JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 18, 563-580 (2002)

Applying Genetic Algorithms to Solve the Fuzzy Optimal Profit Problem

FENG-TSE LIN AND JING-SHING YAO*
Department of Applied Mathematics
Chinese Culture University
Taipei, 111 Taiwan
E-mail: ftlin@staff.pccu.edu.tw
**Department of Mathematics*
National Taiwan University
Taipei, 106 Taiwan

This study investigated the application of genetic algorithms for solving a fuzzy optimization problem that arises in business and economics. In this problem, a fuzzy price is determined using a linear or a quadratic fuzzy demand function as well as a linear cost function. The objective is to find the optimal fuzzy profit, which is derived from the fuzzy price and the fuzzy cost. The traditional methods for solving this problem are (1) using the extension principle, and (2) using the interval arithmetic and α-cuts. However, we argue that the traditional methods for solving this problem are too restrictive to produce an optimal solution, and that an alternative approach is possibly needed. We use genetic algorithms to obtain an approximate solution for this fuzzy optimal profit problem without using the membership functions. We not only give empirical examples to show the effectiveness of this approach, but also give theoretical proofs to validate the correctness of the algorithm. We conclude that genetic algorithms can produce good approximate solutions when applied to solve fuzzy optimization problems.

Keywords: genetic algorithms, fuzzy sets, fuzzy numbers, fuzzy optimization profit problem, fuzzy demand

1. INTRODUCTION

In this study, we investigated the application of genetic algorithms to solve a fuzzy optimization problem that arises in business and economics. This problem is often involved in the study of how changes in such variables as production or price will affect other variables such as revenue or profit [4, 20, 21]. One problem instance is the fuzzy optimal profit problem [20], which is briefly depicted as follows. In a monopolist market, producers can control market prices and product quantities. The demand x for a certain commodity is related to its price by a demand function $P(x)$. This means that, as the price increases, demand usually falls, and that as the price falls, demand rises. Since revenue is equal to the price per unit times the quantity sold, we can determine the revenue received for selling x units of the commodity as $R(x) = xP(x)$. The cost of producing x units of a certain commodity is given by a cost function $C(x)$ and the basic relation between profit, revenue and cost is formulated as $N(x) = R(x) - C(x)$. The monopolist can thus easily obtain the maximum profit by doing some simple calculations.

Received August 9, 2001; revised October 15, 2001; accepted January 9, 2002.
Communicated by Chuen-Tsai Sun.

On the other hand, in a perfect competitive market, the demand x is no longer a fixed value, even for the same price function $P(x)$. The price, of course, will fluctuate at any time in order to reflect various conditions in the market. To deal with this kind of imprecise data, fuzzy sets provide a powerful tool for modeling and solving the optimization problem. Thus, this leads to the use of the fuzzy number $\tilde{x}$ to represent the fuzzy demand and the fuzzy profit. The optimal price can then be obtained using the membership functions and the centroid of the fuzzy profit.

Nevertheless, the main issue in this problem is as follows. Consider a quadratic algebraic expression $ax^2 + bx$ for a, b real parameters and where x is a real variable. Let $y = N(a, b, x) = ax^2 + bx$. After substituting the triangular fuzzy numbers $\tilde{A}, \tilde{B}$, and $\tilde{X}$ into $ax^2 + bx$ for a, b, and x, respectively, we obtain the fuzzy set $\tilde{A}\tilde{X}^2 + \tilde{B}\tilde{X}$. The triangular fuzzy numbers are represented by $\tilde{A} = (a - \Delta_1, a, a + \Delta_2)$, $\tilde{B} = (b - w_1, b, b + w_2)$, and $\tilde{X} = (x - \delta_1, x, x + \delta_2)$ together with their membership functions $\mu(a \mid \tilde{A})$, $\mu(b \mid \tilde{B})$, and $\mu(x \mid \tilde{X})$, respectively. Accordingly, there are two major traditional methods for evaluating the above fuzzy expression in the literature [2, 3]. The first method for finding the value of $\tilde{A}\tilde{X}^2 + \tilde{B}\tilde{X}$ involves using the extension principle. Let $\Pi(a, b, x)$ denote the minimum of $\mu(a \mid \tilde{A})$, $\mu(b \mid \tilde{B})$, and $\mu(x \mid \tilde{X})$. Assume that we set $\tilde{Y}$ equal to $\tilde{A}\tilde{X}^2 + \tilde{B}\tilde{X}$. Then the membership function for $\tilde{Y}$ is defined by $\mu(y \mid \tilde{Y}) = \sup\{\Pi(a, b, x) \mid N(a, b, x) = y\}$. To find α-cuts of $\tilde{Y}$, let $\Phi(\alpha) = \{N(a, b, x) \mid a \in \tilde{A}(\alpha), b \in \tilde{B}(\alpha), x \in \tilde{X}(\alpha)\}$, $0 \le \alpha \le 1$. Then we have $\Phi(\alpha) = \tilde{Y}(\alpha)$. Alternatively, the second method involves evaluating $\tilde{A}\tilde{X}^2 + \tilde{B}\tilde{X}$ using interval arithmetic and α-cuts. For any triangular fuzzy numbers $\tilde{A}$ and $\tilde{B}$, the following operations hold:

(1) $(\tilde{A}\,\tilde{B})(\alpha) = \tilde{A}(\alpha)\,\tilde{B}(\alpha)$, and

(2) $(\tilde{A} \pm \tilde{B})(\alpha) = \tilde{A}(\alpha) \pm \tilde{B}(\alpha)$.

$\tilde{A}\,\tilde{B}$ and $\tilde{A} \pm \tilde{B}$ are computed using the extension principle, and $\tilde{A}(\alpha)\,\tilde{B}(\alpha)$, $\tilde{A}(\alpha) \pm \tilde{B}(\alpha)$ are found using interval arithmetic. Let $\tilde{Z}(\alpha) = \tilde{A}(\alpha)\,\tilde{X}(\alpha)\,\tilde{X}(\alpha) + \tilde{B}(\alpha)\,\tilde{X}(\alpha)$, for $0 \le \alpha \le 1$. We can see that $\tilde{Y}(\alpha)$ is a subset of $\tilde{Z}(\alpha)$. Obviously, evaluating a fuzzy algebraic expression using interval arithmetic and α-cuts can produce a larger fuzzy set than can evaluating using the extension principle. In addition, these traditional methods for solving fuzzy equations are very often too restrictive to produce a solution [3]. The other disadvantages, as noted in [7], are that solving fuzzy equations is very complicated due to the lack of inverse operators, and that the multiple occurrence of parameters in an expression results in increased imprecision.

