Pergamon

Computers Math. Applic. Vol. 34, No. 12, pp. 89–104, 1997

0898-1221/97 $17.00 + 0.00

PII: S0898-1221(97)00237-X

Dependent-Chance Programming: A Class of Stochastic Optimization

BAODING LIU
Department of Applied Mathematics, Tsinghua University
Beijing 100084, P.R. China

(Received February 1997; accepted April 1997)

Abstract—This paper provides a theoretical framework of dependent-chance programming, as well as dependent-chance multiobjective programming and dependent-chance goal programming which are new types of stochastic optimization. A stochastic simulation based genetic algorithm is also designed for solving dependent-chance programming models.

Keywords—Stochastic programming, Genetic algorithm, Dependent-chance programming.

1. INTRODUCTION

Management decisions are usually made in uncertain environments. Stochastic programming offers a means of considering the objectives and constraints with stochastic parameters. The theory of stochastic programming has been summarized by several books such as [1–7].

Complex decision systems are usually multidimensional, multifaceted, multifunctional, and multicriteria. With the requirement of considering randomness, appropriate techniques of stochastic programming have been developed to suit the different purposes of management. The first method dealing with stochastic parameters in stochastic programming is the so-called *expected value models* which optimize the expected objective functions subject to some expected constraints. The second, *chance constrained programming*, was pioneered by Charnes and Cooper [8] as a means of handling uncertainty by specifying a confidence level at which it is desired that the stochastic constraint holds. However, sometimes a decision system undertakes multiple certain tasks called events, and the decision maker wishes to maximize the probabilities of satisfying these events. In order to solve this problem, this paper provides a theoretical framework of the third type of stochastic programming named *dependent-chance programming* and extends it to dependent-chance multiobjective programming and dependent-chance goal programming.

Roughly speaking, the dependent-chance programming is related to maximizing some chance functions of events defined on stochastic sets in a complex uncertain decision system. In deterministic models as well as expected value models and chance constrained programming, the feasible set is essentially assumed to be deterministic after the real problem is modelled, i.e., an optimal solution is always given regardless of whether or not it can be performed in practice. However, the given solution may not be performed if the realization of uncertain parameter goes to bad cases. So the dependent-chance programming model never assumes that the feasible set is deterministic. Although a deterministic solution is given by the dependent-chance programming model, this solution is only requested to be performed as much as possible. This special feature of

dependent-chance programming is very different from the existing stochastic programming techniques. However, such problems do exist in the real world some real and potential applications of dependent-chance programming have been presented by Liu and Ku [9], Liu [10], and Liu and Iwamura [11].

In this paper, we present a concept of stochastic sets for describing the stochastic constraints in an uncertain environment as the basis of dependent-chance programming, as well as dependent-chance multi-objective programming and dependent-chance goal programming, and design a stochastic simulation based genetic algorithm for solving dependent-chance programming models.

2. A BACKGROUND: SUPPLY-ALLOCATION SYSTEM

A lot of decision systems, including a supply-allocation system, can be represented by Figure 1.

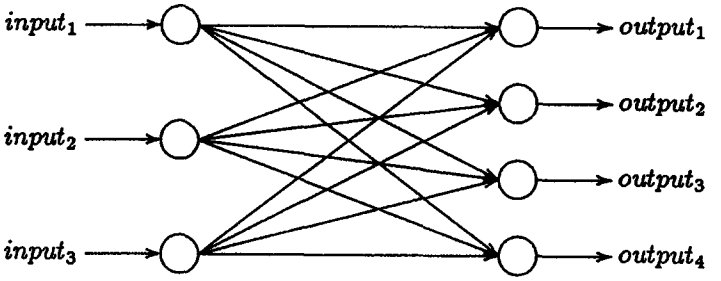

Figure 1. A supply-allocation system.

As an illustrative example, in Figure 1, there are three inputs representing three locations of resources and four outputs representing the demands of four users. We must answer the following two types of questions.

(a) Supply problems. In order to achieve certain goals in the future, a decision must be made concerning the present and future situation. That is, we must determine the optimal combination of *inputs* in certain time, for example, to determine the quantities ordered from the three inputs.

(b) Allocation problems. One of the basic allocation problems is the optimal allocation of the resources. The task of this part is to determine the *outputs* that result from various combinations of resources such that the certain goals are achieved. Certainly, in this system, supply and allocation problems should not be separate.

In order to answer the above two types of problems, we use 12 decision variables $x_1, x_2, \ldots, x_{12}$ to represent an action, where x_1, x_2, x_3, x_4 are quantities ordered from $input_1$ to outputs 1,2,3,4, respectively; x_5, x_6, x_7, x_8 from $input_2$; $x_9, x_{10}, x_{11}, x_{12}$ from $input_3$. In practice, some variables may vanish because of some physical constraints.

We mention that the inputs are available outside resources; they have their own properties. For example, the maximum quantities supplied by the three resources are marked by ξ_1, ξ_2, ξ_3 which may be stochastic variables characterized by density functions ϕ_1, ϕ_2, ϕ_3, respectively. Thus at first, we have the following constraint,

$$\begin{aligned} x_1 + x_2 + x_3 + x_4 &\leq \xi_1, \\ x_5 + x_6 + x_7 + x_8 &\leq \xi_2, \\ x_9 + x_{10} + x_{11} + x_{12} &\leq \xi_3. \end{aligned} \tag{1}$$

If at least one of ξ_1, ξ_2, and ξ_3 is really stochastic, then this constraint is not clear at all because we cannot make a decision before knowing the realization of ξ_1, ξ_2, and ξ_3. In order to make it meaningful, we will employ a so-called *stochastic set* to describe it. The other clear constraint is

$$x_i \geq 0, \qquad i = 1, 2, \ldots, 12, \tag{2}$$

which represents that the quantities ordered from the resources are not negative.

