Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 185 (2008) 133–145
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Discrete Optimization

Solving the knapsack problem with imprecise weight coefficients using genetic algorithms

Feng-Tse Lin

Department of Applied Mathematics, Chinese Culture University, Yangminshan, Taipei 111, Taiwan

Received 28 December 2005; accepted 29 December 2006
Available online 1 February 2007

Abstract

This paper investigates solving the knapsack problem with imprecise weight coefficients using genetic algorithms. This work is based on the assumption that each weight coefficient is imprecise due to decimal truncation or coefficient rough estimation by the decision-maker. To deal with this kind of imprecise data, fuzzy sets provide a powerful tool to model and solve this problem. We investigate the possibility of using genetic algorithms in solving the fuzzy knapsack problem without defining membership functions for each imprecise weight coefficient. The proposed approach simulates a fuzzy number by distributing it into some partition points. We use genetic algorithms to evolve the values in each partition point so that the final values represent the membership grade of a fuzzy number. The empirical results show that the proposed approach can obtain very good solutions within the given bound of each imprecise weight coefficient than the fuzzy knapsack approach. The fuzzy genetic algorithm concept approach is different, but gives better results than the traditional fuzzy approach.

Keywords: Genetic algorithms; Fuzzy sets; Knapsack problem; Fuzzy knapsack problem

1. Introduction

The knapsack problem derives its name from finding a combination of different objects that a hitchhiker chooses for his knapsack in which the total value of all chosen objects is maximized. A similar problem very often appears in business, combinatorics, complexity theory, cryptography and applied mathematics. Knapsack problems have been intensively studied, especially in the last decade, attracting both theorists and experimentalists [11,13,14]. The theoretical interest arises mainly from their structure in which more complex optimization problems can be solved through a series of knapsack-type subproblems. From the practical point of view, these problems can model many industrial situations, such as capital budgeting, or cargo loading [2,10,12]. As a result, a knapsack may correspond to a truck, a ship, a silicon chip, or a resource and the problem is to package these items to produce a variety of applications for cargo loading, cutting stock, bin packing, and economic planning [15,16].

In this study we are interested in investigating how to deal with knapsack problem imprecise data

E-mail address: ftlin@faculty.pccu.edu.tw

0377-2217/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2006.12.046

that would occur in practical situations. For example, the decision-maker (DM) has only a vague knowledge about the objects, which uses rough estimation and fuzzy linguistics, or decimal truncation for the problem weight coefficients. Therefore, we consider the weight of each object in the knapsack problem to be imprecise, which may be in the vicinity of a fixed value or substantially less than or greater than that fixed value. To deal with this kind of imprecise data, fuzzy sets provide a powerful tool to model and solve the problem [9]. The fuzzy concept is stated briefly as follows. For each weight $w_i,\ i = 1,\ldots,n$, the DM should determine an interval $[w_i - \Delta_{i1}, w_i + \Delta_{i2}],\ 0 \leqslant \Delta_{i1} < w_i$ and $0 < \Delta_{i2}$, to represent an acceptable range for the weight of each object. If an estimate of the weight is exactly w_i, then the acceptable grade for that weight will be 1. Otherwise, the acceptable grade will get smaller when an estimate approaches one of the ends of the interval, i.e. $w_i - \Delta_{i1}$ or $w_i + \Delta_{i2}$. Accordingly, the DM needs to determine an appropriate estimate for each weight from the interval $[w_i - \Delta_{i1}, w_i + \Delta_{i2}]$. This leads to the use of fuzzy numbers, $\tilde{w}_i = (w_i - \Delta_{i1}, w_i, w_i + \Delta_{i2})$, for each weight coefficient in the knapsack problems. The advantage of using fuzzy model is that it is much easier for the DM to specify a range value than giving an exact value for each weight coefficient. Hence, we proposed a fuzzy knapsack model in [9] that uses the signed-distance ranking method for defuzzifying the fuzzy weight coefficients and the fuzzy capacity to derive a defuzzified fuzzy knapsack problem.

In this paper, however, we formulate another knapsack problem with imprecise weight coefficients from the previously proposed fuzzy knapsack model. We investigate the application of genetic algorithms (GAs) for the possibility of solving knapsack problem with imprecise weight coefficients without defining the membership functions for each imprecise weight coefficient. In this approach, we simulate a fuzzy number by distributing it into some partition points, and then use GAs to evolve the values in each partition point so that the final values represent the membership grade of that fuzzy number. When GAs are applied to solve the fuzzy knapsack problem, the computation of fuzzy equations does not require the extension principle or interval arithmetic and α-cuts. Instead, GAs use only the usual evolution. The empirical results show that the proposed approach can obtain very good solutions within the given bound for each weight coefficient so that it accomplishes flexible item packing of the knapsack problem. Consequently, the proposed GA approach is a sound and flexible way for specifying the imprecise weight coefficients while dealing with the knapsack problem in practical situations.

2. The problem formulation

2.1. Knapsack problems

The knapsack problem is formulated by numbering the objects from 1 to n and by introducing a vector of binary variables $x_i\ (i = 1,\ldots,n)$ having the following meaning. Object i has a weight w_i and the knapsack has a capacity M. If a fraction $x_i,\ 0 \leqslant x_i \leqslant 1$, of object i is placed in the knapsack, then a profit, $g_i x_i$, is earned. The objective is to find the combination of objects for the knapsack that maximizes the total profit from all of the objects chosen [11]. Since the knapsack's capacity is M, we require the total weight of all of the chosen objects to be at most M. The knapsack problem is stated mathematically as follows:

$$\text{maximize} \quad \sum_{i=1}^{n} g_i x_i \tag{1}$$

$$\text{subject to} \quad \sum_{i=1}^{n} w_i x_i \leqslant M \tag{2}$$

$$\text{and} \quad 0 \leqslant x_i \leqslant 1,\ \ 1 \leqslant i \leqslant n. \tag{3}$$

The optimal solution is obtained using the greedy strategy where the objects are considered in the order of non-increasing density g_i/w_i and are added to the knapsack if they fit.

2.2. Fuzzy knapsack problem

The essential problem we consider here is that the value of each weight coefficient $w_i,\ i = 1,\ldots,n$, is imprecise. This is because the DM has only a vague knowledge about the objects and uses rough estimation, fuzzy linguistics, or decimal truncation for the weight coefficient of the knapsack problem. Thus, the DM should determine an acceptable range of values for each w_i, which is an interval $[w_i - \Delta_{i1}, w_i + \Delta_{i2}],\ 0 \leqslant \Delta_{i1} < w_i$ and $0 < \Delta_{i2}$. After that, the DM chooses a value from the interval $[w_i - \Delta_{i1}, w_i + \Delta_{i2}]$ as an estimate of each weight coefficient. The acceptable grade is 1 if the estimate is exactly w_i; otherwise, the acceptable grade will get smaller when the estimate approaches either $w_i - \Delta_{i1}$ or $w_i + \Delta_{i2}$. We believe that the fuzzy

weight coefficient tolerance interval is the most valid information for the DM when he manages the knapsack problem in practical situations. Let $\tilde{w}_i$ be the fuzzy number denoted by

$$\tilde{w}_i = (w_i - \Delta_{i1}, w_i, w_i + \Delta_{i2}), \quad 0 \leqslant \Delta_{i1} < w_i, \quad 0 \leqslant \Delta_{i2}, \ \ 1 \leqslant i \leqslant n. \tag{4}$$