In this study, we applied a genetic algorithm to obtain an approximate solution to the fuzzy optimal profit optimization problem. When genetic algorithms are applied to solve this problem, the fuzzy equation computation requires no extension principle or interval arithmetic and α-cuts. The genetic algorithm uses only the usual evolution. Assume that $\tilde{X}$ is an arbitrary fuzzy set on the interval $[0, \frac{a}{b}]$, and that $N(\tilde{X})$ is a fuzzy profit. In the genetic algorithm approach, we do not need to define the membership functions of $N(\tilde{X})$. Instead, the interval $[0, \frac{a}{b}]$ is equally divided into m partitions. Let $x_j = j\frac{a}{bm}$, $j = 0, 1, \ldots, m$ be the partition points. Let $\tilde{X}(x_j) = \mu_j \in [0,1]$,

j = 0, 1, …, m, be the membership grade of x_j in $\tilde{X}$. Thus, we obtain a discrete fuzzy set $\tilde{X} = (\mu_0, \mu_1, ..., \mu_m)$, where μ_j, j = 0, 1, …, m, is a random number in [0, 1]. In other words, we wish to find $\tilde{X}$ in $[0, \frac{a}{b}]$ by maximizing $N(\tilde{X})$ via the genetic algorithms. However, we cannot maximize $N(\tilde{X})$ directly since it is a fuzzy set. Instead, we can compute the centroid of the fuzzy profit for the maximization problem. The centroid can be used as the fitness value for the evolution of the genetic algorithms. Therefore, the objective is to simply find a vector $\tilde{X}$ in $[0, 1]^{m+1}$ that maximizes the centroid.

This paper is organized as follows. Section 2 deals with the optimal price for fuzzy profit based on the linear and quadratic demand functions. First, we discuss the traditional method for obtaining the optimal solution using fuzzy optimization and then introduce problem discretization for the purpose of the genetic algorithm application to obtain an approximate solution to this problem. In section 3, we design a genetic algorithm for solving the fuzzy optimal profit problem. Section 4 gives two illustrative examples. Section 5 discusses the main results of this work. Finally, we state our conclusions in section 6.

2. OPTIMAL PRICE FOR FUZZY PROFIT

2.1 Linear Demand Function

First, a linear demand function is introduced. Assume that the linear function is

$$P(x) = a - bx, 0 \le x \le a/b \tag{1}$$

and that the cost function is

$$C(x) = e + gx + kx^2, x \ge 0 \tag{2}$$

where a, b, e, g, k are known positive numbers, $b(a-g) < 2a(b+k)$, and $g < a$. The profit function is defined by

$$N(x) = xP(x) - C(x) = -e + (a-g)x - (b+k)x^2, \quad 0 \le x \le a/b. \tag{3}$$

In a monopolist market, the profit function is surely (3). Since $N'(x) = a - g - 2(b+k)x$, we have $x = \frac{a-g}{2(b+k)}$ $(= x^*)$. According to the assumption that $b(a-g) < 2a(b+k)$, we have $x^* \in$ $(0, \frac{a}{b}]$. Hence, the maximum profit is

$$N(x^*) = N\left(\frac{a-g}{2(b+k)}\right) = -e + \frac{(a-g)^2}{4(b+k)} \quad . \tag{4}$$

However, in a perfect competitive market, the demand x $\left(= \frac{a - P(x)}{b}\right)$ will vary, even for the same price $P(x)$. As a result, the demand x can be represented by the fuzzy number.

2.2 Quadratic Demand Function

Next, consider the quadratic demand function. Given the demand function

$$P(x) = a - bx + cx^2,\ 0 \le x \le x_*, \tag{5}$$

and the cost function $C(x) = e + gx + kx^2,\ x \ge 0,$ (6)

where all variables *a, b, c, d, e, g,* and *k* are known positive numbers, the relations of this quadratic function are

(1) If $b^2 - 4ac \ge 0$, then set $x_* = \dfrac{b - \sqrt{b^2 - 4ac}}{2c}$.

(2) If $b^2 - 4ac < 0$, then set $x_* = -\dfrac{b}{2c}$.

From (5) and (6), we have the profit function as follows:

$$N(x) = -e + (a-g)x - (b+k)x^2 + cx^3,\ 0 \le x \le x_*. \tag{7}$$

Consider a monopolist market. By differentiating (7), we obtain the following equation:

$N'(x) = (a-g) - 2(b+k)x + 3cx^2,\quad 0 \le x \le x_*$. The maximum is found to be

$$3cx^2 - 2(b+k)x + (a-g) = 0. \tag{8}$$

The discriminant for (8) is $D_1 = (b+k)^2 - 3c(a-g)$.

Case 1. If $D_1 > 0$, then (8) has two roots, $d_1 = \dfrac{b + k - \sqrt{D_1}}{3c}$ and $d_2 = \dfrac{b + k + \sqrt{D_1}}{3c}$.
We obtain $N'(x) = 3c(x - d_1)(x - d_2)$. Then, we have the following three possibilities:

(1) When $0 < d_1 < d_2 < x_*$, $\max\limits_{0 \le x \le x_*} N(x) = \max(N(d_1), N(x_*))$. (9)

(2) When $0 < d_1 < x_* < d_2$, $\max\limits_{0 \le x \le x_*} N(x) = N(d_1)$. (10)

(3) When $x_* < d_1 < d_2$, $\max\limits_{0 \le x \le x_*} N(x) = N(x_*)$. (11)

Case 2. If $D_1 = 0$, then $N'(x) = 3c(x - \dfrac{b+k}{3c})^2 > 0$ and $\max\limits_{0 \le x \le x_*} N(x) = N(x_*)$. (12)

Case 3. If $D_1 < 0$, then $N'(x) > 0\ \forall\ 0 < x < x_*$ and $\max\limits_{0 \le x \le x_*} N(x) = N(x_*)$. (13)

Assume that the optimal demand is x^{**}. Then the maximum profit is $N(x^{**})$. Note that in a monopolist market, the demand *x* is uniquely determined by the given price *P*(*x*). However, in a perfect competitive market, the demand and the price do not necessary depend on the demand function. Therefore, the demand $x \left(= \dfrac{a - P(x)}{b} \right)$ will vary, even for the same price *P*(*x*). This is a completely different interpretation of the demand *x* in a competitive market compared with that in a monopolist market. As a result, the demand should be represented using linguistic variables, such as “if the price is $P(x)$, then the demand is in the vicinity of x,” to obtain a reasonable description.

2.3 Traditional Fuzzy Methods

Obviously, the demand in a competitive market can be represented using the triangular fuzzy number $\tilde{x}$. From (7), we obtain the fuzzy profit $N(\tilde{x}) = -e + (a - g)\tilde{x} - (b + k)\tilde{x}^2$. Let $N(x) = y$. Then (3) becomes $(b + k)x^2 - (a - g)x + e + y = 0$. (14)
From (14), we can see that $D(y) = (a-g)^2 - 4(b+k)(e+y) \geq 0$, if $0 \leq y \leq N(x^*)$. If $D(y) > 0$, then (14) has two roots, which are quadratic formulas, shown as follows:

$$r_1(y) = \frac{a-g-\sqrt{D(y)}}{2(b+k)}, \quad r_2(y) = \frac{a-g+\sqrt{D(y)}}{2(b+k)}. \tag{15}$$

Otherwise, if $D(y) = 0$, (14) has a duplicate root, then

$$r_1(y) = r_2(y) = \frac{a-g}{2(b+k)}. \tag{16}$$

After applying the extension principle method, we obtain the membership function of the fuzzy profit $N(\tilde{x})$ as

$$N(\tilde{x})(y) = \sup_{y=N(x)} \tilde{x}(x) = \begin{Bmatrix} \max[\tilde{x}(r_1(y)), \tilde{x}(r_2(y))], \mathit{if}\ 0 \leq y \leq N(x^*) \\ 0 \qquad , \text{otherwise} \end{Bmatrix}, \tag{17}$$

where the membership function of the triangular fuzzy number $\tilde{x}$ is $\tilde{x}(x)$ and the membership function of the fuzzy profit $N(\tilde{x})$ is $N(\tilde{x})(y)$. Once we have $N(\tilde{x})(y)$, the centroid of $N(\tilde{x})$, $E(x)$, can be obtained using the following equation:

$$E(x) = \frac{\int_0^{N(x^*)} yN(\tilde{x})(y)dy}{\int_0^{N(x^*)} N(\tilde{x})(y)dy}. \tag{18}$$

Then the best solution of the fuzzy optimal profit problem is obtained.