Since our object of study is a decision system, some decision criteria to such a system unquestionably exist. We will regard each criterion as an evaluation on a certain event in this system. For example, the four outputs have their own demands marked by c_1, c_2, c_3, and c_4, respectively. Then, we have the following four events:

$$\begin{aligned} x_1 + x_5 + x_9 &= c_1, \\ x_2 + x_6 + x_{10} &= c_2, \\ x_3 + x_7 + x_{11} &= c_3, \\ x_4 + x_8 + x_{12} &= c_4. \end{aligned} \tag{3}$$

These equalities imply that the decisions should satisfy the demands. In view of the uncertainty of this system, we are not sure whether a decision can be performed before knowing the realization of stochastic variables, so we have to employ so-called *chance functions* to evaluate these four events. Let

$$\begin{aligned} f_1(\mathbf{x}) &= \Pr\{x_1 + x_5 + x_9 = c_1\}, \\ f_2(\mathbf{x}) &= \Pr\{x_2 + x_6 + x_{10} = c_2\}, \\ f_3(\mathbf{x}) &= \Pr\{x_3 + x_7 + x_{11} = c_3\}, \\ f_4(\mathbf{x}) &= \Pr\{x_4 + x_8 + x_{12} = c_4\}, \end{aligned} \tag{4}$$

where Pr denotes the probability of the event in $\{\cdot\}$. Although all x_i and c_i themselves are not stochastic variables, the events in the brackets indeed possess randomness because x_i's are constrained by stochastic parameters. Usually, we hope to maximize the four chance functions, i.e., increase the reliability levels of the four events as high as possible. In addition, we may wish to use $input_3$ as little as possible because of some economic or political reasons. Thus, the fifth event is the quantity ordered from $input_3$. The respective criterion is described by

$$\min f_5(\mathbf{x}) = x_9 + x_{10} + x_{11} + x_{12}, \tag{5}$$

which is a deterministic objective.

Until now we have formed our stochastic programming to the supply-allocation problem in an uncertain environment, i.e.,

$$\begin{aligned} \max f_1(\mathbf{x}) &= \Pr\{x_1 + x_5 + x_9 = c_1\}, \\ \max f_2(\mathbf{x}) &= \Pr\{x_2 + x_6 + x_{10} = c_2\}, \\ \max f_3(\mathbf{x}) &= \Pr\{x_3 + x_7 + x_{11} = c_3\}, \\ \max f_4(\mathbf{x}) &= \Pr\{x_4 + x_8 + x_{12} = c_4\}, \\ \min f_5(\mathbf{x}) &= x_9 + x_{10} + x_{11} + x_{12}, \\ \text{subject to:} \quad & \\ x_1 + x_2 + x_3 + x_4 &\le \xi_1, \\ x_5 + x_6 + x_7 + x_8 &\le \xi_2, \\ x_9 + x_{10} + x_{11} + x_{12} &\le \xi_3, \\ x_i &\ge 0, \qquad i = 1, 2, \ldots, 12, \end{aligned} \tag{6}$$

where ξ_1, ξ_2, and ξ_3 are stochastic variables characterized by density functions ϕ_1, ϕ_2, and ϕ_3, respectively. In this stochastic programming, we cannot discuss the realization of each decision variable in isolation. For example, we have to consider all of x_1, x_2, x_3, and x_4 simultaneously, but separately. This means that some variables, as well as the chance functions, are stochastically dependent. We will call the stochastic programming like (6) as *dependent-chance programming* (DCP).

We use the term *dependent-chance programming* for the following reasons:

(i) some constraints are stochastic;

(ii) some objectives are chances of certain events;

(iii) some decision variables are stochastically dependent; and

(iv) the chances of some events are dependent on each other. Chance dependence means that we cannot consider the reliability levels of each individual event in isolation; this usually implies that the objectives and constraints cannot be converted to their deterministic equivalents.

3. STOCHASTIC SETS

In classical mathematical programming, the feasible set is usually represented by a crisp set which is normally defined as a collection of elements $x \in X$. Each single element can either belong to or not belong to a set A, $A \subset X$. Such a crisp set can be described in different ways: one can either list the elements that belong to the set; describe the set analytically by a sequence of equalities and inequalities (constraints); or define the member elements by using the characteristic function, in which 1 indicates membership and 0 nonmembership. Recall that a fuzzy set is described by a membership function. Analogously, as an extension of deterministic set, we use a probability function to describe a feasible set. Probability is 1 means the statement x *belongs to* A is true, 0 the statement is false. When the boundaries of a feasible set are defined by some stochastic factors, the probability function does allow various values between 0 and 1. This case arises in stochastic constraints of stochastic mathematical programming. In order to describe such a kind of set, we employ a term *stochastic set.* Following the definition of fuzzy set, we define the stochastic set in the way that follows.

DEFINITION 1. *If Ω is a collection of objects denoted generally by x, then the stochastic set A in Ω is defined as a set of ordered pairs:*

$$A = \{(x, \mu_A(x)) \mid x \in \Omega\}, \tag{7}$$

where $\mu_A(x)$ is called the probability function of x in A.

In the above definition, the statement that $\mu_A(x)$ is the probability function of x in A means that the probability of realization of x in A is $\mu_A(\mathbf{x})$ in some given stochastic environment.

For example, we suppose that the water marks ξ_1 and ξ_2 of a reservoir and a river have densities ϕ_1 and ϕ_2, respectively. How much water can we draw from the reservoir and river? We cannot answer it deterministically. Assume $\mathbf{x} = (x_1, x_2)$ denotes a vector whose first element x_1 is the water quantity drawn from the reservoir, second element x_2 drawn from the river. Theoretically, all possible realizing vectors $\mathbf{x}$ form the first quadrant of the Euclidean plane. Let $\Omega = R_2^+$ and A be the set of all possible realizing vectors satisfying $1 \leq x_1 + x_2 \leq 5$. Recall that the stochastic constraints $x_1 \leq \xi_1$, $x_2 \leq \xi_2$, i.e., we cannot draw water more than the existing volumes in storage. Thus, A is a stochastic set whose probability function can be written as follows:

$$\mu_A(\mathbf{x}) = \Pr\{(x_1, x_2) \mid 0 \leq x_1 \leq \xi_1;\ 0 \leq x_2 \leq \xi_2;\ 1 \leq x_1 + x_2 \leq 5\},$$

which is equivalent to

$$\mu_A(\mathbf{x}) = \begin{cases} \int_{x_1}^{+\infty} \phi_1(\xi)\, d\xi \cdot \int_{x_2}^{+\infty} \phi_2(\xi)\, d\xi, & x_1 \geq 0, \quad x_2 \geq 0, \quad 1 \leq x_1 + x_2 \leq 5, \\ 0, & \text{otherwise.} \end{cases}$$

We mention that the probability function $\mu_A(x)$ is determined by two factors, one is from the realizations of stochastic variables, for example, $x_1 \leq \xi_1$ and $x_2 \leq \xi_2$, the other is from the real constraints given by the decision maker or physical system, for example, $x_1 \geq 0$, $x_2 \geq 0$ and $1 \leq x_1 + x_2 \leq 5$.

From the definition of stochastic set, we know that $\mu_A(\mathbf{x}) = 0$ means that $\mathbf{x}$ is impossible to be realized; $\mu_A(\mathbf{x}) = 1$ implies that we can reach $\mathbf{x}$ for any realizations of stochastic variables; $\mu_A(\mathbf{x}) = \alpha (0 < \alpha < 1)$ represents that the probability that $\mathbf{x}$ can be performed is α.

