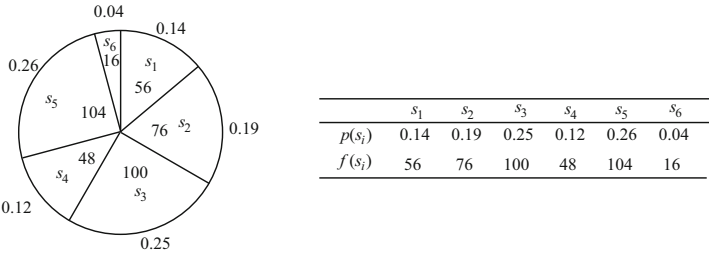


Fig. 2.6 Biased roulette wheel.

Crossover creates offsprings into the next population by combining the genetic material of two parents. The variation caused by the crossover process may bring offsprings better fitness values, and thus it is thought that crossover plays an important role in genetic algorithms. Although there are many different types of crossover, we provide a simple example here: a single-point crossover operator is the most simple operator of crossover. In this operation, two parent strings  $s_1$  and  $s_2$  are randomly chosen from a mating pool in which newly reproduced individuals are entered temporarily, and then one crossover point in the strings is chosen at random. Two offsprings are made by exchanging the substrings which are parts of the left-hand side of the parent strings  $s_1$  and  $s_2$  from the crossover point. The crossover operation is illustrated in Fig. 2.7.

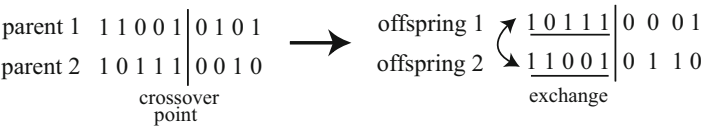


Fig. 2.7 Crossover operation.

With a small probability, an operation of mutation provides the string of an individual with a randomly tiny alteration, and it is recognized that mutation serves as local search. In the representation of the 0-1 bit strings, mutation means changing a

1 to a 0 and vice versa. A simple version of mutation operator is illustrated in Fig. 2.8.



**Fig. 2.8** Mutation operation.

### 2.5.2 Genetic algorithm for integer programming

So far we have discussed fundamental elements in genetic algorithms. In the remaining chapters of this book, we will deal with mathematical programming problems not only with continuous decision variables but also with discrete decision variables, and for solving mathematical programming problems with discrete decision variables, i.e., integer programming problems, we give a computational method based on the framework of genetic algorithms, which is called GADSLPRRSU by Sakawa (2001).

GADSLPRRSU is an abbreviation for a genetic algorithm with double strings based on linear programming relaxation and reference solution updating. This method includes three key ideas: double strings (DS), linear programming relaxation (LPR), and reference solution updating (RSU). These ideas were introduced one by one in the development of this method.

Sakawa, Kato, Sunada and Shibano (1997) first attempt to solve a multidimensional 0-1 knapsack problem by applying a genetic algorithm. For a given set of  $n$  kinds of items, the aim of the multidimensional knapsack problem is to select a subset of the items so as to maximize the total values of the selected items under the multiple constraints such as capacities, budgets, and so forth. Let  $x_j$  be a decision variable which indicates whether the  $j$ th item is selected or not. Namely, the decision variable  $x_j$  is 1 if the  $j$ th item is selected, and otherwise it is 0. Then, the general form of multidimensional knapsack problems is given as

$$\left. \begin{array}{ll} \text{minimize} & \mathbf{c}\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & x_j \in \{0, 1\}, \quad j = 1, \dots, n, \end{array} \right\} \quad (2.100)$$

where  $\mathbf{c}$ ,  $\mathbf{A}$  and  $\mathbf{b}$  are an  $n$  dimensional row vector of positive coefficients in the objective function, an  $m \times n$  matrix of positive coefficients in the left-hand side constraints and an  $m$  dimensional column vector of positive constants in the right-hand side constraints.