The membership function of $\tilde{w}_i$ is as shown below:

$$\mu_{\tilde{w}_i}(x) = \begin{cases} \frac{x - w_i + \Delta_{i1}}{\Delta_{i1}}, & w_i - \Delta_{i1} \leqslant x \leqslant w_i, \\ \frac{w_i + \Delta_{i2} - x}{\Delta_{i2}}, & w_i \leqslant x \leqslant w_i + \Delta_{i2}, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Obviously, when an estimate x equals w_i, the membership grade of x in $\tilde{w}_i$ is 1. The farther the position of w_i is from an estimate x, the less the x membership grade in $\tilde{w}_i$ obtained. We already proposed a fuzzy knapsack model using the signed-distance ranking method for defuzzifying the fuzzy weight coefficients and the fuzzy capacity to derive a defuzzified fuzzy knapsack problem [9]. We summarize the proposed fuzzy knapsack model as shown below. Let $F_{\mathrm{P}}(1)$ denote the family of all level 1 fuzzy points. When fuzzifying the knapsack capacity, we let

$$\tilde{M} = (M, M, M) \in F_{\mathrm{P}}(1) \quad \text{and} \quad \mathrm{d}(\tilde{M}, \tilde{0}_1) = M. \tag{6}$$

Note that $\mathrm{d}(\tilde{M}, \tilde{0}_1) = M$ is the signed distance ranking of $\tilde{M}$ measured from $\tilde{0}_1$ (y-axis). After defuzzifying the fuzzy number $\tilde{w}_i$ using the signed distance ranking method, we obtain an estimate of the weight from the interval $[w_i - \Delta_{i1}, w_i + \Delta_{i2}]$, i.e.

$$w_i^* = \mathrm{d}(\tilde{w}_i, \tilde{0}_1) = w_i + \frac{1}{4}(\Delta_{i2} - \Delta_{i1}), \quad i = 1, 2, \ldots, n. \tag{7}$$

This equation is the signed distance of $\tilde{w}_i$ measured from $\tilde{0}_1$. The DM can then make use of the equation to obtain a value as an estimate of each weight for solving the fuzzy knapsack problem. Since $\mathrm{d}(\tilde{w}_i, \tilde{0}_1) = \frac{3}{4}w_i + \frac{1}{4}\Delta_{i2} + \frac{1}{4}(w_i - \Delta_{i1}) > 0$, w_i^* is a positive number measured from 0. Therefore, w_i^* in (7) represents an estimate of the weight coefficient w_i, which is equal to w_i plus some fuzzy quantity $\frac{1}{4}(\Delta_{i2} - \Delta_{i1})$, $w_i^* \in [w_i - \Delta_{i1}, w_i + \Delta_{i2}]$. We fuzzify (2) to obtain (8), where $\lesssim$ is the ranking defined on $F = \{(a, b, c) | \forall a \leqslant b \leqslant c, a, b, c \in R\}$.

$$\sum_{i=1}^{n} \tilde{w}_i x_i \lesssim \tilde{M}. \tag{8}$$

From (1), (3), and (8), we formulate the following fuzzy knapsack problem.

$$\text{maximize} \quad \sum_{i=1}^{n} g_i x_i \tag{9}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \tilde{w}_i x_i \lesssim \tilde{M} \tag{10}$$

$$0 \leqslant x_i \leqslant 1, \quad i = 1, \ldots, n. \tag{11}$$

Finally, we derive the following defuzzified fuzzy knapsack problem.

$$\text{maximize} \quad \sum_{i=1}^{n} g_i x_i \tag{12}$$

$$\text{subject to} \quad \sum_{i=1}^{n} w_i^* x_i \leqslant M \tag{13}$$

$$0 \leqslant x_i \leqslant 1, \ \ i = 1, \ldots, n, \tag{14}$$

where

$$w_i^* = w_i + \tfrac{1}{4}(\Delta_{i2} - \Delta_{i1}), \ \ 0 \leqslant \Delta_{i1} < w_i, \ \ 0 \leqslant \Delta_{i2}.$$

3. The knapsack problem with imprecise weight coefficients

In this section, we formulate the knapsack problem with imprecise weight coefficients from the fuzzy knapsack problem defined in (9)–(11). The imprecise knapsack problem weight coefficient concept is depicted as follows. Let the triangular fuzzy number $\tilde{w} = (w - \Delta_1, w, w + \Delta_2)$ be replaced by an arbitrary fuzzy set $\tilde{W}$ in a given interval $[a, b]$ (see Fig. 1).

We divide the interval $[a, b]$ into t partitions, $p_k = a + k \times \frac{b-a}{t}$, $k = 0, 1, \ldots, t$, and call each p_k a partition point (i.e. a separate piece). Let the membership grade of $\tilde{W}$ at p_k be $\tilde{W}(p_k) = \mu_k$, $k = 0, 1, \ldots, t$ and $\mu_k \in [0, 1]$. A discrete fuzzy set is then obtained as follows:

$$\tilde{W} = (\mu_0, \mu_1, \ldots, \mu_t) = \frac{\mu_0}{p_0} + \frac{\mu_1}{p_1} + \cdots + \frac{\mu_t}{p_t}. \tag{15}$$

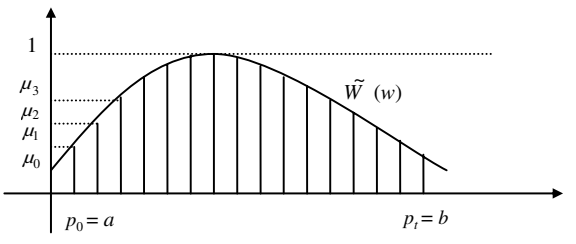

Fig. 1. Dividing fuzzy set $\tilde{W}$ into partition points.

For each coefficient in (8), the DM can provide some leeway in the constraints in order to perform flexible item packing. Assume that $\tilde{w}$ is a triangular fuzzy number or triangular shaped fuzzy number defined on the interval $[w-\Delta_{i1}, w+\Delta_{i2}]$. This interval is further divided equally into t partitions. Let $p_k = w-\Delta_{i1}+k\times\frac{\Delta_{i2}+\Delta_{i1}}{t}$, $k=0,1,\ldots,t$ be the partition points and let $\tilde{W}(p_k)=\mu_k\in[0,1]$, $k=0,1,\ldots,t$, be the membership grade of p_k in an arbitrary fuzzy set $\tilde{W}$. Thus, we obtain a discrete fuzzy set $\tilde{W}=(\mu_0,\mu_1,\ldots,\mu_t)$, each μ_k, $k=0,1,\ldots,t$, is a random number in $[0,1]$. In the proposed approach, we wish to find an estimated value of w in $[w-\Delta_{i1}, w+\Delta_{i2}]$ via GAs. After computing the centroid of the fuzzy number $\tilde{w}$, which is defined on the discrete fuzzy set $\tilde{W}=(\mu_0,\mu_1,\ldots,\mu_t)$, we obtain the estimated value w^* as follows:

$$w^* = \frac{\sum_{k=0}^{t} p_k \times \mu_k}{\sum_{i=0}^{t}\mu_k}. \tag{16}$$

Consider an arbitrary given function $N(x)=y$ in the knapsack problem. If each $N(x_k)$, $k=0,1,\ldots,t$, are different, then the fuzzy function of $N(x)$ is defined as

$$\begin{aligned} N(\tilde{X}) &= N(\mu_0,\mu_1,\ldots,\mu_t) \\ &= \frac{\mu_0}{N(x_0)}+\frac{\mu_1}{N(x_1)}+\cdots+\frac{\mu_t}{N(x_t)}, \end{aligned} \tag{17}$$

and the centroid is defined as

$$\theta(N(\tilde{X})) = \frac{\sum_{k=0}^{t} N(x_k)\mu_k}{\sum_{k=0}^{t}\mu_k}. \tag{18}$$

Following this concept, we rewrite (8) as

$$\sum_{i=1}^{n} w^* x_i \leqslant M^*. \tag{19}$$

Each w^* in (16) is an estimated value obtained from computing the centroid of $\tilde{w}$. Similarly, M^* in (19) is also an estimated value obtained from computing the centroid of $\tilde{M}$. Finally, we transform the fuzzy knapsack problem in (9)–(11) into the following form:

$$\text{maximize} \quad Z=\sum_{i=1}^{n} g_i x_i \tag{20}$$

$$\text{subject to} \quad \sum_{i=1}^{n} w_i^* x_i \leqslant M^* \tag{21}$$

$$\text{and} \quad 0\leqslant x_i\leqslant 1,\ \ i=1,\ldots,n. \tag{22}$$

Z will be used as the fitness value of each chromosome in the population. We call (20)–(22) the knapsack problem with imprecise weight coefficients.