If $\mu_A(\mathbf{x})$ only maps Ω to the two points 0 and 1, A is deterministic and $\mu_A(\mathbf{x})$ is identical to the characteristic function.

Similar to the concept of α-level set in fuzzy set theory, we define the α-level set of stochastic set as follows.

DEFINITION 2. *The set of elements that belong to the stochastic set A at least to the probability α is called the α-level set:*

$$A_\alpha = \{x \in \Omega \mid \mu_A(x) \geq \alpha\}. \tag{8}$$

4. STOCHASTIC CONSTRAINTS

We mention that a stochastic programming like

$$\begin{aligned} &\max f(\mathbf{x}, \xi), \\ \text{subject to:} \quad &g_i(\mathbf{x}, \xi) \leq 0, \qquad i = 1, 2, \ldots, p \end{aligned} \tag{9}$$

is not well defined since the meanings of max as well as of the constraints are not clear at all if ξ is a stochastic vector. Therefore, some techniques attempt to present some meaningful forms of stochastic programming. As a successful technique, a typical formulation of chance constrained programming (CCP) may be written as follows:

$$\begin{aligned} &\max E_\xi f(\mathbf{x}, \xi), \\ \text{subject to:} \quad &\Pr\{\xi \mid g_i(\mathbf{x}, \xi) \leq 0,\ i = 1, 2, \ldots, p\} \geq \alpha, \end{aligned} \tag{10}$$

where $\mathbf{x}$ is a decision vector, ξ is a stochastic vector, $f(\mathbf{x}, \xi)$ is the return function, E_ξ denotes the expected value operator on ξ, $g_i(\mathbf{x}, \xi)$ are stochastic constraint functions, $i = 1, 2, \ldots, p$, $\Pr\{\cdot\}$, denotes the probability of the event in $\{\cdot\}$, and α is a predetermined confidence level. So a point $\mathbf{x}$ is feasible if and only if the probability measure of the set $\{\xi \mid g_i(\mathbf{x}, \xi) \leq 0, i = 1, 2, \ldots, p\}$ is at least α. In other words, the constraints will be violated at most $(1 - \alpha)$ of time. Thus, the stochastic programming is converted to its deterministic equivalent via CCP (10). Recall the concept of stochastic sets proposed in Section 3; we can represent the stochastic constraint in (9) by a stochastic set S as

$$S = \{(\mathbf{x}, \mu_S(\mathbf{x})) \mid \mathbf{x} \in R^n\}, \tag{11}$$

where the probability function $\mu_S(\mathbf{x})$ is defined by

$$\mu_S(\mathbf{x}) = \Pr\{\xi \mid g_i(\mathbf{x}, \xi) \leq 0,\ i = 1, 2, \ldots, p\}, \tag{12}$$

then the probabilistic constraint set in (10) is just the α-level set S_α of stochastic set S defined by equation (11). So CCP (10) reads as

$$\max_{\mathbf{x} \in S_\alpha} E_\xi f(\mathbf{x}, \xi). \tag{13}$$

We also mention that when $\alpha = 1$, the optimal solution is just a fat solution since $\mathbf{x} \in S_1$ is always feasible for any realization of stochastic vector ξ. Certainly, this form of CCP does not do anything rather than (10).

Different from the deterministic case, as well as expected value model and chance constrained programming, we cannot say a point is feasible or not when our problem is defined on a stochastic set. We have to say a point $\mathbf{x}^*$ is feasible with a probability α, where α is the value of probability function $\mu_S(\mathbf{x}^*)$.

Perhaps the first problem of application of stochastic set is how to get the probability function of a stochastic set for a given sequence of stochastic constraints. Fortunately, it is not difficult to

calculate the probability functions $\mu_S(\mathbf{x})$ by equation (12) for real management problems. As a useful example, we consider the following set of stochastic constraints:

$$r_j(\mathbf{x}) \leq \xi_j, \qquad j = 1, 2, \ldots, p, \tag{14}$$

where the stochastic variables ξ_j have density functions $\phi_j(\cdot)$, $j = 1, 2, \ldots, p$, respectively. Then the set of all $\mathbf{x}$ satisfying (14) is a stochastic set labeled by S with probability function $\mu_S(\mathbf{x})$ which is able to be determined by

$$\mu_S(\mathbf{x}) = \int_{r_1(\mathbf{x})}^{+\infty} \phi_1(\xi)\, d\xi \cdots \int_{r_p(\mathbf{x})}^{+\infty} \phi_p(\xi)\, d\xi.$$

When the constraints (12) fail to be regular or hard to be calculated, it is more convenient to calculate it by the stochastic simulation (also called Monte Carlo simulation). Suppose that $\xi = (\xi_1, \xi_2, \ldots, \xi_t)$ is a t-dimensional stochastic vector, and each ξ_i has a given distribution. For any given $\mathbf{x}$, we use the following stochastic simulation technique to compute $\mu_S(\mathbf{x})$. We generate N independent random vectors $\xi^{(i)} = (\xi_1^{(i)}, \xi_2^{(i)}, \ldots, \xi_t^{(i)})$, $i = 1, 2, \ldots, N$, from their probability distributions, where the generating methods have been well discussed by numerous literature and summarized by Rubinstein [12]. Let N' be the number of occasions on which

$$g_j\left(\mathbf{x}, \xi^{(i)}\right) \leq 0, \qquad j = 1, 2, \ldots, p,$$

i.e., the number of random vectors satisfying the constraints. Then, by the basic definition of probability, $\mu_S(\mathbf{x})$ can be estimated by

$$\mu_S(\mathbf{x}) = \frac{N'}{N}. \tag{15}$$

Certainly, this estimation is approximate and may change from a simulation to another. But it is available to real practice problems since the determination of the confidence level α itself is not precise. At least, this estimation is satisfactory in practice.

Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. For the sake of clear statements, we call $\mathbf{x}$ the decision vector and x_i, $i = 1, 2, \ldots, n$ the decision components throughout this paper. Sometimes the term decision is omitted for brevity.

We have mentioned in Section 2 that some decision components are stochastically dependent. We say two components x' and x'' are stochastically dependent if the realization of x' is dependent on x'', and vice versa, where dependence means that the chance of each of x' and x'' is a function of at least x' and x''. However, unfortunately, the probability function itself does not provide the detailed information as to whether components are stochastically dependent or not. So we have to analyze them from the original material.