If a simple genetic algorithm described above is applied to a multidimensional 0-1 knapsack problem, an individual in the population is not always decoded into a feasible solution to the multidimensional 0-1 knapsack problem. Sakawa, Kato,

Sunada and Shibano (1997) resolve this difficulty by introducing the double strings representation of individuals and the corresponding decoding algorithm. Furthermore, to apply this method to a multidimensional integer knapsack problem in which the domain of a decision variable is extended to a set of nonnegative integers, they utilize information of optimal solutions to linear programming relaxation problems (Sakawa *et al.*, 2000). By using the fact that the zero solution,  $\mathbf{x} = \mathbf{0}$ , is feasible to a multidimensional 0-1 or integer knapsack problem, these methods decode any individual into a feasible solution to the problem. However, in an integer programming problem which includes a multidimensional 0-1 or integer knapsack problem as a special case, the zero solution is not always feasible, and therefore these methods cannot directly apply to integer programming problems. An integer programming problem is formally given as

$$\left. \begin{array}{ll} \text{minimize} & \mathbf{c}\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & x_j \in \{0, 1, \dots, v_j\}, j = 1, \dots, n, \end{array} \right\} \quad (2.101)$$

where  $\mathbf{A} = [\mathbf{p}_1, \dots, \mathbf{p}_n]$  is an  $m \times n$  coefficient matrix,  $\mathbf{b} = (b_1, \dots, b_m)^T$  is an  $m$  dimensional constant column vector and  $\mathbf{c} = (c_1, \dots, c_n)$  is an  $n$  dimensional coefficient row vector. In contrast to knapsack problems such as (2.100), any coefficient of (2.101) is not always positive.

To overcome this problem, they propose a revised version (GADSLPRRSU) of their methods with a decoding algorithm with reference solutions which are found in advance and are updated if necessary (Sakawa, 2001). GADSLPRRSU for solving integer programming problems (2.101) is summarized as follows. As we pointed out, GADSLPRRSU employs the double string representation of individuals depicted in Fig. 2.9.

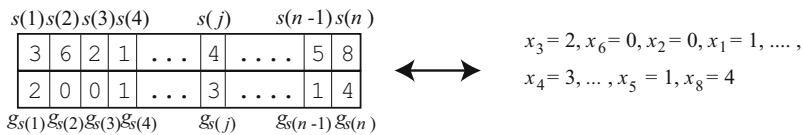


Fig. 2.9 Double string representation.

In the figure, each of  $s(j)$ ,  $j = 1, \dots, n$  is the index of an element in a solution vector and each of  $g_{s(j)} \in \{0, 1, \dots, v_{s(j)}\}$ ,  $j = 1, \dots, n$  is the value of the element, respectively. For example, a pair  $(s(j), g_{s(j)})$  means that the value of the  $s(j)$ th element  $x_{s(j)}$  is  $g_{s(j)}$ , i.e.,  $x_{s(j)} = g_{s(j)}$ .

A decoding algorithm for the double string representation with reference points generates feasible solutions from any individuals based on a certain reference solution  $\mathbf{x}^*$  which is used as the origin of decoding. In the following algorithm,  $\mathbf{b}^+$  denotes a column vector of a part of the right-hand side constants in (2.101) which

are positive, and the corresponding coefficient matrix of the left-hand side of the constraints is denoted by  $A^+ = [p_1^+, \dots, p_n^+]$ .