4. The proposed genetic algorithm approach

Genetic algorithms (GAs) were introduced as a computational analogy of adaptive systems. They are stochastic search techniques based on the principles and mechanisms of natural genetics and modeled on evolutionary principles via natural selection employing a population of individuals. A fitness function is used to evaluate individuals and reproductive success varies with fitness, modeled on natural genetic inheritance and Darwinian survival-of-the-fittest [3]. The basic GA concept is that the system starts with a population of randomly generated candidates that evolve towards better solutions by applying genetic operators, such as crossover and mutation. GAs are useful and efficient when the search space is large, complex or poorly understood, domain knowledge is scarce or expert knowledge is difficult to encode to narrow the search space, mathematical analysis is unavailable, and traditional search methods fail [3].

GAs can be used to compute the membership functions of fuzzy sets [6,7]. Given some functional mapping for a system, some membership functions and their shapes are assumed for the various fuzzy variables defined for a problem. 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 GAs [4]. One involves the application of GAs for solving optimization and search problems related to fuzzy systems [1,5,18]. The other, is the use of fuzzy tools and fuzzy logic-based techniques for modeling different GA components and adapting GA control parameters, with the goal of improving performance [4,8,17,19]. In this paper, however, we propose a different view of the GA approach to simulate a fuzzy number by distributing it into some partition points. We then use GAs to evolve the values in each partition point and the final values represent the membership grades of that fuzzy number. The proposed GA approach for solving the knapsack problem with imprecise weight coefficients is stated in the following.

Step 1. Generate an initial population. An initial population of size n is randomly generated from $[0,1]^{t+1}$ according to the uniform distribution in the closed interval $[0,1]$. Let the population be $\tilde{W}_h=(\mu_{h_0},\mu_{h_1},\ldots,\mu_{h_t})=\frac{\mu_{h_0}}{p_0}+\frac{\mu_{h_1}}{p_1}+\cdots+\frac{\mu_{h_t}}{p_t}$, where $h=1,2,$

$\ldots, n$, μ_{h_k} is a real number in $[0,1]$ and p_k is a regular partition point in a given interval, $k = 0, 1, 2, \ldots, t$. Each individual $\tilde{W}_h$, $h = 1, 2, \ldots, n$, in a population is a chromosome.

Step 2. Calculate the fitness value for each chromosome. Each chromosome $\tilde{W}_h$, $h = 1, 2, \ldots, n$, in the population is evaluated, according to (16), by creating the estimated value for each coefficient in (21). The fitness value of each chromosome is then obtained from (20). The chromosomes in the population can be rated in terms of their fitness values. Let the total fitness value of the population be T. The cumulative fitness value (partial sum) for each chromosome S_h, $h = 1, 2, \ldots, n$, is calculated. The 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.

Step 3. Selection and reproduction. Reproduction is a process in which each chromosome is copied according to the selection process. The roulette wheel selection mechanism is the selection strategy chosen in this paper. The selection process begins with spinning 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 $\tilde{W}_1$ is selected; otherwise, the kth chromosome $\tilde{W}_k$, $2 \leqslant k \leqslant n$, 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. This procedure tends to choose more $\tilde{W}_h$, $h = 1, 2, \ldots, n$, with higher fitness to go on into the next population.

Step 4. Perform crossover. Crossover is the key to the power of genetic algorithms. The purpose of crossover is to generate the rearrangement of co-adapted groups of information from high performance structures [3]. The crossover method used here is the one-point method, which randomly selects one cut-point and exchanges the right parts of two randomly selected parents in the population to generate offspring. Let r_c is the probability of a crossover, then $0 \leqslant r_c \leqslant 1$. Usually, r_c is set to be 0.9, so we expect that on 90% of the chromosomes will undergo crossover on average.

Step 5. Perform mutation. Mutation is a background operator that produces random changes in various chromosomes [3]. Mutation resets a selected position in a chromosome to a randomly generate real number in $[0,1]$, for example say 0.35. The number of selected positions for mutation in the population is related to the mutation rate. Let r_m be the mutation rate (probability), $0 \leqslant r_m \leqslant 1$. Usually r_m is a very small value, around 0.003, so we expect that, on average, 0.3% of the total population will undergo mutation. After this step, an 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 $\tilde{W}_1^*, \tilde{W}_2^*, \ldots, \tilde{W}_n^*$. The maximum value of fitness is the best chromosome in the population. The best chromosome represents an approximately best solution obtained by GA for the problem. Let the best chromosome be $\tilde{W}_h^* = (\mu_{h0}^*, \mu_{h1}^*, \ldots, \mu_{ht}^*) = \frac{\mu_{h0}^*}{p_0} + \frac{\mu_{h1}^*}{p_1} + \ldots + \frac{\mu_{ht}^*}{p_t}, 1 \leqslant h \leqslant n$. The estimated value of each fuzzy coefficient $\tilde{w}$ in (8) is calculated as $w^* = \frac{\sum_{k=0}^{t} p_k \times \mu_{hk}^*}{\sum_{k=0}^{t} \mu_{hk}^*}$, where $p_k = w - \Delta_{i1} + k \times \frac{\Delta_{i2} + \Delta_{i1}}{t}$, $k = 0, 1, \ldots, t$, and is defined on the interval $[w - \Delta_{i1}, w + \Delta_{i2}]$. Similarly, $M^* = \frac{\sum_{k=0}^{t} p_k \times \mu_{hk}^*}{\sum_{k=0}^{t} \mu_{hk}^*}$, where $p_k = M - \Delta_{i1} + k \times \frac{\Delta_{i2} + \Delta_{i1}}{t}$, $k = 0, 1, \ldots, t$, is defined on the interval $[M - \Delta_{i1}, M + \Delta_{i2}]$. In our implementation of this fuzzy knapsack problem, we use $t = 10$ and $0.2 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 2.0$, which will be discussed in the next section. For this particular problem we have chosen the following parameters: the number of generations $K = 5000$, population size 100, the probability of crossover $r_c = 0.9$, and the probability of mutation $r_m = 0.003$. We discuss these parameter settings in the Appendix.

5. Empirical results

Two empirical examples are given below to illustrate the effectiveness of GAs upon applying them to simulate the fuzzy numbers for solving the knapsack problem with imprecise weight coefficients.

Example 1. An instance of knapsack problem is given in Table 1.

The optimal solution obtained using the greedy strategy for the crisp knapsack problem defined in (1)–(3) is $(x_i) = (1, 1, 1, 0, 1, 0.610)$, $i = 1, \ldots, 6$, with the optimal profit of 78.244. Consider the fuzzy knapsack problem defined in (9)–(11). Each weight coefficient and the knapsack's capacity is a fuzzy number defined in (4), $\tilde{w}_i = (w_i - \Delta_{i1},\ w_i, w_i + \Delta_{i2})$, $0 \leqslant \Delta_{i1} < w_i, 0 \leqslant \Delta_{i2}, 1 \leqslant i \leqslant n$. It has been assumed that the DM has determined the values of Δ_{i1} and Δ_{i2}, $i = 1, \ldots, 6$, for each weight coefficient and the knapsack's capacity, as shown in Table 2. According to (7), the DM calculates each estimated weight coefficient w_i^* using $w_i + \frac{1}{4} \times (\Delta_{i2} - \Delta_{i1})$. We can see that $\frac{1}{4}(\Delta_{i2} - \Delta_{i1})$ is the fuzzy quantity for each fuzzy weight coefficient after using the signed distance ranking method for defuzzification. Thus, the estimated weight coefficients for the fuzzy knapsack problem defined in Eqs. (12)–(14) are $w_1^* = 8.2$, $w_2^* = 12.3$, $w_3^* = 13.05$, $w_4^* = 63.9$, $w_5^* = 22.15$, and $w_6^* = 40.9$. According to (6), the estimated knapsack capacity M^* measured from $\tilde{0}_1$ (y-axis) is 80. These results are shown in Table 2. After solving the fuzzy knapsack problem, we obtain the solution vector $(x_i) = (1, 1, 1, 0, 1, 0.558)$, $i = 1, \ldots, 6$, with the best profit of 77.853. We compare the result obtained from the fuzzy knapsack problem with that of the crisp knapsack problem as $\frac{77.853 - 78.244}{78.244} = -0.499\%$.