Let us consider the stochastic constraints (1), we find that $\{x_1, x_2, x_3, x_4\}$ is a stochastically dependent group because the realization of any one of them is dependent on each other. Moreover, the chance of any one when realizing x_1, x_2, x_3, and x_4 is a function of all of them. In fact,

$$f(x_i) = \int_{x_1+x_2+x_3+x_4}^{+\infty} \phi_1(\xi)\, d\xi, \qquad i = 1, 2, 3, 4,$$

where $f(x_i)$ is the chance of realization of x_i. Similar reason shows that $\{x_5, x_6, x_7, x_8\}$ and $\{x_9, x_{10}, x_{11}, x_{12}\}$ are the other two groups. On the other hand, there is no stochastic relationship among the above three groups. Please mention that we do not regard a deterministic relationship as a special stochastic relationship. This property is very important in discussion of the multiple objective case.

In this paper, we suppose that we know the following stochastic relationship among the decision components.

STOCHASTIC RELATIONSHIP. *There is a known partition of n components of a decision vector into k groups such that these k groups are mutually stochastically independent and in each group any elements are stochastically dependent and have the same chance to appear if they require to be realized simultaneously.*

For our supply-allocation system, if we only want to satisfy $output_1$ and $output_2$, then the components $x_1, x_2, x_5, x_6, x_9, x_{10}$ require to be realized simultaneously. Thus, by the partition $\{x_1, x_2, x_3, x_4\}$, $\{x_5, x_6, x_7, x_8\}$, and $\{x_9, x_{10}, x_{11}, x_{12}\}$, we know that x_1 and x_2 are dependent and their chance of occurrence is

$$f(x_i) = \int_{x_1+x_2}^{+\infty} \phi_1(\xi)\, d\xi, \qquad i = 1, 2,$$

because x_3 and x_4 do not require to be realized. The same case also occurs in the groups $\{x_5, x_6\}$ and $\{x_9, x_{10}\}$.

In practice, this stochastic relationship is always clear and can be obtained easily. Theoretically, we will find that the algorithm for dependent-chance multi-objective programming, as well as dependent-chance goal programming cannot work if we do not input such a kind of stochastic relationship on a stochastic set in the computer program.

5. CHANCE OBJECTIVE FUNCTIONS

In stochastic programming, we have three types of objective functions for different evaluation criteria.

(i) Expected target, for example, expected cost minimization, expected profit maximization, etc. This is also available to the deterministic case if we regard the probability of the event as 1.

(ii) Target with a limit on chance, i.e., given a chance which is usually represented by a confidence level, the target with a lower/upper limit on chance is to be optimized.

(iii) Chance of a certain event, i.e., given an event, the chance of this event is to be optimized.

Types (i) and (ii) are well discussed by expected value models and chance constrained programming, respectively. Here we consider type (iii). We suppose that the chance of each certain event will be maximized, otherwise, we can define the chance as the probability of its complement. We have announced that the chance function is a probability of a certain event. For example, the following equality

$$x_1 + x_5 + x_9 = c_1$$

means satisfying the demand of $output_1$ in the supply-allocation system. In fact, the decision vectors $\mathbf{x} = (x_1, x_2, \ldots, x_{12})$ meeting the event also form a stochastic set. (Deterministic set is a special stochastic set!) We denote this stochastic set by E with probability function $\mu_E(\mathbf{x})$. Here, $\mu_E(\mathbf{x})$ is identical to the characteristic function of E since it is deterministic. Thus,

$$\mu_E(\mathbf{x}) = \begin{cases} 1, & x_1 + x_5 + x_9 = c_1, \\ 0, & \text{otherwise.} \end{cases}$$

We are now concerned with the following questions. What is the probability of this event on a stochastic set S? How do we represent the chance function?

It is clear that this event is met only on the stochastic set $E \cap S$ which is an intersection of event E and stochastic constraint S, and

$$\mu_{E\cap S}(\mathbf{x}) = \mu_E(\mathbf{x}) \cdot \mu_S(\mathbf{x}) = \begin{cases} \mu_S(\mathbf{x}), & \mathbf{x} \in E, \\ 0, & \mathbf{x} \notin E. \end{cases} \tag{16}$$

Recall the example, the set E is composed of all possible decisions $\mathbf{x} = (x_1, x_2, \ldots, x_{12})$ meeting the event $x_1 + x_5 + x_9 = c_1$. However, we are only interested in three components of them, x_1, x_5,

and x_9, but not the others. That is, we are only concerned with the realizations of x_1, x_5, and x_9 which meet the given event $x_1 + x_5 + x_9 = c_1$. The realizations of other components will be ignored. Let $V(E)$ denote the set of all components of $\mathbf{x}$ in which we are interested. We mention that only x_1, x_5, and x_9 are necessary components for meeting the event $x_1 + x_5 + x_9 = c_1$, thus $V(E) = \{x_1, x_5, x_9\}$.

Generally, let the single chance function be $f(\mathbf{x})$ which is a probability of given event named E. The event E is usually represented by all possible decisions meeting the demand of E. We use $V(E)$ to denote the set of all components of $\mathbf{x}$ which are necessary to this event. Since we are only interested in one event, this event will claim precedence over all other potential events, thus the chance objective function will be the maximum probability of realizations of components in $V(E)$, i.e.,

$$f(\mathbf{x}) = \begin{cases} \max\limits_{\mathbf{y} \in E^*} \mu_S(\mathbf{y}), & \mathbf{x} \in E, \\ 0, & \mathbf{x} \notin E, \end{cases} \tag{17}$$

where

$$E^* = \{\mathbf{y} = (y_1, y_2, \ldots, y_n) \in E \mid y_j = x_j \text{ if } x_j \in V(E),\ j = 1, 2, \ldots, n\}. \tag{18}$$

Suppose that the original stochastic constraints of our problem are

$$g_j(\mathbf{x}, \xi) \leq 0, \qquad j = 1, 2, \ldots, p, \tag{19}$$

where ξ is a stochastic vector. Then the probability function of stochastic set is

$$\mu_S(\mathbf{x}) = \Pr\{\xi \mid g_j(\mathbf{x}, \xi) \leq 0,\ j = 1, 2, \ldots, p\}. \tag{20}$$

In practice, for each $\mathbf{x} \in E$, it is not difficult to determine the values x_j^* of decision components out of $V(E)$ for the decision vector $\mathbf{x}^*$,

$$\mathbf{x}^* = (x_1, x_2, \ldots, x_n) :\ x_j = \begin{cases} x_j, & x_j \in V(E), \\ x_j^*, & x_j \notin V(E), \end{cases} \quad 1 \leq j \leq n \tag{21}$$

such that

$$\max_{\mathbf{y} \in E^*} \mu_S(\mathbf{y}) = \Pr\{\xi \mid g_j(\mathbf{x}^*, \xi) \leq 0,\ j = 1, 2, \ldots, p\}. \tag{22}$$

Therefore, the chance function of the event E is

$$f(\mathbf{x}) = \begin{cases} \Pr\{\xi \mid g_j(\mathbf{x}^*, \xi) \leq 0,\ j = 1, 2, \ldots, p\}, & \mathbf{x} \in E, \\ 0, & \mathbf{x} \notin E. \end{cases} \tag{23}$$

Usually, the point $\mathbf{x}^*$ associated with $\mathbf{x}$ occurs at some extreme place. For example, in the supply-allocation system, the values x_j^* of decision components out of $V(E)$ should be zero. We will call the constraints

$$g_j(\mathbf{x}^*, \xi) \leq 0, \qquad j = 1, 2, \ldots, p \tag{24}$$

as the *induced constraints* on the event E. The induced constraints will play an important role in calculating the chance functions by using stochastic simulation technique.