Similarly, consider the quadratic function. After we apply the extension principle, the membership function of $N(\tilde{x})$ is $\mu_{N(\tilde{x})}(y) = \sup_{x \in N^{-1}(y)} \mu_{\tilde{x}}(x)$ if $N^{-1}(y) \neq \varnothing$. Note that $N(x) = y$ is a cubic equation. Clearly, producing $N^{-1}(y)$ for a cubic equation is not an easy task. (see section 5.3.1 for a detailed discussion).

2.4 Genetic Algorithms Approach

The genetic algorithms approach is depicted as follows. Let the triangular fuzzy number $\tilde{x} = (x - \Delta_1, x, x + \Delta_2)$ be replaced by an arbitrary fuzzy set $\tilde{X}$ in an interval $[0, x_*]$, where x_* is defined in (5) (see Fig. 1).

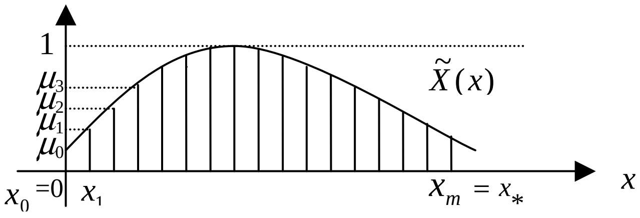


Fig. 1. Fuzzy set $\tilde{X}$.

The interval $[0, x_*]$ is divided into m partitions, $x_j = j\frac{x_*}{m}$, where $j = 0, 1, \ldots, m$ are the partition points. Let the membership grade of $\tilde{X}$ at x_j be $\tilde{X}(x_j) = \mu_j$, $j = 0, 1, \ldots, m$ and $\mu_j \in [0,1]$. Then, a discrete fuzzy set is obtained as follows:

$$\tilde{X} = (\mu_0, \mu_1, \ldots, \mu_m) = \frac{\mu_0}{x_0} + \frac{\mu_1}{x_1} + \ldots + \frac{\mu_m}{x_m}. \tag{19}$$

From $N(x) = y$, if each $N(x_k)$, $k = 0, 1, \ldots, m$, is different, then the fuzzy profit is defined by $N(\tilde{X}) = N(\mu_0, \mu_1, \ldots, \mu_m) = \frac{\mu_0}{N(x_0)} + \frac{\mu_1}{N(x_1)} + \ldots + \frac{\mu_m}{N(x_m)}$, (20)

and the centroid is defined by $\theta(N(\tilde{X})) = \frac{\sum_{j=0}^{m} N(x_j)\mu_j}{\sum_{j=0}^{m} \mu_j}$. (21)

From (7), let $N(x) = y$; we obtain a cubic equation:

$$cx^3 - (b+k)x^2 + (a-g)x - (e+y) = 0.$$

Assume that there are three roots for this equation. Let $N(x_{j_1}) = N(x_{j_2}) = N(x_{j_3}) = y_0$, where $j_1 \neq j_2 \neq j_3$. According to the extension principle, we obtain $N(\mu_0, \mu_1, \ldots, \mu_N)(y_0)$ $= \sup_{y_0 = N(x)} \tilde{X}(x) = \max(\tilde{X}(x_{j_1}), \tilde{X}(x_{j_2}), \tilde{X}(x_{j_3})) = \max[\mu_{j_1}, \mu_{j_2}, \mu_{j_3}]$. Eq. (20) becomes $\frac{\max[\mu_{j_1}, \mu_{j_2}, \mu_{j_3}]}{N(x_{j_1})}$. The centroid of $N(\tilde{X})$ is $\theta(N(\tilde{X})) = \frac{Q}{P}$, (22)

where $P = \sum_{j \neq j1, j2, j3} \mu_j + \max(\mu_{j1}, \mu_{j2}, \mu_{j3})$ and

$$Q = \sum_{j \neq j1, j2, j3} N(x_j)\mu_j + N(x_{j1})(\max[\mu_{j1}, \mu_{j2}, \mu_{j3}]).$$

Let the centroid of the fuzzy profit $N(\tilde{X}) = N(\mu_0, \mu_1, \ldots, \mu_m)$ be

$$\theta(N(\tilde{X})) = centroid\,of\; N(\mu_0, \mu_1, \ldots, \mu_m).$$

Finally, a vector $\tilde{X}$ is determined in $[0, 1]^{m+1}$ (real numbers) in a population that maximizes $\theta(N(\tilde{X}))$ by means of genetic algorithms.

3. DESIGNING A GENETIC ALGORITHM FOR SOLVING THE FUZZY OPTIMAL PROFIT PROBLEM

Genetic algorithms are stochastic search techniques based on the principles and mechanisms of natural genetics and selection [8, 15]. Genetic algorithms grew out of Holland's [11] study on adaptation in artificial and natural systems. The basic concept of genetic algorithms is that they start with a population of randomly generated candidates and evolve towards better solutions by applying genetic operators, such as crossover and mutation, modeled on natural genetic inheritance and Darwinian survival-of-the-fittest [8, 15]. Therefore, genetic algorithms are theoretically and empirically have been proven to process robust search capabilities in complex spaces, thus offering a valid approach to problems requiring efficient and effective searching [6].

Genetic algorithms can be used to compute the membership functions of fuzzy sets [12, 13]. Given some functional mapping for a system, some membership functions and their shapes are assumed for the various fuzzy variables defined for a problem [16]. The membership functions are coded as bit strings that are then concatenated. An evaluation function is used to evaluate the fitness of each set of membership functions. There are two possible ways to integrate fuzzy logic and genetic algorithms [9]. One involves the application of genetic algorithms for solving optimization and search problems related to fuzzy systems [1, 10, 17]. The other, is the use of fuzzy tools and fuzzy logic-based techniques for modeling different genetic algorithm components and adapting genetic algorithm control parameters, with the goal of improving performance [9, 14, 18, 19]. Now, a genetic algorithm for solving the fuzzy optimal profit problem is given below.

Step1. Generate an initial population.

An initial population of size *n* is randomly generated from $[0, 1]^{m+1}$ according to the uniform distribution in the closed interval [0,1]. Let the population be

$$\tilde{X}_j = (\mu_{j_0}, \mu_{j_1}, \ldots, \mu_{j_m}) = \frac{\mu_{j_0}}{x_0} + \frac{\mu_{j_1}}{x_1} + \ldots + \frac{\mu_{j_m}}{x_m},$$

where $j = 1, 2, \ldots, n$ and μ_{j_i} is a real number in [0, 1], $i = 0, 1, 2, \ldots, m$. Each individual $\tilde{X}_j$ in a population is a chromosome.

Step 2. Calculate the fitness value for each chromosome.