Another set of values for Δ_{i1} and Δ_{i2} determined by the DM are considered, as shown in Table 3. In this data set, the estimated weight coefficients for the fuzzy knapsack problem are $w_1^* = 7.9$, $w_2^* = 12.1$, $w_3^* = 12.7$, $w_4^* = 63.9$, $w_5^* = 22.1$, and $w_6^* = 41.2$. The best solution in this instance is $(x_i) = (1, 1, 1, 0, 1, 0.612)$, $i = 1, \ldots, 6$, with the best profit of 78.291. Again, we compare the result obtained with that from the crisp problem as $\frac{78.291 - 78.244}{78.244} = 0.0604\%$. The profit obtained at this time is better than the previous one. However, according to (7), if $\Delta_{i1} = \Delta_{i2}$, we obtain $w_i^* = w_i$. In this case, the fuzzy knapsack problem becomes the crisp knapsack problem.

Now, we use the proposed GA approach, via the simulation of fuzzy numbers by distributing them in some partition points, to solve the knapsack problem with imprecise weight coefficients defined in (20)–(22). Let $t = 10$. The partition points are defined by $p_k = w_i - \Delta_{i1} + k \times \frac{\Delta_{i2} + \Delta_{i1}}{10}$, $k = 0, 1, \ldots, 10$. This means that each imprecise weight coefficient w_i is divided into 11 partition points by GAs to simulate the fuzzy number $\tilde{w}_i = (w_i - \Delta_{i1}, w_i, w_i + \Delta_{i2})$. The parameters used in the GA approach to solve the knapsack problem with imprecise weight coefficients are: (1) the population size is 100; (2) the probability of a crossover is 0.9, (3) the probability of a mutation is 0.003, and (4) the number of generations is 5000 (see the Appendix in detail). According to the values of each Δ_{i1} and Δ_{i2} as shown in Table 3, the range for each imprecise weight coefficient are $w_1 = [7.5, 8.2]$, $w_2 = [11.6, 13.8]$, $w_3 = [11.8, 13.2]$, $w_4 = [63.5, 64.2]$, $w_5 = [21.7, 23.0]$, and $w_6 = [40.8, 42.0]$. The range of imprecise knapsack capacity $M = [79.5,\ 81.5]$. The partition points are $p_k = w_i - \Delta_{i1} + k \times \frac{\Delta_{i2} + \Delta_{i1}}{10}$, $k = 0, 1, \ldots, 10$. Table 4 depicts the membership grades of 11 partition points for each imprecise weight coefficient w_i, $i = 1, \ldots, 6$, and the knapsack capacity M, which are all obtained from the last GA program generation.

For example, the membership grade of 11 partition points for w_1 are $\mu_0 = 0.166$, $\mu_1 = 0.989$, $\mu_2 = 0.588$, $\mu_3 = 0.161$, $\mu_4 = 0.491$, $\mu_5 = 0.240$, $\mu_6 = 0.255$, $\mu_7 = 0.444$, $\mu_8 = 0.525$, $\mu_9 = 0.492$, and $\mu_{10} = 0.400$. The GA program calculates the estimated weight coefficients and the estimated knapsack capacity using the membership grade defined in (16) for each generation. The evolution continues until the stop criterion is met. In the last generation, the estimated weight coefficients obtained are $w_1^* = 7.821$, $w_2^* = 12.253$, $w_3^* = 12.412$, $w_4^* = 63.734$,

Table 1
A knapsack problem with $n = 6$ and $M = 80$

i	1	2	3	4	5	6
w_i	8	12	13	64	22	41
g_i	10	15	20	12	18	25

Table 2
The values of Δ_{i1} and Δ_{i2} for each weight coefficient and the knapsack capacity

i	1	2	3	4	5	6	M
$(\Delta_{i1}, \Delta_{i2})$	(0.2, 1)	(0.4, 1.6)	(0.6, 0.8)	(1.0, 0.6)	(0.7, 1.3)	(1.0, 0.6)	(0.3, 0.8)
w_i^*	8.2	12.3	13.05	63.9	22.15	40.9	

Table 3
Another set of the values for each Δ_{i1} and Δ_{i2} and w_i^*

i	1	2	3	4	5	6	M
$(\Delta_{i1}, \Delta_{i2})$	(0.5, 0.2)	(0.4, 0.8)	(1.2, 0.2)	(0.5, 0.2)	(0.3, 1.0)	(0.2, 1.0)	(0.3, 0.5)
w_i^*	7.9	12.1	12.7	63.9	22.1	41.2	

Table 4
The membership grades of 11 partition points obtained from the last GA program generation

partition points	$p0$	$p1$	p 2	p 3	$p4$	$p5$	p 6	$p7$	$p8$	$p9$	$p10$
$w1$	0.166	0.989	0.588	0.161	0.491	0.240	0.255	0.444	0.525	0.492	0.400
$w2$	0.180	0.805	0.908	0.168	0.296	0.373	0.955	0.957	0.887	0.706	0.755
$w3$	0.450	0.345	0.646	0.770	0.938	0.058	0.496	0.742	0.300	0.196	0.046
$w4$	0.761	0.950	0.857	0.210	0.562	0.346	0.027	0.217	0.042	0.077	0.481
$w5$	0.249	0.673	0.151	0.064	0.104	0.215	0.498	0.657	0.277	0.803	0.641
$w6$	0.189	0.543	0.770	0.670	0.422	0.078	0.406	0.821	0.103	0.944	0.660
M	0.384	0.441	0.305	0.799	0.167	0.838	0.709	0.685	0.887	0.249	0.332

$w_5^* = 22.503$, and $w_6^* = 41.437$, and the estimated knapsack capacity $M^* = 80.126$. The solution vector is $(x_i) = (1, 1, 1, 0, 1, 0.640)$, $i = 1, \ldots, 6$, with the best profit of 78.991. The comparison of the result obtained in the first run with that of the crisp problem is $\frac{78.991-78.244}{78.244} = 0.9558\%$. Obviously, the GA approach shows better performance than the fuzzy approach.

Three triangular fuzzy number shapes, as shown in Fig. 2, were considered in this study for analyzing the effectiveness and efficiency of the GA approach. The first type is a symmetrical shape, the second is the right-skewed shape, and the third is the left-skewed shape.

The symmetrical shape of the fuzzy number is an isosceles triangle, where $\Delta = \Delta_{i1} = \Delta_{i2}$. We assume $\Delta = 0.5$, the fuzzy number $\tilde{w}_i$ is $(w_i - \Delta, w_i, w_i + \Delta)$. The range of each imprecise weight coefficient are $w_1 = [7.5, 9.5]$, $w_2 = [11.5, 13.5]$, $w_3 = [12.5, 14.5]$, $w_4 = [63.5, 65.5]$, $w_5 = [21.5, 23.5]$, and $w_6 = [40.5, 42.5]$. The imprecise knapsack capacity $M = [79.5, 81.5]$. The partition points are $p_k = w_i - \Delta + k \times \frac{\Delta+\Delta}{10} = w_i - \Delta + 0.2 \times k$, $k = 0, 1, \ldots, 10$. We run the symmetrical shape GA program (the first type) 10 times. The estimated weight coefficients in the first run are $w_1^* = 7.928$, $w_2^* = 11.940$, $w_3^* = 12.938$, $w_4^* = 63.843$, $w_5^* = 21.993$, and $w_6^* = 41.013$, and the estimated knapsack capacity $M^* = 80.039$. The solution vector is $(x_i) = (1, 1, 1, 0, 1, 0.615)$, $i = 1, \ldots, 6$, with a best profit of 78.385. The comparison of the result obtained in the first run with that of the crisp problem is $\frac{78.385-78.244}{78.244} = 0.1812\%$. The best profits obtained in other nine runs, i.e. run#2 to run#10, are shown in Fig. 3. The percentage of relative differences between the best profit obtained in these runs is compared with that from the crisp problem, and are 0.0305, 0.0103, −0.0242, −0.0710, 0.1170, −0.0147, −0.0089, 0.0135, and 0.0905, respectively. Note that the estimated values of each weight coefficient w_i are in the interval $[w_i - 0.5, w_i + 0.5]$ for all runs.