6. DEPENDENT-CHANCE PROGRAMMING

In this section, we consider the simplest dependent-chance programming, i.e., the single objective case. A typical formulation of dependent-chance programming (DCP) is given as follows:

$$\max_{\mathbf{x} \in S} f(\mathbf{x}), \tag{25}$$

or represented as

$$\begin{aligned} &\max f(\mathbf{x}), \\ \text{subject to:} \quad &g_j(\mathbf{x},\xi) \le 0, \qquad j = 1,2,\ldots,p, \end{aligned} \tag{26}$$

where $\mathbf{x}$ is an n-dimensional decision vector, S is a stochastic set on R^n with probability function

$$\mu_S(\mathbf{x}) = \Pr\{\xi \mid g_j(\mathbf{x},\xi) \le 0,\ j = 1,2,\ldots,p\}$$

without known stochastic relationship assumed at Section 4, $f(\mathbf{x})$ is a chance function of certain event, borrowing the symbol $\in$ from classical set theory, $\mathbf{x} \in S$ means that $\mathbf{x}$ is feasible with probability $\mu_S(\mathbf{x})$. Please mention that the set S is stochastic but deterministic. A point $\mathbf{x}^* \in S$ is called an optimal solution of (25) if $f(\mathbf{x}^*) \ge f(\mathbf{x})$ for any $\mathbf{x} \in S$. Let E denote the set of all $\mathbf{x}$ meeting the event, then the chance objective function is determined by (17) and (18) or (23). Thus, dependent-chance programming (DCP) (25) is equivalent to

$$\max_{\mathbf{x}\in S} f(\mathbf{x}) = \max_{\mathbf{x}\in E} \max_{\mathbf{y}\in E^*} \mu_S(\mathbf{y}) = \max_{\mathbf{x}\in E} \mu_S(\mathbf{x}). \tag{27}$$

In this equation, the DCP (25) is proved to be equivalent to $\max_{\mathbf{x}\in E} \mu_S(\mathbf{x})$. But we should not think that we can generally represent DCP by the mathematical programming $\max_{\mathbf{x}\in E} \mu_S(\mathbf{x})$, because it is only a special case for single objective DCP.

We now go back to our supply-allocation system. Suppose that the stochastic constraint is (3) together with $x_i \ge 0$, $i = 1,2,\ldots,12$. Then this constraint can be represented by a stochastic set S with probability function

$$\mu_S(\mathbf{x}) = \begin{cases} \int_{x'}^{+\infty} \phi_1(\xi)\,d\xi \cdot \int_{x''}^{+\infty} \phi_2(\xi)\,d\xi \cdot \int_{x'''}^{+\infty} \phi_3(\xi)\,d\xi, & x_i \ge 0, \forall i, \\ 0, & \text{otherwise}, \end{cases}$$

where $x' = x_1+x_2+x_3+x_4$, $x'' = x_5+x_6+x_7+x_8$, and $x''' = x_9+x_{10}+x_{11}+x_{12}$. Here we suppose that ϕ_1, ϕ_2, and ϕ_3 are two-parameter lognormal densities with parameters $u = 1.56, 1.36, 0.95$, and $\sigma = 0.56, 0.45, 0.38$, respectively. Our single interesting event is to satisfy the demand $c_1 = 6$ of $output_1$, i.e., $x_1 + x_5 + x_9 = c_1$. Thus the set E associated with this event is

$$E = \{\mathbf{x} = (x_1, x_2, \ldots, x_{12}) \mid x_1 + x_5 + x_9 = c_1 = 6\},$$

whose probability function is

$$\mu_E(\mathbf{x}) = \begin{cases} 1, & x_1 + x_5 + x_9 = 6, \\ 0, & \text{otherwise}. \end{cases}$$

By equation (27), our problem is written as

$$\max_{\mathbf{x}\in E} \mu_S(\mathbf{x}), \tag{28}$$

which is equivalent to

$$\max_{\mathbf{x}\in E'} \int_{x'}^{+\infty} \phi_1(\xi)\,d\xi \cdot \int_{x''}^{+\infty} \phi_2(\xi)\,d\xi \cdot \int_{x'''}^{+\infty} \phi_3(\xi)\,d\xi, \tag{29}$$

where $E' = \{\mathbf{y} \in E \mid y_i \ge 0,\ i = 1,2,\ldots,12\}$. Equation (29) is common mathematical programming. By the property of integral, (29) is equivalent to

$$\max_{\mathbf{x}\in E''} \int_{x_1}^{+\infty} \phi_1(\xi)\,d\xi \cdot \int_{x_5}^{+\infty} \phi_2(\xi)\,d\xi \cdot \int_{x_9}^{+\infty} \phi_3(\xi)\,d\xi, \tag{30}$$

where $E'' = \{\mathbf{y} \in E \mid y_i \ge 0, i = 1,5,9;\ y_i = 0, i \ne 1,5,9\}$. A run of the computer program for this problem shows that the optimal solution is $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_{12}^*)$ in which all x_i^* are zero except for that

$$x_1^* = 2.6, \qquad x_5^* = 2.1, \qquad x_9^* = 1.3,$$

with $f(\mathbf{x}^*) = 0.71$, i.e., the reliability level of this event is 71% if we give priority to the performance of $\mathbf{x}^*$.

7. DEPENDENT-CHANCE MULTI-OBJECTIVE PROGRAMMING

Since a complex decision system undertakes multiple tasks, there undoubtedly exist multiple potential objectives (some of them may not be chance functions) in the decision process. A typical formulation of dependent-chance multi-objective programming (DCMOP) is given as follows:

$$\max_{\mathbf{x}\in S} f(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})], \tag{31}$$

or represented as

$$\begin{aligned} &\max [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})], \\ \text{subject to:} \quad &g_j(\mathbf{x}, \xi) \leq 0, \quad j = 1, 2, \ldots, p, \end{aligned} \tag{32}$$

where $\mathbf{x}$ is an n-dimensional decision vector, S is a stochastic set with probability function $\mu_S(\mathbf{x})$ with known stochastic relationship assumed at Section 4, $f(\mathbf{x})$ is a vector of m real-valued functions f_i which are chance functions or not, the symbol $\mathbf{x} \in S$ means that $\mathbf{x}$ is feasible with probability $\mu_S(\mathbf{x})$.