For each chromosome $\tilde{X}_j$, $j = 1, 2, \ldots, n$, the centroid $\theta(N(\tilde{X}_j))$ is calculated as the fitness value. The chromosomes in the population can be rated in terms of their fitness values. Let the total fitness value of the population be $T = \sum_{j=1}^{n} \theta(N(\tilde{X}_j))$. The cumulative fitness value (partial sum) for each chromosome, $S_k = \sum_{j=1}^{k} \theta(N(\tilde{X}_j))$, $k = 1, 2, \ldots, n$, is calculated. Next, intervals $I_1 = [0, S_1]$, $I_j = [S_{j-1}, S_j]$, $j = 2, 3, \ldots, n-1$, and $I_n = [S_{n-1}, S_n]$

are constructed for the purpose of selection. In our implementation, a roulette wheel approach [6, 8] is adopted as the selection mechanism.

Step 3. Selection and reproduction.

Reproduction is a process in which each chromosome is copied according to the selection process. The selection process begins with spinning of the roulette wheel n times. Each time, a single chromosome is selected from the current generation to create a new generation. The selection process is as follows. Each time, a random number r from the range $[0, T]$ is generated. If $r \in I_1$, then chromosome $\tilde{X}_1$ is selected; otherwise, the kth chromosome $\tilde{X}_k, 2 \le k \le n$, is selected if $r \in I_k$. This selection process is continued until the new population has been created. Finally, the new population is renamed $\tilde{Y}_1, \tilde{Y}_2, \tilde{Y}_3, \ldots$ in the order they were picked. The probability of selection for each chromosome $\tilde{X}_j$ such that it appears in the new population is $\frac{\theta(N(\tilde{X}_j))}{T}$, and its expected value is $n\frac{\theta(N(\tilde{X}_j))}{T}$. Thus, this procedure tends to choose more $\tilde{X}_j$ with higher $\theta(N(\tilde{X}_j))$ to go on into the next population.

Step 4. Perform crossover.

Crossover is the key to genetic algorithms' power. The purpose of crossover is to generate rearrangements of coadapted groups of information from high performance structures [8]. The crossover method used here is the one-point method, which randomly selects one cut-point and exchanges the right parts of two parents to generate offspring. Each pair $(\tilde{Y}_1, \tilde{Y}_2)$, $(\tilde{Y}_3, \tilde{Y}_4)$, …, $(\tilde{Y}_{n-1}, \tilde{Y}_n)$, where n is even, produces two children via crossover. Let p be the probability of a crossover, $0 \le p \le 1$. Usually, p is between 0.6 and 0.9 [8], so we expect that, on average, 60% to 90% of the chromosomes will undergo crossover. Consider two chromosomes, $\tilde{Y}_1$ and $\tilde{Y}_2$, for crossover. A random number r is generated in the interval [0, 1]. If $r \le p$, then crossover is performed on $\tilde{Y}_1$ and $\tilde{Y}_2$. Otherwise, if $r > p$, the two children $\tilde{Y}_1'$ and $\tilde{Y}_2'$ are identical to their parents, $\tilde{Y}_1$ and $\tilde{Y}_2$. Suppose that crossover needs to be performed. Another random integer number u is generated from the range $[0, m - 1]$. Assume that u equals 3; the two chromosomes $\tilde{Y}_1$ and $\tilde{Y}_2$ are cut after the fourth position, and the offspring $\tilde{Y}_1'$ and $\tilde{Y}_2'$ are generated by exchanging the right parts of each chromosome.

Step 5. Perform mutation.

Mutation is a background operator that produces random changes in various chromosomes [6]. Mutation resets one randomly selected position to a real number in [0, 1] or to zero with a probability equal to the mutation rate (see Examples in section 4). Let q be the probability of a mutation, $0 \le q \le 1$. Usually q is a very small value, around 0.001 to 0.01 [8], so we expect that, on average, 0.1% to 1% of the total population will undergo mutation. There are $n \times m$ positions in the whole population. We expect $0.001\, n \times m$ to $0.01\, n \times m$ mutations per generation. For example, consider chromosome $\tilde{Y}_1'$. Let w_i be a random number in [0, 1], $0 \le i \le m$. If $w_i < q$, then the i+1th position in $\tilde{Y}_1'$ is reset to a real random number in [0, 1] or to zero. After the mutation is completed on the whole population, we let $\tilde{X}_j = \tilde{Y}_j'$, $j = 1, 2, \ldots, n$. One iteration of the genetic algorithm has been completed. Step 2 through step 5 is done *K* times, where *K* is the maximum number of iterations.

Finally, the algorithm is terminated after *K* generations are produced. Let the last population be $\tilde{X}_1^*, \tilde{X}_2^*, ..., \tilde{X}_n^*$. The centroid of each chromosome $\tilde{X}_j^*$, $1 \le j \le n$, in the last population must now be calculated. The maximum value of $\theta(N(\tilde{X}_k^*))$ is the best chromosome in the population, i.e., $\max\limits_{1 \le j \le n} \theta(N(\tilde{X}_j^*)) = \theta(N(\tilde{X}_k^*))$ for some $k \in \{1, 2, ..., n\}$. The best chromosome could be a sub-optimal or the optimal solution for the problem. Let the best chromosome be $\tilde{X}_k^* = (\mu_{k0}^*, \mu_{k1}^*, ..., \mu_{km}^*) = \frac{\mu_{k0}^*}{x_0} + \frac{\mu_{k1}^*}{x_1} + \ ... \ + \frac{\mu_{km}^*}{x_m}$. If $P(x_i) = a - bx_i + cx_i^2$, for each *i*, where *i* = 0, 1, …, *m*, are different values, then the fuzzy price function is

$$P(\tilde{X}_k^*) = a - b\tilde{X}_k^* + c\tilde{X}_k^{*2} = \frac{\mu_{k0}^*}{a - bx_0 + cx_0^2} + \frac{\mu_{k1}^*}{a - bx_1 + cx_1^2} + ... + \frac{\mu_{km}^*}{a - bx_m + cx_m^2}. \tag{23}$$

Therefore, the centroid of $P(\tilde{X}_k^*)$ is

$$E(P(\tilde{X}_k^*)) = \frac{\sum_{j=0}^{m} (a - bx_j + cx_j^2)\mu_{kj}^*}{\sum_{j=0}^{m} \mu_{kj}^*}. \tag{24}$$

This is an estimate of the maximal profit in the fuzzy sense. In particular, if

$P(x_i) = P(x_j), i \ne j$, then $\frac{\mu_{k,i}^*}{P(x_i)} + \frac{\mu_{k,j}^*}{P(x_j)}$ of (23) will become $\frac{\max(\mu_{k,i}^*, \mu_{k,j}^*)}{P(x_i)}$.

In the following, two properties are given to show how good the final solution obtained by genetic algorithms is, and it is compared with the known crisp optimal value.

Property 1. The optimal value $\theta(N(\tilde{X}_k^*))$ obtained using genetic algorithms is less than or equal to the crisp optimal value $N(x^{**})$. That is, $\theta(N(\tilde{X}_k^*)) \le N(x^{**})$.