Fig. 4 shows another 20 runs of the first type program to obtain 20 different best profits. The dotted line in Fig. 4 represents the crisp optimal solution, 78.244. The average of estimated weight coefficients obtained in these 20 runs are $w_1^* = 8.007$, $w_2^* = 11.978$, $w_3^* = 12.978$, $w_4^* = 64.009$, $w_5^* = 21.994$, and $w_6^* = 40.986$. The average estimated capacity M^* is

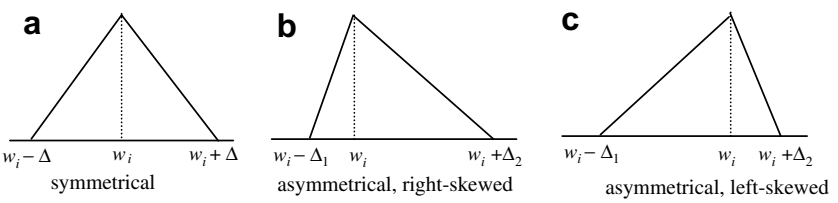

Fig. 2. Three triangular fuzzy number shapes considered in the example.

80.019, the average profit obtained is 78.321, and the average relative difference is 0.099% (slightly above the optimal crisp solution) in 20 runs.

The second type is asymmetrical right-skewed shape of fuzzy number, where $\Delta_{i1} < \Delta_{i2}$. In our implementation, we use $\Delta_{i2} = 3\Delta_{i1}$ and $\Delta_{i1} = 0.5$. The ranges for each imprecise weight coefficient of the second type are $w_1 = [7.5, 9.5]$, $w_2 = [11.5, 13.5]$, $w_3 = [12.5, 14.5]$, $w_4 = [63.5, 65.5]$, $w_5 = [21.5, 23.5]$, and $w_6 = [40.5, 42.5]$. The imprecise knapsack capacity M is [79.5, 81.5]. The partition points are $p_k = w_i - \Delta_{i1} + k \times \frac{\Delta_{i2}+\Delta_{i1}}{10} = w_i - 0.5 + 0.2 \times k$, $k = 0, 1, \ldots, 10$. The estimated weight coefficients obtained in the first run are $w_1^* = 8.770$, $w_2^* = 12.407$, $w_3^* = 13.440$, $w_4^* = 64.459$, $w_5^* = 22.530$, and $w_6^* = 41.392$, and the estimated knapsack capacity $M^* = 80.553$. The solution vector is $(x_i) = (1, 1, 1, 0, 1, 0.565)$, $i = 1, \ldots, 6$, with the best profit of 77.136. The comparison of the result obtained from the first run with that of the crisp problem is given as $\frac{77.136-78.244}{78.244} = -1.4152\%$. We execute the second type of program 20 times to obtain 20 different best profits, which are shown in Fig. 5. Again, the dotted line in Fig. 5 represents the crisp solution, 78.244. The average estimated weight coefficients obtained in 20 runs are $w_1^* = 8.510$, $w_2^* = 12.478$, $w_3^* = 13.468$, $w_4^* = 64.484$, $w_5^* = 22.488$, and $w_6^* = 41.477$. The average estimated capacity M^* is 80.532, the average profit obtained is 77.290, and the average relative difference rate is -1.219% (below the crisp solution).

The third type is the asymmetrical left-skewed shape of fuzzy number, where $\Delta_{i1} > \Delta_{i2}$. In our implementation, we use $\Delta_{i1} = 3\Delta_{i2}$ and $\Delta_{i2} = 0.5$. The range for each imprecise weight coefficient in the third type are $w_1 = [6.5, 8.5]$, $w_2 = [10.5, 12.5]$, $w_3 = [11.5, 13.5]$, $w_4 = [62.5, 64.5]$, $w_5 = [20.5, 22.5]$, and $w_6 = [39.5, 41.5]$. The imprecise knapsack capacity M is [78.5, 80.5]. The partition points are $p_k = w_i - \Delta_{i1} + k \times \frac{\Delta_{i2}+\Delta_{i1}}{10} = w_i - 1.5 + 0.2 \times k$, $k = 0, 1, \ldots, 10$. The estimated weight coefficients obtained from the first run are $w_1^* = 7.323$, $w_2^* = 11.355$, $w_3^* = 12.306$, $w_4^* = 63.639$, $w_5^* = 21.361$, and $w_6^* = 40.649$. The estimated knapsack capacity M^* is 79.666 and the solution vector is $(x_i) = (1, 1, 1, 0, 1, 0.672)$, $i = 1, \ldots, 6$, with the best profit of 79.803. The result comparison obtained from the first run with that from the crisp problem is given as $\frac{77.136-78.244}{78.244} = 1.9925\%$. The third type of program was also executed in another 20 runs to obtain 20 different best profits, which are shown in Fig. 6. The average of estimated weight coefficients obtained in 20 runs are $w_1^* = 7.479$, $w_2^* = 11.485$, $w_3^* = 12.497$, $w_4^* = 63.500$, $w_5^* = 21.463$, and $w_6^* = 40.470$. The average estimated capacity M^* is 79.535, the average profit obtained is 79.514, and

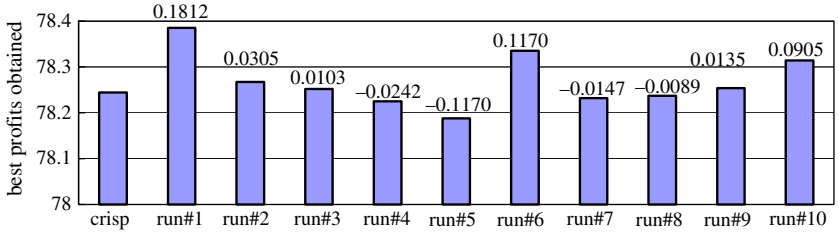

Fig. 3. Comparison of the best profits obtained from 10 runs with the optimal profit of the crisp problem.

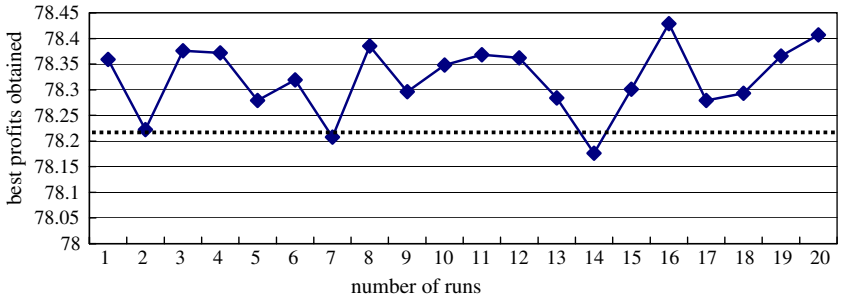

Fig. 4. The best profits curve obtained in 20 runs of the first type of program. The dotted line represents the crisp optimal solution, 78.244.

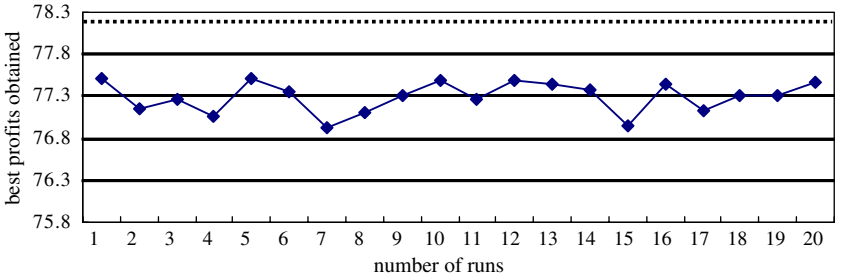

Fig. 5. The best profits curve obtained in 20 runs of the second type of program.