We have given an algorithm for computing a single chance function on a stochastic set. Next, we consider the multiple objective case. If some of $f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})$ are not chance functions, then we can obtain the values of nonchance functions immediately. Without loss of generality, suppose that all of $f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})$ are chance functions. Each chance function $f_i(\mathbf{x})$ represents a probability of a certain event which is represented by

$$E_i = \{\mathbf{x} = (x_1, x_2, \ldots, x_n) \mid \mathbf{x} \text{ satisfies the event } i\},$$

for each i with $1 \leq i \leq m$. We write

$$E = E_1 \cap E_2 \cap \cdots \cap E_m$$

and

$$V(E) = V(E_1) \cup V(E_2) \cup \cdots \cup V(E_m),$$

where $V(E_i)$ are the sets of all necessary components to the events i, $i = 1, 2, \ldots, m$, respectively. Thus, $V(E)$ is the set of all necessary components to the m events. We also define $E^*, E_i^*, i = 1, 2, \ldots, m$ in the same way as equation (18). Based on the stochastic relationship, let $D(E)$ denote the set of all components which are stochastically dependent of any elements in $V(E)$. Thus, we have $V(E) \subset D(E)$ and $V(E_i) \subset D(E_i)$, $i = 1, 2, \ldots, m$.

Generally speaking, we have

$$\max_{\mathbf{y}\in E_i^*} \mu_S(\mathbf{y}) \geq f_i(\mathbf{x}) \geq \max_{\mathbf{y}\in E^*} \mu_S(\mathbf{y}), \qquad i = 1, 2, \ldots, m,$$

in which the first inequality means that the probability of realization of E_i in multi-event case is usually less than that in the single-event case because we cannot perform the event at the expense of others, the second means that the probability of realization of E_i in multi-event case is usually greater than the probability of realization of all events because some events are stochastically independent, i.e., the realization of one event does not need the realization of all others.

Now we discuss the relationship between a decision vector and its respective chance objective functions. In fact, our way is to realize each event, say E_i, as much as possible but not sacrifice the chances of other events. So we have to treat all elements in the stochastically dependent set $D(E_i)$ of $V(E_i)$ at an equitable level, i.e., these elements would have the same chance to be realized. On the other hand, we are not concerned with the elements out of $V(E)$ because they do not make any contribution to the events we are going to realize. Thus, we must consider all the elements in and only in $D(E_i) \cap V(E)$ simultaneously for the event E_i. By the stochastic relationship, we know that all of the elements in $D(E_i) \cap V(E)$ are independent of any other

elements in $V(E)$. Hence, we can perform the elements in $D(E_i) \cap V(E)$ as much as possible. Therefore, we have

$$f_i(\mathbf{x}) = \begin{cases} \max\limits_{\mathbf{y} \in E_i^*} \mu_S(\mathbf{y}), & \mathbf{x} \in E_i, \\ 0, & \mathbf{x} \notin E_i, \end{cases} \qquad i = 1, 2, \ldots, m,$$

where

$$E_i^* = \{\mathbf{y} = (y_1, \ldots, y_n) \in E_i \mid y_j = x_j \text{ if } x_j \in D(E_i) \cap V(E),\ j = 1, \ldots, n\},$$

for $i = 1, 2, \ldots, m$. After constructing a relationship between decision vectors and chance objective functions, we can calculate the chance functions directly or by the technique of stochastic simulation for complex stochastic constraints. Then we can solve DCMOP by utility theory if complete information of the preference function is given by the decision maker or finding all of the efficient solutions if no information is available. In practice, the decision maker can only provide partial information. In this case, we have to employ the so-called interactive methods such as feasible region reduction, weighting vector space reduction, criterion cone contraction, or line search [13].

8. DEPENDENT-CHANCE GOAL PROGRAMMING

Dependent-chance goal programming (DCGP) may be considered as an extension of goal programming in a complex stochastic decision system. When some management targets are given, the objective function may minimize the deviations, positive, negative or both, with a certain priority structure set by the decision maker. Then we may formulate the stochastic decision system as the following dependent-chance goal programming (DCGP):

$$\begin{aligned} & \min \sum_{j=1}^{l} P_j \sum_{i=1}^{m} (u_{ij} d_i^+ + v_{ij} d_i^-) \\ \text{subject to:} \quad & f_i(\mathbf{x}) + d_i^- - d_i^+ = b_i, && i = 1, 2, \ldots, m, \\ & d_i^-, d_i^+ \geq 0, && i = 1, 2, \ldots, m, \\ & \mathbf{x} \in S, \end{aligned} \tag{33}$$

or represented as

$$\begin{aligned} & \min \sum_{j=1}^{l} P_j \sum_{i=1}^{m} (u_{ij} d_i^+ + v_{ij} d_i^-) \\ \text{subject to:} \quad & f_i(\mathbf{x}) + d_i^- - d_i^+ = b_i, && i = 1, 2, \ldots, m, \\ & g_j(\mathbf{x}, \xi) \leq 0, && j = 1, 2, \ldots, p, \\ & d_i^-, d_i^+ \geq 0, && i = 1, 2, \ldots, m, \end{aligned} \tag{34}$$

where

P_j = the preemptive priority factor which expresses the relative importance of various goals, $P_j \gg P_{j+1}$, for all j,
u_{ij} = weighting factor corresponding to positive deviation for goal i with priority j assigned,
v_{ij} = weighting factor corresponding to negative deviation for goal i with priority j assigned,
d_i^+ = positive deviation from the target of goal i,
d_i^- = negative deviation from the target of goal i,
$\mathbf{x}$ = n-dimensional decision vector,
f_i = a function (chance or not): $\Re^n \to \Re^1$, in goal constraints,

b_i = the target value according to goal i,
l = number of priorities,
m = number of goal constraints,
S = a stochastic set with probability function $\mu_S(\mathbf{x})$ with known stochastic relationship assumed at Section 4, defined on $\Re^n$.