Proof. For any discrete fuzzy set, $\tilde{X} = (\mu_0, \mu_1, ..., \mu_m) = \frac{\mu_0}{x_0} + \frac{\mu_1}{x_1} + ... + \frac{\mu_m}{x_m}$, where $x_j \in [0, x_*]$, $\mu_j \in [0,1]$, *j* = 0, 1, … *m*, its fuzzy profit is $N(\tilde{X}) = N(\mu_0, \mu_1, ..., \mu_m)$. When $N(x_k)$, *k* = 0, 1, …, *m*, are different values, the centroid is obtained by $\theta(N(\tilde{X})) = \frac{\sum_{j=0}^{m} N(x_j)\mu_j}{\sum_{j=0}^{m} \mu_j}$. Note that $N(x^{**})$ is the crisp optimal solution (i.e. maximum profit). For each $x_j \in [0, x_*]$, *j* = 1, 2, …, *m*, we have $N(x_j) \le N(x^{**})$, $0 \le j \le m$.

Since $0 \le \mu_j \le 1$ and $0 \le j \le m$, we obtain $\sum_{j=0}^{m} N(x_j)\mu_j \le N(x^{**})\sum_{j=0}^{m}\mu_j$. Therefore, $\theta(N(\tilde{X})) \le N(x^{**})$. Furthermore, for each $N(x_j)$, j = 0, 1, …, m, if there exist three identical values, e.g. $N(x_{j1}) = N(x_{j2}) = N(x_{j3})$, $j_1 \ne j_2 \ne j_3$, then from (22), the centroid of $N(\tilde{X})$ is $\theta(N(\tilde{X})) = \frac{Q}{P}$, where $P = \sum_{j \ne j1, j2, j3} \mu_j + \max(\mu_{j1}, \mu_{j2}, \mu_{j3})$ and $Q = \sum_{j \ne j1, j2, j3} N(x_j)\mu_j + N(x_{j1})(\max[\mu_{j1}, \mu_{j2}, \mu_{j3}])$. We can see that $\theta(N(\tilde{X})) \le N(x^{**})$. Therefore, we obtain $\theta(N(\tilde{X}_k^*)) \le N(x^{**})$. **Q.E.D**

Property 2. The search space for genetic algorithms includes the crisp optimal profit solution. However, the probability that the value of $\theta(N(\tilde{X}^*))$ is equal to the crisp value $N(x^{**})$ is small.

Proof. The crisp optimal demand x^{**} is in the interval [0, x_*]. When the interval [0, x_*] is divided into m partitions, $x_j = j\frac{x_*}{bm}$, j = 0, 1, … , m, we have the possibility that one of the partition includes x^{**}. Let x_i be x^{**}, $i \in \{0, 1, \dots, m\}$. From (19), we have $\tilde{X} = \frac{\mu_0}{x_0} + \frac{\mu_1}{x_1} + \dots + \frac{\mu_m}{x_m}$, where μ_j is a random number in [0, 1], j = 0, 1, … , m. We can see that there is a possibility of having $\mu_j = 0$ for $\forall j \ne i$ and $\mu_i \ne 0$. Thus, we have $\tilde{X} = \frac{\mu_i}{x_i} = \frac{\mu_i}{x^{**}}$. From (21), we obtain $N(0, \dots 0, \mu_i, 0, \dots, 0) = \frac{\mu_i}{N(x_i)} = \frac{\mu_i}{N(x^{**})}$. Then, the centroid $\theta(N(0,\dots,0,\mu_i,0,\dots,0) = N(x_i)$ is equal to $N(x^{**})$. Obviously, the genetic algorithm search space theoretically includes the crisp optimal profit solution. However, the probability that the value of $\theta(N(\tilde{X}^*))$ equals the crisp value $N(x^{**})$ is small. **Q.E.D.**

4. EMPIRICAL RESULTS

Two empirical examples are given below to illustrate the effectiveness of genetic algorithms when we applying them to solve the fuzzy optimal profit problem.

Example 1. The first instance is as follows: the price function, $P(x) = 100 - 2x$, $0 \le x \le 50$; the cost function, $C(x) = 10 + x$; and the profit function, $N(x) = xP(x) - C(x) = -10 + 99x - 2x^2$, $0 \le x \le 50$. Since $N'(x) = 99 - 4x = 0$, we obtain $x = 24.75$. Thus, the optimal solutions for the crisp case are $P(24.75) = 50.5$ and $N(24.75) = 1215.125$. Now, consider that the demand x is vague. Let $m = 10$. We have $x_k = k\frac{50}{10} = 5k$, k = 0, 1, …, 10, which means that ten partition points are given. Then, we obtain $N(x_0) = -10$, $N(x_1)$

= 435, $N(x_2) = 780$, $N(x_3) = 1025$, $N(x_4) = 1170$, $N(x_5) = 1215$, $N(x_6) = 1160$, $N(x_7) = 1005$, $N(x_8) = 750$, $N(x_9) = 395$, and $N(x_{10}) = -60$. From (20) and (21), we have $N(\tilde{X}) = N(\mu_0, \mu_1, \ldots, \mu_{10})$ and the centroid $\theta(N(\tilde{X})) = \dfrac{\sum_{j=0}^{10} N(x_j)\mu_j}{\sum_{j=0}^{10} \mu_j}$. The estimated price in the fuzzy sense is given by $E(P(\tilde{X})) = \dfrac{\sum_{j=0}^{10} P(x_j)\mu_j}{\sum_{j=0}^{10} \mu_j}$. The parameters for genetic algorithms are (1) the probability of a crossover $p = 0.8$ and (2) the probability of a mutation $q = 0.003$. An essential feature of genetic algorithms when they are used to solve the fuzzy optimal profit problem is their effectiveness in solving the fuzzy problem. Two different approaches were taken in this study for the purpose of analyzing the effectiveness of the algorithm. The first approach was based on using the same number of generations but with different population sizes. The second approach was based on using the same population size but with different numbers of generations. Each approach had five or six runs. In the first approach, the number of generations was 10000, and the population size for each run was 100, 200, 300, 400, and 500, respectively. The best results for $\tilde{X}$ obtained in five runs are shown in Fig. 2. The partition points x_j are 5, 10, 15, 20, 25, …, etc. We can see that the genetic algorithm was trying to approximate $\tilde{X}(x_j) = 1.0$ for x_j = 20, 25, and 30, and $\tilde{X}(x_j) = 0.0$ otherwise, where a bell-shaped discrete fuzzy set was obtained. In these runs, one of the best solutions obtained was $\tilde{X}$ = (0.04, 0.02, 0.03, 0.21, 0.92, 0.98, 0.87, 0.06, 0.04, 0.03, 0.02) with the largest value being $\theta(N(\tilde{X}))$ = 1193.882 and the estimated price in the fuzzy sense being $E(P(\tilde{X}))$ = 50.64. A comparison of the above obtained result with that of the crisp case is $\dfrac{1193.882 - 1215.125}{1215.125} = -1.7482\%$ and $\dfrac{50.64 - 50.5}{50.5} = 0.2737\%$.

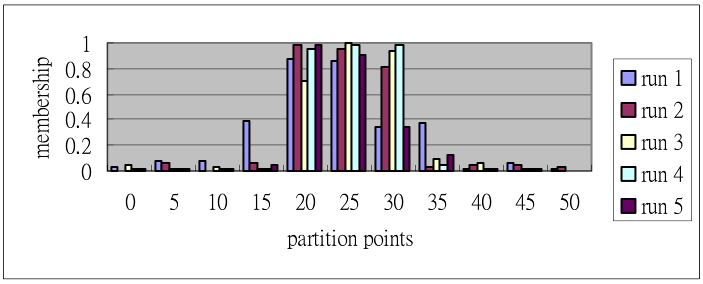


Fig. 2. A bell-shaped discrete fuzzy set obtained using the first approach. The number of generations is 10000, and the population size in each run is 100, 200, 300, 400, and 500, respectively.