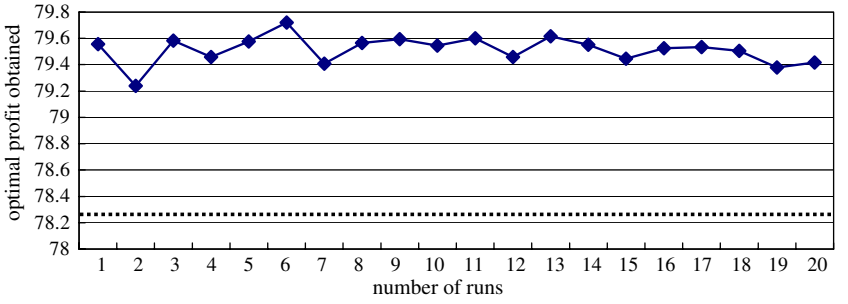

Fig. 6. The best profits curve obtained in 20 runs of the third type of program.

the average relative difference rate is 1.623% (above the crisp solution).

Example 2. Consider another instance of the knapsack problem given in Table 5. The optimal solution obtained using the greedy strategy for this crisp problem is $(x_i) = (0.452, 1, 1, 0, 1, 0, 0)$ with an optimal profit of 138.548. In this example, we first run the GA program using the delta ranges $0.2 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 1.2$. The actual values of Δ_{i1} and Δ_{i2} for each weight coefficient w_i, $i = 1, \ldots, 7$, used in the first run are shown in Table 6.

The GA program calculates the estimated weight coefficients according to (16), based on the membership grades obtained in the 11 partition points. For example, the estimated weight $w_1^* = 30.761$ is obtained from the range $[31 - 1.018, 31 + 0.802]$ based on the membership grades $\mu_0 = 0.668$, $\mu_1 = 0.640$, $\mu_2 = 0.614$, $\mu_3 = 0.511$, $\mu_4 = 0.093$, $\mu_5 = 0.740$, $\mu_6 = 0.232$, $\mu_7 = 0.799$, $\mu_8 = 0.843$, $\mu_9 = 0.014$ and $\mu_{10} = 0.128$. Similarly, we obtain $w_2^* = 10.106$, $w_3^* = 20.219$, $w_4^* = 29.027$, $w_5^* = 5.949$, $w_6^* = 3.577$ and $w_7^* = 6.090$. The estimated capacity M^* is 50.141, which is obtained from the interval $[50 - 0.548, 50 + 0.578]$ based on the membership grades $\mu_0 = 0.106$, $\mu_1 = 0.131$, $\mu_2 = 0.748$, $\mu_3 = 0.179$, $\mu_4 = 0.774$, $\mu_5 = 0.672$, $\mu_6 = 0.364$, $\mu_7 = 0.678$, $\mu_8 = 0.484$, $\mu_9 = 0.972$, and $\mu_{10} = 0.788$. The solution vector is $(x_i) = (0.374, 1, 1, 0, 1, 1, 1)$ with the best profit of 140.010. The comparison of the result obtained with that from the crisp problem is $\frac{140.010 - 138.548}{138.548} = 1.0558\%$.

Table 5
Another knapsack problem instance with $n = 7$ and $M = 50$

i	1	2	3	4	5	6	7
w_i	31.0	10.0	20.0	29.0	6.0	3.5	6.0
g_i	45.5	39.0	70.0	38.5	9.0	5.0	8.0

Following is a comparison of the estimated weight coefficients and the best profits obtained from the GA program with that of the fuzzy approach defined in (9)–(11). According to (6), the knapsack capacity M^* measured from $\tilde{0}_1$ is 50. According to (7), the estimated weight coefficients based on the signed-distance ranking method are $w_1^* = 30.946$, $w_2^* = 10.033$, $w_3^* = 19.983$, $w_4^* = 29.084$, $w_5^* = 5.998$, $w_6^* = 3.507$, and $w_7^* = 6.061$. The solution vector is $(x_i) = (0.452, 1, 1, 0, 1, 0, 0)$, $i = 1, \ldots, 7$, with a best profit of 138.563. The result comparison obtained from the defuzzified fuzzy knapsack problem defined in (12)–(14) with that from the crisp problem is $\frac{138.563 - 138.548}{138.548} = 0.0113\%$. Again, we can see that the performance

Table 6
The values of Δ_{i1} and Δ_{i2} used in the first run, $0.2 \leqslant \Delta_{i2}, \Delta_{i2} \leqslant 1.2$

i	1	2	3	4	5	6	7
$(\Delta_{i1}, \Delta_{i2})$	(1.018, 0.802)	(0.382, 0.514)	(0.996, 0.927)	(0.485, 0.821)	(1.084, 1.077)	(0.472, 0.500)	(0.260, 0.503)
w_i^* (GA)	30.761	10.106	20.219	29.027	5.949	3.577	6.090
w_i^* (fuzzy)	30.946	10.033	19.983	29.084	5.998	3.507	6.061

of the proposed GA approach is better than that from the fuzzy approach.

We execute the GA program for another seven runs using the delta ranges varying from $0.3 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 1.3$, $0.4 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 1.4$, $0.5 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 1.5$, $0.6 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 1.6$, $0.7 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 1.7$, $0.8 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 1.8$, to $0.9 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 1.9$. In Table 7, we list the actual values of $(\Delta_{i1}, \Delta_{i2})$ for each weight coefficient used in each run, and also show the estimated weight coefficients obtained from the GA program as well as from the defuzzified fuzzy knapsack problem defined in (12)–(14). The comparison of the best profits obtained from the GA approach based on different delta values and those obtained from the fuzzy approach, are shown in Fig. 7. We can see that the proposed GA approach to solve imprecise weight coefficients provided better performance than the defuzzified fuzzy knapsack problem defined in (12)–(14).

Finally, we execute the GA program with $0.2 \leqslant \Delta_{i1}, \Delta_{i2} \leqslant 2.0$ 10 times. Fig. 8 shows a comparison of the best profits obtained from the GA with those obtained from the fuzzy approach. Table 8 lists the actual values of $(\Delta_{i1}, \Delta_{i2})$ for each weight coefficient used in each run, and the estimated weight coefficients obtained from the GA program and the defuzzified fuzzy knapsack problem.

6. Conclusion

This study has investigated the GA approach, via the simulation of fuzzy numbers in some partition points, to solve the knapsack problem with imprecise weight coefficients. Three triangular fuzzy number shapes were considered in this study for analyzing the effectiveness and efficiency of the GA approach. The first type is a symmetrical shape, the second is the right-skewed shape, and the third is