The key problem of solving DCGP is to calculate the value of $f_i(\mathbf{x})$ for any given solution $\mathbf{x}$. In the first priority level, we suppose that there are t goals which are listed as

$$f_i(\mathbf{x}) + d_i^- - d_i^+ = b_i, \qquad i = 1, 2, \ldots, t.$$

If some of $f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_t(\mathbf{x})$ are not chance functions, then we can obtain the respective deviations immediately. Without loss of generality, we assume that all of them are chance functions. Similar to the case of DCMOP, each chance function $f_i(\mathbf{x})$ represents a probability of a certain event which is represented by E_i. We write

$$E = E_1 \cap E_2 \cap \cdots \cap E_t$$

and

$$V(E) = V(E_1) \cup V(E_2) \cup \cdots \cup V(E_t).$$

Then, mentioning that the goals in a higher-priority level will be satisfied as much as possible regardless of the lower-priority goals, by a similar discussion of multi-objective case, we have

$$f_i(\mathbf{x}) = \begin{cases} \max\limits_{\mathbf{y} \in E_i^*} \mu_S(\mathbf{y}), & \mathbf{x} \in E_i, \\ 0, & \mathbf{x} \notin E_i, \end{cases} \qquad i = 1, 2, \ldots, t,$$

where

$$E_i^* = \{\mathbf{y} = (y_1, \ldots, y_n) \in E_i \mid y_j = x_j \text{ if } x_j \in D(E_i) \cap V(E),\ j = 1, \ldots, n\},$$

for $i = 1, 2, \ldots, t$.

In the second priority level, a similar process will yield the respective deviations except for the fact that E is replaced by

$$E = E_1 \cap \cdots \cap E_t \cap E_{t+1} \cap \cdots \cap E_{t'}$$

and

$$V(E) = V(E_1) \cup \cdots \cup V(E_t) \cup V(E_{t+1}) \cup \cdots \cup V(E_{t'}),$$

where $E_{t+1}, E_{t+2}, \ldots, E_{t'}$ are the sets of all $\mathbf{x}$ meeting the events associated with the chance functions $f_{t+1}(\mathbf{x}), f_{t+2}(\mathbf{x}), \ldots, f_{t'}(\mathbf{x})$ which are assumed to be the goals in this priority level because the decisions about events $E_1, E_2, \ldots, E_t$ have been made in a higher-priority level and the remaining decisions for lower-priority goals must follow the state resulting from the higher-priority decisions.

Certainly, the successive levels will be similarly discussed until all chance functions are solved.

We can solve dependent-chance goal programming by combining the above process and the techniques of nonlinear goal programming.

9. STOCHASTIC SIMULATION BASED GENETIC ALGORITHM

Genetic algorithms are a stochastic search method for optimization problems based on the mechanics of natural selection and natural genetics.Genetic algorithms have demonstrated con-

siderable success in providing good solutions to many complex optimization problems and received more and more attention during the past three decades. When the objective functions to be optimized in the optimization problems are multimodal or the search spaces are particularly irregular, algorithms need to be highly robust in order to avoid getting stuck at local optimal solution. The advantage of genetic algorithms is just to obtain the global optimal solution fairly. Genetic algorithms (including evolution programs and evolution strategies) have been well discussed and summarized by much literature, such as [14–16], and applied to a wide variety of problems.

In view of the complexity of stochastic programming models, we need to design a stochastic simulation based genetic algorithm for solving them. Iwamura and Liu [17] proposed a stochastic simulation based genetic algorithm for solving chance constrained programming models in which stochastic simulation is employed to check the chance constraints. Liu [10] designed a genetic algorithm for dependent-chance goal programming models. In this section, we only present a detailed algorithm for not only dependent-chance goal programming, but also dependent-chance programming and dependent-chance multi-objective programming. The effectiveness has been shown by numerous examples, and the readers may consult Liu [10].

9.1. Representation Structure

We can use a binary vector as a chromosome to represent real value of a decision variable, where the length of the vector depends on the required precision. The necessity for binary codings has received considerable criticism. An alternative approach to represent a solution is the floating point implementation in which each chromosome vector is coded as a vector of floating numbers, of the same length as the solution vector. Here, we use a vector $V = (x_1, x_2, \ldots, x_n)$ as a chromosome to represent a solution to the optimization problem, where n is the dimension.

9.2. Initialization Process

At first, we should eliminate all equality constraints by replacing some variables by the representation of the remaining variables. Then we define an integer pop_size as the number of chromosomes and initialize pop_size chromosomes randomly. Usually, it is difficult for complex optimization problems to produce feasible chromosome explicitly. So we employ one of the following two ways as the initialization process, depending on what kind of information the decision maker can give.

First case is that the decision maker can determine an interior point, denoted by V_0, in the constraint set. This is very possible for a real decision problem. We also need to define a large positive number M which ensures that all the genetic operators are probabilistically complete for the feasible solutions. This number M is used for not only initialization process, but also mutation operation. The pop_size chromosomes will be produced as follows. We randomly select a direction d in $\Re^n$ and define a chromosome V as $V_0 + M \cdot d$ if it is feasible for the inequality constraints, otherwise, we set M by a random number between 0 and M until $V_0 + M \cdot d$ is feasible. We mention that a feasible solution for the inequality constraints can be found in finite times by taking random number since V_0 is an interior point. Repeat this process pop_size times and produce pop_size initial feasible solutions $V_1, V_2, \ldots, V_{\text{pop\_size}}$.

If the decision maker fails to give such an interior point, but can predetermine a region which contains the feasible set. Usually, this region will be designed to have nice sharp, for example, an n-dimensional hypercube, because the computer can easily sample points from a hypercube. We generate a random point from the hypercube and check the the feasibility of this point. If it is feasible, then it will be accepted as a chromosome. If not, then regenerate a point from the hypercube randomly until a feasible one is obtained. Repeat the above process pop_size times, we can make pop_size initial feasible chromosomes $V_1, V_2, \ldots, V_{\text{pop\_size}}$.

9.3. Evaluation Function

Evaluation function, denoted by eval(V), is to assign a probability of reproduction to each chromosome V so that its likelihood of being selected is proportional to its fitness relative to the other chromosomes in the population, that is, the chromosomes with higher fitness will have more chance to produce offspring by using *roulette wheel selection.*

Let $V_1, V_2, \ldots, V_{\text{pop\_size}}$ be the pop_size chromosomes at the current generation. One well-known method is based on allocation of reproductive trials according to rank rather than actual objective values. No matter what kind of mathematical programming (single-objective or multi-objective), it is reasonable to assume that the user can give an order relationship among the pop_size chromosomes $V_1, V_2, \ldots, V_{\text{pop\_size}}$ such that the pop_size chromosomes can be rearranged from good to bad, i.e., the better the chromosome is, the smaller ordinal number it has. This arrangement is usually based on the values of objective functions. For the chance objective functions, as we discussed before, we can employ the stochastic simulation to calculate them in general. Now let a parameter $a \in (0,1)$ in the genetic system be given, then we can define the so-called *rank-based evaluation function* as follows:

$$\text{eval}(V_i) = a(1-a)^{i-1}, \qquad i = 1, 2, \ldots, \text{pop\_size}. \tag{35}$$

We mention that $i = 1$ means the best individual, $i = \text{pop\_size}$ the worst individual.