### Decoding algorithm using a reference solution

- Step 1: Let  $j := 1$  and  $psum := 0$ .  
 Step 2: If  $g_{s(j)} = 0$ , set  $q_{s(j)} := 0$  and  $j := j + 1$ , and go to step 4.  
 Step 3: If  $psum + g_{s(j)}p_{s(j)}^+ \leq b^+$ , set  $q_{s(j)} := g_{s(j)}$ ,  $psum := psum + g_{s(j)}p_{s(j)}^+$  and  $j := j + 1$ . Otherwise, set  $q_{s(j)} := 0$  and  $j := j + 1$ .  
 Step 4: If  $j \leq n$ , return to step 2.  
 Step 5: Let  $j := 1$ ,  $l := 0$  and  $sum := 0$ .  
 Step 6: If  $g_{s(j)} = 0$ , set  $j := j + 1$  and go to step 8. If  $g_{s(j)} \neq 0$ , set  $sum := sum + g_{s(j)}p_{s(j)}$ .  
 Step 7: If  $sum \leq b$ , set  $l := j$ ,  $j := j + 1$ . Otherwise, set  $j := j + 1$ .  
 Step 8: If  $j \leq n$ , return to step 6.  
 Step 9: If  $l \leq 0$ , go to step 11.  
 Step 10: For  $x_{s(j)}$  satisfying  $1 \leq j \leq l$ , let  $x_{s(j)} := g_{s(j)}$ . For  $x_{s(j)}$  satisfying  $l + 1 \leq j \leq n$ , let  $x_{s(j)} := 0$ , and the algorithm terminates.  
 Step 11: Let  $sum := \sum_{k=1}^n x_{s(k)}^* p_{s(k)}$  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.  
 Step 13: If  $sum - x_{s(j)}^* p_{s(j)} + g_{s(j)} p_{s(j)} \leq b$ , set  $sum := sum - x_{s(j)}^* p_{s(j)} + g_{s(j)} p_{s(j)}$  and  $x_{s(j)} := g_{s(j)}$ , and go to step 16.  
 Step 14: Let  $t_{s(j)} := \lfloor 0.5(x_{s(j)}^* + g_{s(j)}) \rfloor$ .  
 Step 15: If  $sum - x_{s(j)}^* p_{s(j)} + t_{s(j)} p_{s(j)} \leq b$ , set  $sum := sum - x_{s(j)}^* p_{s(j)} + t_{s(j)} p_{s(j)}$ ,  $g_{s(j)} := t_{s(j)}$  and  $x_{s(j)} := t_{s(j)}$ . Otherwise, set  $x_{s(j)} := x_{s(j)}^*$ .  
 Step 16: If  $j > n$ , the algorithm terminates. Otherwise, return to step 12.

For general integer programming problems involving positive and negative coefficients in the constraints, this decoding algorithm produces feasible solution from any individuals in the population. However, the diversity of generated feasible solutions depends on the reference solution used in the decoding algorithm. To overcome this difficulty, GADSLPRRSU adopts the reference solution updating procedure in which the current reference solution is updated by another feasible solution if the diversity of generated solutions seems to be lost.

It is expected that an optimal solution to the following linear programming relaxation problem becomes a good approximate optimal solution of the original integer programming problem (2.101):

$$\left. \begin{array}{ll} \text{minimize} & cx \\ \text{subject to} & Ax \leq b \\ & 0 \leq x_j \leq v_j, \quad j = 1, \dots, n. \end{array} \right\} \quad (2.102)$$

From this observation, GADSLPRRSU exploits the information about the optimal solution to the continuous relaxation problem to find an approximate optimal solution with high accuracy in reasonable time. The computational procedure of GADSLPRRSU is summarized as follows:

**Computational procedure of GADSLPRRSU**

- Step 0: Determine values of the parameters used in the genetic algorithm.
- Step 1: Generate the initial population consisting of  $N$  individuals based on the information of an optimal solution to the continuous relaxation problem (2.102).
- 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, the procedure stops. Otherwise, let  $t := t + 1$ .
- Step 4: Apply the reproduction operator using the elitist expected value selection after linear scaling.
- Step 5: Apply the crossover operator, called PMX (Partially Matched Crossover) for a double string.
- Step 6: Apply the mutation based on the information of an optimal solution to the continuous relaxation problem (2.102).
- Step 7: Apply the inversion operator, and return to step 2.

For more information about GADSLPRRSU, it is recommended to read Sakawa (2001).

Fuzzy Stochastic Multiobjective Programming

Sakawa, M.; Nishizaki, I.; Katagiri, H.

2011, XII, 264 p., Hardcover

ISBN: 978-1-4419-8401-2