Table 7
The values of $(\Delta_{i1}, \Delta_{i1})$ for each weight coefficient and the estimated weights obtained from the GA and the fuzzy approach in each run

i	1	2	3	4	5	6	7
0.3–1.3 $(\Delta_{i1}, \Delta_{i2})$	(0.762, 0.425)	(0.375, 0.568)	(0.780, 0.517)	(0.445, 0.972)	(0.777, 0.756)	(0.478, 0.306)	(0.429, 0.971)
w_i^* (GA)	30.814	10.021	19.845	29.213	6.085	3.307	6.117
w_i^* (Fuzzy)	30.916	10.048	19.934	29.132	5.995	3.457	6.135
0.4–1.4 $(\Delta_{i1}, \Delta_{i2})$	(0.797, 0.621)	(1.323, 1.225)	(1.397, 1.042)	(0.740, 0.894)	(0.740, 0.628)	(1.137, 1.229)	(1.175, 0.954)
w_i^* (GA)	30.822	9.855	19.888	29.087	5.974	3.542	5.773
w_i^* (Fuzzy)	30.956	9.976	19.911	29.038	5.972	3.523	5.945
0.5–1.5 $(\Delta_{i1}, \Delta_{i2})$	(0.680, 1.499)	(0.801, 0.745)	(1.473, 0.786)	(1.337, 0.770)	(1.037, 0.581)	(1.286, 0.525)	(1.151, 0.529)
w_i^* (GA)	31.241	9.897	19.460	28.749	5.802	3.162	5.819
w_i^* (Fuzzy)	31.205	9.986	19.828	28.038	5.886	3.310	5.845
0.6–1.6 $(\Delta_{i1}, \Delta_{i2})$	(1.582, 1.199)	(1.316, 1.296)	(1.155, 1.299)	(0.830, 1.452)	(1.135, 1.368)	(1.539, 0.759)	(1.366, 0.849)
w_i^* (GA)	30.479	10.203	20.022	29.316	6.040	3.191	5.900
w_i^* (Fuzzy)	30.904	9.995	20.036	29.155	6.058	3.305	5.871
0.7–1.7 $(\Delta_{i1}, \Delta_{i2})$	(1.269, 1.025)	(0.855, 1.461)	(1.679, 1.465)	(1.644, 1.416)	(0.919, 1.208)	(1.678, 1.576)	(1.505, 1.692)
w_i^* (GA)	30.801	10.052	19.707	28.796	6.039	3.476	6.107
w_i^* (Fuzzy)	30.939	10.151	19.946	28.943	6.072	3.475	6.047
0.8–1.8 $(\Delta_{i1}, \Delta_{i2})$	(1.358, 1.485)	(1.471, 0.993)	(1.736, 0.832)	(1.317, 0.811)	(1.192, 0.827)	(0.979, 1.667)	(1.719, 1.077)
w_i^* (GA)	30.949	9.711	19.726	28.745	5.795	3.849	5.656
w_i^* (Fuzzy)	31.032	9.880	19.774	28.874	5.909	3.672	5.840
0.9–1.9 $(\Delta_{i1}, \Delta_{i2})$	(1.558, 1.333)	(0.902, 1.717)	(1.585, 0.903)	(1.110, 1.737)	(1.416, 1.271)	(1.823, 0.946)	(1.354, 0.952)
w_i^* (GA)	30.865	10.434	19.751	29.153	6.033	2.906	5.815
w_i^* (Fuzzy)	30.944	10.204	19.829	29.157	5.964	3.281	5.899

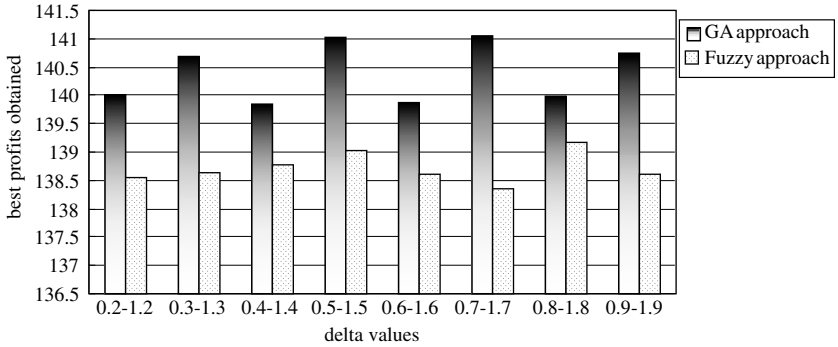

Fig. 7. Comparison of the best profits obtained in eight runs based on different delta values from the GA program and the fuzzy approach.

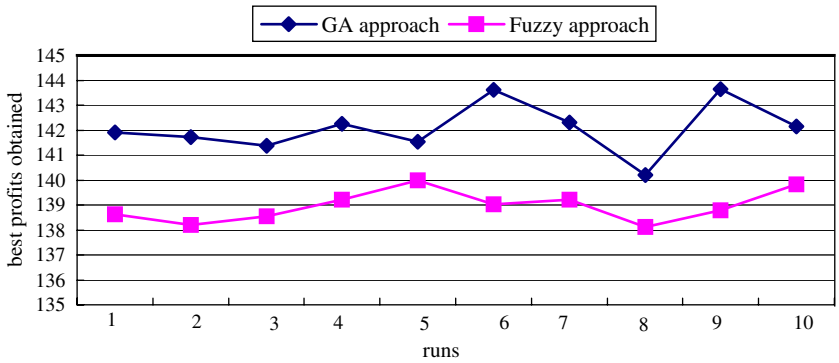

Fig. 8. Comparison of the best profits obtained in 10 runs based on $0.2 \leqslant \Delta_{i1}, \Delta_{i2}, \leqslant 2.0$.

the left-skewed shape. When GAs are applied to solve the fuzzy knapsack problem, the computations need not define the membership function, nor use the extension principle or the interval arithmetic and α-cuts. The empirical results show that the proposed GA approach can obtain better results than that from the defuzzified fuzzy approach defined in (12)–(14). We conclude that the fuzzy GA approach concept is different but gives better results than the traditional fuzzy methods.

Appendix

An essential issue in applying GAs to solve optimization problems is the GA parameter settings for the particular problem. We use a GA program in Example 1 to illustrate how the crossover r_c probability was determined in this research. Since the proposed GA approach simulates a fuzzy number by distributing it into some partition points, we need to choose an adequate crossover rate to evolve the values in each partition point. If we choose an appropriate crossover rate, the GA will try to approximate each partition point to obtain a bell-shaped like discrete fuzzy set for symmetric fuzzy number. We use $w_1 = 8$ in Example 1 of Section 5 and use $\Delta = \Delta_{i1} = \Delta_{i2} = 0.5$ to create 11 partition points, ranging from 7.5 to 8.5, as an instance to illustrate how to choose r_c. Fig. 9 shows the messy curves generated using nine different r_c of the program, starting from 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, to 0.5, respectively. Obviously, none of the curves are bell-shaped. The parameters used in this program are the number of generations 5000, the population size 100, and the probability of mutation 0.003. We then increased the crossover rate starting from 0.55, 0.6, 0.65, to 0.7. The curves for each partition points generated by 0.55, 0.6, 0.65, and 0.7 have a better look than the previous curves appearing in Fig. 9.

We finally increased the crossover rate starting from 0.75, 0.8, 0.85, 0.9, to 0.95 in the test program. The GA creates five approximate bell-shaped curves, which are shown in Fig. 10. Therefore, the

Table 8
The values of $(\Delta_{i1}, \Delta_{i2})$ for each weight coefficient used in each run and the estimated weight coefficients obtained from the GA and the fuzzy approach