9.4. Selection Process

The selection process is based on spinning the roulette wheel pop_size times, each time we select a single chromosome for a new population in the following way.

Step 1. Calculate the cumulative probability q_i for each chromosome V_i,

$$\begin{aligned} q_0 &= 0, \\ q_i &= \sum_{j=1}^{i} \text{eval}(V_j), \qquad i = 1, 2, \ldots, \text{pop\_size}. \end{aligned} \tag{36}$$

Step 2. Generate a random real number r in $[0, q_{\text{pop\_size}}]$.

Step 3. Select the i^{th} chromosome V_i ($1 \le i \le \text{pop\_size}$) such that $q_{i-1} < r \le q_i$.

Step 4. Repeat steps 2 and 3 pop_size times and obtain pop_size copies of chromosomes.

9.5. Crossover Operation

We define a parameter P_c of a genetic system as the probability of crossover. This probability gives us the expected number $P_c \cdot \text{pop\_size}$ of chromosomes which undergo the crossover operation.

In order to determine the parents for crossover operation, let us do the following process repeatedly from $i = 1$ to pop_size: generating a random real number r from the interval $[0,1]$, the chromosome V_i is selected as a parent if $r < P_c$.

We denote the selected parents as $V_1', V_2', V_3', \ldots$ and divide them into the following pairs:

$$(V_1', V_2'), \quad (V_3', V_4'), \quad (V_5', V_6'), \quad \ldots.$$

Let us illustrate the crossover operator on each pair by (V_1', V_2'). At first, generate a random number c from the open interval $(0,1)$, then the crossover operator on V_1' and V_2' will produce two children X and Y as follows:

$$X = c \cdot V_1' + (1-c) \cdot V_2' \quad \text{and} \quad Y = (1-c) \cdot V_1' + c \cdot V_2'. \tag{37}$$

If the feasible set is convex, this arithmetical crossover operation ensures that both children are feasible if both parents are. However, in many cases, the feasible set is not necessarily convex or hard to verify the convexity. So we must check the feasibility of each child. If both children are feasible, then we replace the parents by them. If not, we keep the feasible one if it exists, and then redo the crossover operator by regenerating the random number c until two feasible children are obtained or a given number of cycles is finished. In this case, we only replace the parents by the feasible children.

9.6. Mutation Operation

We define a parameter P_m of a genetic system as the probability of mutation. This probability gives us the expected number of $P_m \cdot$ pop_size of chromosomes which undergo the mutation operations.

Similar to the process of selecting parents for crossover operation, we repeat the following steps from $i = 1$ to pop_size: generating a random real number r from the interval $[0, 1]$, the chromosome V_i is selected as a parent for mutation if $r < P_m$.

For each selected parent, denoted by $V = (x_1, x_2, \ldots, x_n)$, we mutate it in the following way. We choose a mutation direction d in $\Re^n$ randomly, if $V + M \cdot d$ is not feasible for the constraints, then we set M as a random number between 0 and M until it is feasible, where M is a large positive number defined in Section 9.2. If the above process cannot find a feasible solution in a predetermined number of iterations, then sets $M = 0$. We replace the parent V by its child $X = V + M \cdot d$.

Following selection, crossover, and mutation, the new population is ready for its next evaluation. The genetic algorithm will terminate after a given number of cyclic repetitions of the above steps. Following selection, crossover, and mutation, the new population is ready for its next evaluation. The genetic algorithm will terminate after a given number of cyclic repetitions of the above steps.

10. CONCLUSION

Following expected value model and chance-constrained programming, this paper developed the third kind of technique of stochastic programming, i.e., dependent-chance programming, dependent-chance multiobjective programming and dependent-chance goal programming. These techniques are available to the systems in which there are multiple stochastic inputs and multiple tasks whose reliability levels are required to be optimized. We also presented a stochastic simulation based genetic algorithm for solving dependent-chance programming models.

In order to understand the general concepts of dependent-chance programming correctly, we strongly remind the potential readers that the feasible set of dependent-chance programming is stochastic. However, if the stochastic feasible set reduced to be deterministic, dependent-chance programming usually reduced to chance constrained programming. This fact leads to that there is a common part between chance constrained programming and dependent-chance programming.

REFERENCES

1. J.K. Sengupta, *Stochastic Programming: Methods and Applications*, North-Holland, Amsterdam, (1972).
2. S. Vajda, *Probabilistic Programming*, Academic Press, New York, (1972).
3. P. Kall, *Stochastic Linear Programming*, Springer-Verlag, Berlin, (1976).
4. V.V. Kolbin, *Stochastic Programming*, D. Reidel, Dordrecht, (1977).
5. M.A.H. Dempster, Editor, *Stochastic Programming*, Academic Press, London, (1980).
6. Y. Ermoliev and R.J.-B. Wets, Editors, *Numerical Techniques for Stochastic Optimization*, Springer-Verlag, Berlin, (1988).
7. P. Kall and S.W. Wallace, *Stochastic Programming*, John Wiley & Sons, (1994).
8. A. Charnes and W.W. Cooper, Chance-constrained programming, *Management Science* **6** (1), 73–79, (1959).
9. B. Liu and C. Ku, Dependent-chance goal programming and an application, *Journal of Systems Engineering & Electronics* **4** (2), 40–47, (1993).
10. B. Liu, Dependent-chance goal programming and its genetic algorithm based approach, *Mathl. Comput. Modelling* **24** (7), 43–52, (1996).
11. B. Liu and K. Iwamura, Modelling stochastic decision systems using dependent-chance programming, *Europearn Journal of Operational Research* (to appear).
12. R.Y. Rubinstein, *Simulation and the Monte Carlo Method*, John Wiley & Sons, (1981).
13. R.E. Steuer, *Multiple Criteria Optimization: Theory, Computation and Application*, John Wiley & Sons, (1986).
14. D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, (1989).
15. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 2nd edition, Springer-Verlag, New York, (1994).

16. D.B. Fogel, An introduction to simulated evolutionary optimization, *IEEE Transactions on Neural Networks* **5**, 3–14, (1994).
17. K. Iwamura and B. Liu, A genetic algorithm for chance constrained programming, *Journal of Information & Optimization Sciences* **17** (2), 409–422, (1996).