As for the second approach, the population size was 200 in all six runs. The number of generations in each run was 10,000, 12,000, 14,000, 16,000, 18,000, and

20,000, respectively. The best results for $\tilde{X}$ obtained in six runs are shown in Fig. 3. Once again, the genetic algorithm was trying to approximate $\tilde{X}(x_j) = 1.0$ for x_j = 20, 25, and 30, and $\tilde{X}(x_j) = 0.0$ otherwise, in these runs. Note that the curve for "run 6" shown in Fig. 3 has a bell-shaped discrete fuzzy set with $\tilde{X}(20) =$ 0.86, $\tilde{X}(25) =$ 0.92, and $\tilde{X}(30) = 0.58$.

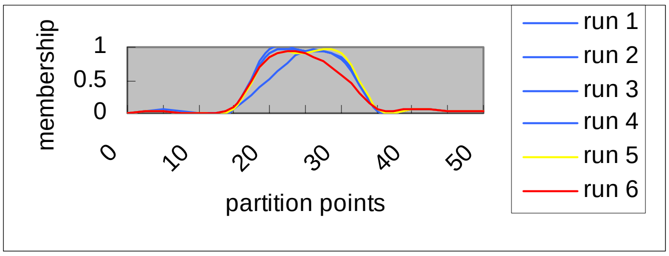


Fig. 3. A bell-shaped discrete fuzzy set was obtained from "run 6", which had a population size of 200 and 20,000 generations.

If we let the mutation reset one randomly selected position to zero (i.e. force the membership grade to 0), the best result obtained in each run became $\tilde{X}$ = (0.00, 0.00, 0.00, 0.00, 0.00, 0.80, 0.00, …, 0.00). In this case, the maximum profit obtained by genetic algorithms was the discrete fuzzy set $\tilde{X}_k^*$ = $\frac{0.8}{25}$ (where the membership grade was 0.8 when x was 25) with an optimal value of 1215 and an estimated price of 50. Since the partition points $x_k = k\frac{50}{10} = 5k$, k = 0, 1, …, 10 did not include the optimal demand x^* = 24.75 (the nearest point was x_5 = 25.0), the best profit obtained by the genetic algorithms was only 1215 (the optimal profit in the crisp case was 1215.125).

Finally, consider the situation when more partition points are given. Let m = 15. The number of partition points was 15. Fig. 4 shows the membership curves obtained from six runs based on using the same population size but different numbers of generations. The number of generations in each run was 10,000, 12,000, 14,000, 16,000, 18,000, and 20,000, respectively. All these curves are trying to approximate $\tilde{X}(x_j) = 1.0$ for x_j = 23.3 and 26.7. Note that the curve from "run 6" has a bell-shaped discrete fuzzy set with $\tilde{X}(20)$ = 0.64, $\tilde{X}(23.33)$ = 0.86, $\tilde{X}(26.67) =$ 0.85, and $\tilde{X}(30) =$ 0.76.

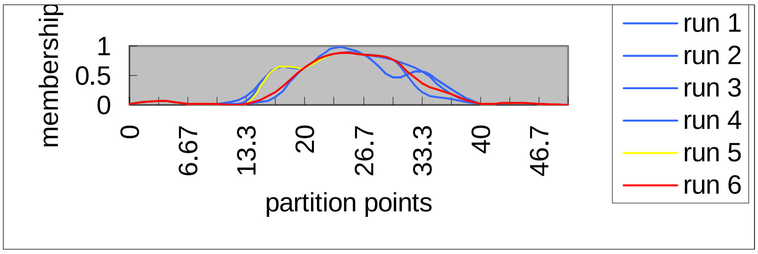


Fig. 4. The bell-shaped discrete fuzzy set obtained from "run 6" using the second approach with 15 partition points.

Example 2. The problem instance is as follows: the price function is a quadratic, $P(x) = 101 - 10x + x^2$, $0 \le x \le 5$; the cost function is $C(x) = 10 + x$, $x \ge 0$; and the profit function is $N(x) = -10 + 100x - 10x^2 + x^3$, $0 \le x \le 5$. Since $b^2 - 4ac = 100 - 404 < 0$, we have $x = 10/2 = 5$ and $N'(x) = 100 - 20x + 2x^2 = 2(x-5)^2 + 50 > 0$. In the crisp case of the problem, when $x = 5$, the optimal solutions are $P(5) = 76$ and $N(5) = 365$. Now, consider that x is vague. Let $m = 10$; we have $x_k = k\frac{5}{10} = 0.5k$, $k = 0, 1, \ldots, 10$. We obtain $N(x_0) = -10$, $N(x_1) = 37.625$, $N(x_2) = 81$, $N(x_3) = 120.875$, $N(x_4) = 158$, $N(x_5) = 193.125$, $N(x_6) = 227$, $N(x_7) = 260.375$, $N(x_8) = 294$, $N(x_9) = 328.625$, and $N(x_{10}) = 365$. The centroid is $\theta(N(\tilde{X})) = \frac{\sum_{j=0}^{10} N(x_j)\mu_j}{\sum_{j=0}^{10} \mu_j}$, and the optimal price in the fuzzy sense is $E(P(\tilde{X})) = \frac{\sum_{j=0}^{10} P(x_j)\mu_j}{\sum_{j=0}^{10} \mu_j}$. The parameters and the two approaches used for genetic algorithms were all the same as in Example 1. The best results for $\tilde{X}$ obtained in five runs using the first approach with the same number of generations but with different population sizes are shown in Fig. 5. The partition points x_j are 0.5, 1.0, 1.5, 2.0, 2.5, …, etc. We can see that the genetic algorithm was trying to approximate $\tilde{X}(x_j) = 1.0$ for $x_j = 4.5$, 5.0 and $\tilde{X}(x_j) = 0.0$ otherwise, where a half bell-shaped discrete fuzzy set was obtained. In these runs, one of the best solutions obtained was $\tilde{X} = (0.001, 0.004, 0.007, 0.058, 0.038, 0.052, 0.062, 0.112, 0.070, 0.849, 0.982)$ with the largest value being $\theta(N(\tilde{X})) = 359.367$ and the estimated price in the fuzzy sense being $E(P(\tilde{X})) = 76.808$. A comparison of the results obtained with that of the crisp case is

$$\frac{359.367 - 365.0}{365.0} \times 100\% = -1.5432\%, \text{ and } \frac{76.808 - 76.0}{76.0} \times 100\% = 1.0631\%.$$

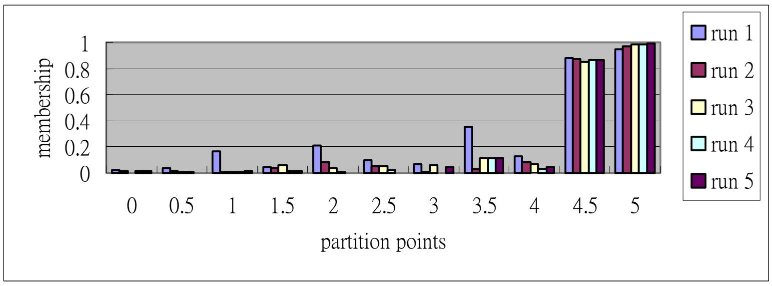


Fig. 5. A half bell-shaped discrete fuzzy set obtained using the first approach.