$0.2 \leqslant \Delta_1, \Delta_2 \leqslant 2.0$	$i=1$	2	3	4	5	6	7
run#1							
$(\Delta_{i1}, \Delta_{i2})$	(0.200, 0.756)	(1.309, 1.071)	(0.742, 0.311)	(0.200, 1.418)	(0.617, 1.488)	(1.932, 1.156)	(0.265, 1.871)
w_i^* (GA)	31.243	9.750	19.753	29.598	6.377	2.855	6.762
w_i^* (Fuzzy)	31.139	9.940	19.892	29.304	6.218	3.306	6.401
run#2							
$(\Delta_{i1}, \Delta_{i2})$	(1.188, 1.886)	(1.330, 1.896)	(0.200, 0.510)	(0.200, 1.687)	(0.200, 0.611)	(1.119, 0.200)	(2.000, 1.141)
w_i^* (GA)	31.393	10.386	20.128	29.614	6.182	3.050	5.223
w_i^* (Fuzzy)	31.175	10.141	20.077	29.372	6.103	3.270	5.785
run#3							
$(\Delta_{i1}, \Delta_{i2})$	(1.969, 1.486)	(1.776, 0.919)	(0.525, 1.979)	(1.160, 1.419)	(1.406, 1.266)	(1.001, 0.353)	(1.748, 0.543)
w_i^* (GA)	31.011	9.505	20.795	28.898	6.083	3.354	5.597
w_i^* (Fuzzy)	30.879	9.786	20.363	29.065	5.965	3.338	5.699
run#4							
$(\Delta_{i1}, \Delta_{i2})$	(1.264, 0.200)	(1.776, 1.560)	(0.944, 0.345)	(0.557, 1.753)	(0.411, 0.200)	(1.895, 1.109)	(1.920, 0.539)
w_i^* (GA)	30.436	9.987	19.671	29.769	5.858	2.969	5.338
w_i^* (Fuzzy)	30.734	9.946	19.850	29.299	5.947	3.304	5.655
run#5							
$(\Delta_{i1}, \Delta_{i2})$	(2.000, 0.200)	(1.157, 1.845)	(1.174, 0.200)	(1.790, 1.209)	(1.086, 1.007)	(1.269, 1.296)	(1.228, 1.216)
w_i^* (GA)	29.843	10.188	19.547	28.576	6.039	3.376	6.132
w_i^* (Fuzzy)	30.550	10.172	19.756	28.855	5.980	3.507	5.997
run#6							
$(\Delta_{i1}, \Delta_{i2})$	(0.200, 0.200)	(1.815, 0.200)	(1.262, 1.711)	(1.337, 0.470)	(1.140, 0.977)	(0.590, 1.639)	(1.920, 0.976)
w_i^* (GA)	31.000	8.886	20.003	28.686	5.689	4.106	50.656
w_i^* (Fuzzy)	31.000	9.596	20.112	28.783	5.959	3.762	5.764
run#7							
$(\Delta_{i1}, \Delta_{i2})$	(0.744, 0.347)	(1.138, 0.846)	(2.000, 1.343)	(0.626, 0.827)	(1.428, 0.749)	(0.580, 1.127)	(1.676, 1.888)
w_i^* (GA)	30.757	9.939	19.744	29.268	5.675	3.638	6.233
w_i^* (Fuzzy)	30.901	9.927	19.836	29.050	5.830	3.637	6.053
run#8							
$(\Delta_{i1}, \Delta_{i2})$	(0.200, 0.200)	(1.157, 1.941)	(1.338, 1.180)	(0.791, 1.288)	(0.319, 1.020)	(0.287, 0.798)	(0.200, 0.872)
w_i^* (GA)	31.009	10.402	19.893	29.310	6.327	3.780	6.360
w_i^* (Fuzzy)	31.000	10.196	19.961	29.124	6.175	3.628	6.168
run#9							
$(\Delta_{i1}, \Delta_{i2})$	(0.821, 1.057)	(0.263, 1.810)	(1.928, 2.000)	(0.200, 1.989)	(0.265, 1.816)	(1.223, 1.934)	(1.230, 1.312)
w_i^* (GA)	31.8168	10.811	19.876	29.921	6.705	3.567	6.240
w_i^* (Fuzzy)	31.059	10.387	20.018	29.447	6.388	3.675	6.020
run#10							
$(\Delta_{i1}, \Delta_{i2})$	(2.000, 0.712)	(1.137, 0.782)	(2.000, 0.200)	(1.899, 1.564)	(0.930, 1.151)	(1.757, 0.286)	(0.426, 1.029)
w_i^* (GA)	30.367	10.044	19.287	28.877	6.270	2.622	6.385
w_i^* (Fuzzy)	30.678	9.911	19.550	28.916	6.055	3.132	6.151

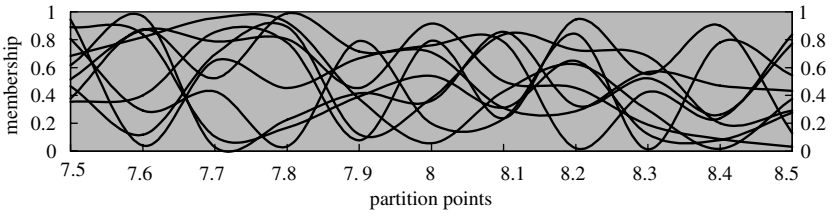

Fig. 9. The messy curves generated using 10 different r_c starting from 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, to 0.5, respectively.

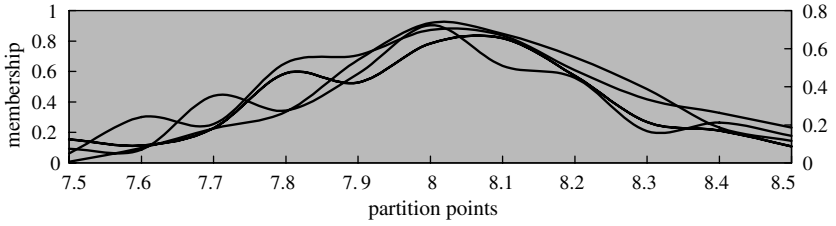

Fig. 10. The bell-shaped like curves obtained from $r_c = 0.75$, 0.8, 0.85, 0.9, and 0.95.

adequate range of crossover rate is in the range of $[7.5, 9.5]$. In this fuzzy knapsack problem, we choose $r_c = 0.9$.

References

[1] J.J. Buckly, Y. Hayashi, Fuzzy genetic algorithm and applications, Fuzzy Sets and Systems 61 (1994) 129–136.
[2] J. Gavish, H. Pirkul, Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality, Mathematical Programming 31 (1985) 78–105.
[3] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[4] F. Herrera, M. Lozano, Fuzzy Genetic Algorithms: Issues and Models, Dept. of Science and A.I., University of Granada, Technical Report No. 18071, Granada, Spain, 1999.
[5] F. Herrera, M. Lozano, J.L. Verdegay, Applying genetic algorithms in fuzzy optimization problems, Fuzzy Sets & Artificial Intelligence 3 (1994) 39–52.
[6] A. Homaifar, E. McCormick, Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms, IEEE Transactions on Fuzzy Systems 3 (2) (1995) 129–139.
[7] C.L. Karr, E.J. Gentry, Fuzzy Control of pH using Genetic Algorithms, IEEE Transactions of Fuzzy Systems 1 (1) (1993) 46–53.
[8] M.A. Lee, H. Takagi, Dynamic control of genetic algorithms using fuzzy logic techniques, in: Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA '93), 1993, pp. 76–83.
[9] F.T. Lin, J.S. Yao, Using fuzzy number in knapsack problem, European Journal of Operational Research 135 (1) (2001) 158–176.
[10] M. Magazine, O. Oguz, A heuristic algorithm for the multidimensional zero-one knapsack problem, European Journal of Operational Research 16 (1984) 319–326.
[11] S. Martello, P. Toth, Knapsack problems: Algorithms and Computer Implementations, John Wiley & Sons Ltd., Chichester, 1990.
[12] S. Martello, P. Toth, Heuristic algorithms for the multiple knapsack problem, Computing 27 (1981) 93–112.
[13] S. Martello, D. Pisinger, P. Toth, New Trend in exact algorithms for the 0-1 knapsack problem, European Journal of Operational Research 123 (2000) 325–332.
[14] S. Okada, M. Gen, Fuzzy multiple choice knapsack problem, Fuzzy Sets and Systems 67 (1994) 71–80.
[15] C.C. Peterson, Computational experience with variants of the Balas algorithm applied to the selection of R&D projects, Management Science 13 (1967) 736–750.
[16] D. Pisinger, An expanding-core algorithm for the exact 0-1 knapsack problem, European Journal of Operational Research 87 (1995) 175–187.
[17] E. Sanchez, T. Shibata, L.A. Zadeh (Eds.), Genetic Algorithms and Fuzzy Logic Systems, Soft Computing Perspectives, World Scientific, 1996.
[18] M. Sakawa, K. Kato, H. Sunada, T. Shibano, Fuzzy programming for multiobjective 0-1 programming problems through revised genetic algorithms, European Journal of Operational Research (97) (1997) 149–158.
[19] Y. Tsujimura, M. Gen, E. Kubota, Solving fuzzy assembly-line balancing problem with genetic algorithms, in: Proceedings of 17th International Conference on Computers and Industrial Engineering, vol. 29, no. 1–4, 1995, pp. 543–547.