For the second approach, using the same population size but with different numbers of generations, the best results for $\tilde{X}$ obtained from six runs are shown in Fig. 6.

From Fig. 6, we can see that the genetic algorithm was trying to approximate $\tilde{X}(x_j) = 1.0$ for x_j = 4.5, 5.0 and $\tilde{X}(x_j) = 0.0$ otherwise, where a half bell-shaped discrete fuzzy set was again obtained. Once again, if we let the mutation reset one randomly selected position to zero, the best result obtained in each run became $\tilde{X}$ = (0.00, 0.00, 0.00, …, 0.00, 0.00, 0.915). In this case, the maximum profit obtained by the genetic algorithms was the discrete fuzzy set $\tilde{X}_k^*$ = $\frac{0.915}{5.0}$ (the membership grade was 0.915 when x was 5.0) with a value of 365.0.

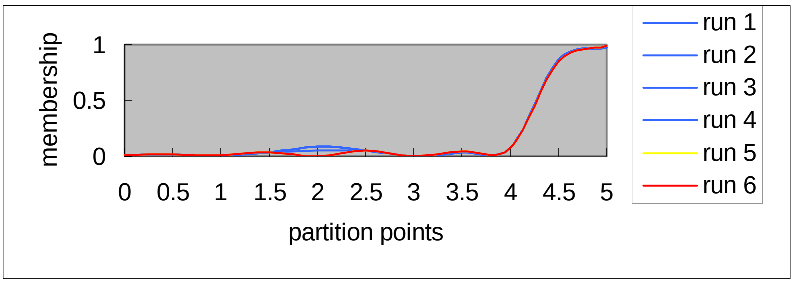


Fig. 6. A half bell-shaped discrete fuzzy set obtained using the second approach.

5. DISCUSSION

In this section, we will point out that our work has produced the following main results based on the application of genetic algorithms to solve the fuzzy optimal profit problem.

5.1 Genetic Algorithms Can Find the Optimal Solution

As explained in Section 2, the crisp optimal demand x^{**}, was in $(0, x_*]$, and we can take m as a suitable value to divide the interval $(0, x_*]$ into small partitions, i.e. $x_j = j\frac{x_*}{bm}$, j = 0, 1, …, m. According to Properties 1 and 2, let x_i = x^{**} for some i $\in \{0, 1, 2, \ldots, m\}$. From (19), the random numbers $\mu_1, \mu_2, \ldots, \mu_m$, generated from the interval [0, 1], should have a possibility of $\mu_j = 0, \forall j \neq i$ and $\mu_i \neq 0$. Therefore, (19) becomes $\tilde{X}$ = $(0,\ldots,0,\mu_i,0,\ldots,0)$ = $\frac{\mu_i}{x_i}$ = $\frac{\mu_i}{x^{**}}$. According to (20), we have $N(0,\ldots,0,\mu_i,0,\ldots,0)$ = $\frac{\mu_i}{N(x^{**})}$, where $N(x^{**})$ is the crisp maximum profit. We conclude that the search space for genetic algorithms includes the crisp optimal solution and crisp near-optimal solutions.

5.2 Mutation Strategies

In our implementation, two strategies are used to perform mutation in genetic algorithms. Normally, we use Strategy 1. Strategy 1 resets one randomly selected position to a real number in [0, 1] (sets the membership grade to the other value). On the other hand, Strategy 2 resets one randomly selected position in a chromosome to zero (forces the membership grade to zero). Example 1 in Section 4 shows that Strategy 2 had in more likely than Strategy 1 to include the crisp optimal solution in its search space. Empirical results show that the speed of convergence to the crisp optimal solution using Strategy 2 was much faster than that achieved using Strategy 1. The underlying theorem for Strategy 2 is based on Property 1.

5.3 Comparison Between the Traditional Fuzzy Method and Genetic Algorithms

The major difference in obtaining the fuzzy optimal profit between using the traditional fuzzy method [20, 21] and genetic algorithms is stated as follows.

5.3.1 Using the extension principle method

As explained in Section 2, when the demand is fuzzified into a triangular fuzzy number $\tilde{x} = (x-\Delta_1, x, x+\Delta_2)$, $0 < \Delta_1 < x, \Delta_2 > 0$, obviously, the fuzzy profit is $N(\tilde{x})$. By applying the extension principle method, we can obtain the membership function of $N(\tilde{x})$, $\mu_{N(\tilde{x})}(y) = \sup_{x \in N^{-1}(y)} \mu_{\tilde{x}}(x)$. Since $N(x) = y$ is a cubic function, i.e., $-e + (a - g)x - (b + k)\, x^2 + cx^3$, it is difficult to obtain $x = N^{-1}(y)$. As a result, the membership function $\mu_{N(\tilde{x})}(y)$ is difficult to obtain by applying the extension principle. Furthermore, if we let the three roots of $N(x) = y$ be $t_1(y)$, $t_2(y)$, and $t_3(y)$, then we can see that $\mu_{N(\tilde{x})}(y) = \sup \min[\mu_{\tilde{x}}(t_1(y)),\ \mu_{\tilde{x}}(t_2(y)),\ \mu_{\tilde{x}}(t_3(y))]$, where

$$\mu_{\tilde{x}}(t) = \begin{cases} \dfrac{t - x + \Delta_1}{\Delta_1}, & x - \Delta_1 \le t \le x \\ \dfrac{x + \Delta_2 - t}{\Delta_2}, & x \le t \le x + \Delta_2 \\ 0, & \text{otherwise} \end{cases}$$

Obviously, it is also difficult to obtain $\mu_{N(\tilde{x})}(y)$ from the above equation. In this case, the proposed genetic algorithm approach can be considered as an alternative way to find a near optimal solution.

5.3.2 Using genetic algorithms

With the genetic algorithm approach, however, no specific fuzzy set is needed (i.e. use of a triangular fuzzy number or other special fuzzy sets is not necessary). Initially, a population of discrete fuzzy sets, $\tilde{X}_j = \frac{\mu_0}{x_0} + \frac{\mu_1}{x_1} + \ldots + \frac{\mu_m}{x_m}$, $j = 1, 2, \ldots, n$, is created based on the generation of random numbers in [0, 1]. After the completion of the evolution by genetic algorithms, the best discrete fuzzy set in the population is $\tilde{X}_k^* = \frac{\mu_{k0}^*}{x_0} + \frac{\mu_{k1}^*}{x_1} + \ldots + \frac{\mu_{km}^*}{x_m}$. The fuzzy profit is $N(\tilde{X}_k^*) = \frac{\mu_{k0}^*}{N(x_0)} + \frac{\mu_{k1}^*}{N(x_1)} + \ldots + \frac{\mu_{km}^*}{N(x_m)}$. After the centroid method is used for defuzzification, the estimated profit in the fuzzy sense is obtained: $\theta(N(\tilde{X}_k^*)) = \frac{\sum_{j=0}^{m} N(x_j)\mu_{kj}^*}{\sum_{j=0}^{m} \mu_{kj}^*}$. Consider Example 2 in section 4. The crisp optimal solutions are $P(5) = 76$ and $N(5) = 365$. The best discrete fuzzy set obtained from six runs is given in Fig. 6, represented by $\tilde{X}_k^* = \sum_{j=0}^{10} \frac{\mu_j}{x_j}$. The fuzzy profit is $N(\tilde{X}_k^*) = \sum_{j=0}^{10} \frac{\mu_j}{N(x_j)}$, $N(x_j) = 0, j = 1, \ldots, 10$, and the centroid is $\theta(N(\tilde{X}_k^*)) = \frac{\sum_{j=0}^{10} N(x_j)\mu_j}{\sum_{j=0}^{10} \mu_j} = 359.367$. In addition, the best discrete fuzzy set in the population when using Strategy 2 is $\tilde{X}_k^* = \frac{0.915}{5.0}$. The fuzzy profit is $N(\tilde{X}_k^*) = \frac{0.915}{N(5.0)} = \frac{0.915}{365}$, and the centroid is $\theta(N(\tilde{X}_k^*)) = \frac{365 \times 0.915}{0.915} = 365$.

6. CONCLUSIONS

This study has investigated the genetic algorithm approach to solving fuzzy optimization equations without using the membership functions of fuzzy numbers. When genetic algorithms are applied to solve fuzzy equations, the computation uses neither the extension principle nor the interval arithmetic and α-cuts. The results of this study may lead to the development of effective genetic algorithms for solving general fuzzy optimization problems. In summary, the fuzzy concept of the proposed genetic algorithm approach is different and gives almost the same results as the traditional methods.

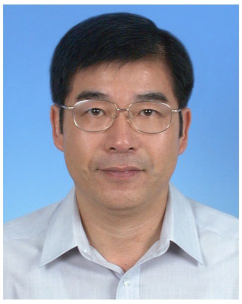
ACKNOWLEDGEMENT

The authors are grateful to the anonymous referees, whose valuable comments helped to improve the content of this paper.

REFERENCES

1. J. J. Buckly and Y. Hayashi, “Fuzzy genetic algorithm and applications,” *Fuzzy Sets and Systems*, Vol. 61, 1994, pp. 129-136.
2. J. J. Buckley and Y. Qu, “On using α-cuts to evaluate fuzzy equations,” *Fuzzy Sets and Systems*, Vol. 38, 1990, pp. 309-312.
3. J. J. Buckley and Y. Qu, “Solving linear and quadratic fuzzy equations,” *Fuzzy Sets and Systems*, Vol. 38, 1990, pp. 43-59.
4. J. J. Buckley, “Solving fuzzy equations in economics and finance,” *Fuzzy Sets and Systems*, Vol. 48, 1992, pp. 289-296.
5. L. Davis, ed., *Genetic Algorithms and Simulated Annealing*, San Mateo, CA: Morgan Kaufmann, 1987.
6. M. Gen and R. Cheng, *Genetic Algorithms & Engineering Design*, John Wiley & Sons, Inc., New York, 1997.
7. R. E. Giachetti, “Evaluating engineering functions with imprecise quantities,” *7th International Fuzzy Systems Association Congress*, Prague, Czech Republic, 1997.
8. D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA, 1989.
9. F. Herrera and M. Lozano, “Fuzzy genetic algorithms: issues and models,” Dept. of Science and A. I., University of Granada, Technical Report No. 18071, Granada, Spain, 1999.
10. F. Herrera, M. Lozano, and J. L. Verdegay, “Applying genetic algorithms in fuzzy optimization problems,” *Fuzzy Sets & Artificial Intelligence*, Vol. 3, 1994, pp. 39-52.
11. J. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.
12. A. Homaifar and E. McCormick, “Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms,” *IEEE Transcations on Fuzzy Systems*, Vol. 3, 1995, pp. 129-139.
13. C. L. Karr and E. J. Gentry, “Fuzzy control of pH using genetic algorithms,” *IEEE Transactions of Fuzzy Systems*, Vol. 1, 1993, pp. 46-53.
14. M. A. Lee and H. Takagi, “Dynamic control of genetic algorithms using fuzzy logic techniques,” in *Proceedings of Fifth International Conference on Genetic Algorithms* (ICGA ’93), 1993, pp. 76-83.
15. M. Mitchell, *An Introduction to Genetic Algorithms*, A Bradford Book, The MIT Press, Mass., 1996.
16. T. J. Ross, *Fuzzy Logic with Engineering Applications*, McGraw-Hill Inc., New York, 1995.
17. M. Sakawa, K. Kato, H. Sunada, and T. Shibano, “Fuzzy programming for multiobjective 0-1 programming problems through revised genetic algorithms,” *European Journal of Operational Research*, Vol. 97, 1997, pp. 149-158.
18. E. Sanchez, T. Shibata, and L. A. Zadeh, ed., *Genetic Algorithms and Fuzzy Logic Systems, Soft Computing Perspectives*, World Scientific, 1996.
19. C. H. Wang, T. P. Hong, and S. S. Tseng, “Integrating fuzzy knowledge by genetic algorithms,” *IEEE Transactions on Evolutionary Computation*, Vol. 2, 1998, pp. 138-149.
20. J. S. Yao and D. C. Lin, “Optimal fuzzy profit for price in fuzzy sense,” *Fuzzy Sets*

and Systems, Vol. 111, 2000, pp. 455-464.
21. J. S. Yao and S. C. Chang, "Economic principle of profit in fuzzy sense," *Fuzzy Sets and Systems*, Vol. 111, 2000, pp. 465-495.



Feng-Tse Lin (林豐澤) was born in Taipei, Taiwan. He received the M.S. degree in computer engineering from the National Chiao-Tung University, Hsinchu, Taiwan, in 1984, and the Ph.D. degree in computer science and information engineering from the National Taiwan University, Taipei, Taiwan, in 1992. From 1984 to 1987, he was a research assistant at Telecommunication Laboratories, Chung-Li, Taiwan. Currently, he is an Associate Professor with the Department of Applied Mathematics, Chinese Culture University, Yangminshan, Taipei, Taiwan. In 1997, he was a Visiting Associate Professor, Division of Engineering and Applied Sciences, Harvard University, Cambridge, MA. His current research interests include fuzzy logic and its applications, combinational optimization problems, genetic algorithms, learning classifier systems, and intelligent agents.

Dr. Lin is a member of Taiwanese Association for Artificial Intelligence (TAAI), Chinese Fuzzy Systems Association Taiwan (CFSAT), Computer Society of the ROC, and IEEE Systems, Man, and Cybernetics Society.

Jing-Shing Yao (姚景星) was born in Taipei, Taiwan. He received the Ph.D. degree in Mathematics from National Kyushu University, Japan, in 1973. He has been with the Department of Mathematics, National Taiwan University, Taipei, Taiwan, since 1965. He was an associate professor in 1965-1966 and as a professor since 1967. He headed the Department of Mathematics, National Taiwan University and also served as Director of Mathematical center, National Science Council, R. O. C., from 1976 to 1978. From 1993 to 1998, he was with Department of Applied Mathematics, Chinese Culture University, Yangminshan, Taipei, Taiwan. He is currently a Professor Emeritus at National Taiwan University, Taipei, Taiwan, and also an Honorary Professor at Management College, Tamkang University, Tamsui, Taiwan. His current research interests focus on operation research, fuzzy mathematics and statistics. Dr. Yao has published more than 50 theoretical papers in various international journals in the above areas.