Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER Computers and Mathematics with Applications 49 (2005) 1157–1176
An International Journal
computers & mathematics with applications
www.elsevier.com/locate/camwa

Fuzzy Approach to Multilevel Knapsack Problems

HSU-SHIH SHIH
Graduate Institute of Management Science
Tamkang University, Tamsui, Taipei 251, Taiwan, R.O.C.
hshih@mail.tku.edu.tw

(Received and accepted July 2004)

Abstract—This study proposes a fuzzy approach for solving the multiobjective and multilevel knapsack problems (KPs). The problem was first formulated as a multilevel programming problem with multiple decision makers (DMs). Then the degree of satisfaction of each DM was established and represented by their individual membership functions. The recursive formulation of dynamic programming was used to solve the decisions of the interrelated stages. The overall satisfaction of the decision was obtained through this stage-wise operation on the hierarchical structure. Capacity allocation was developed and a step-by-step solution procedure was illustrated. A detailed comparison between multiobjective and multilevel KPs was also carried out. Finally, the possible use of turnpike theorem in KPs was scrutinized in the fuzzy domain.

Keywords—Knapsack problem, Multilevel programming, Multiobjective programming, Dynamic programming, Resource allocation, Turnpike theorem.

1. INTRODUCTION

The well-known knapsack problem (KP) can be summarized as follows: a hiker must decide, among the $i = 1, 2, \ldots, N$ objects, which objects to include in her or his knapsack on a forthcoming trip. Objective i has weight w_i and utility c_i to the hiker. The objective is to maximize the total utility of the hiker's trip subject to a weight limitation, W [1]. This problem has been studied extensively and has been applied to various areas such as capital budgeting, cargo loading, cutting stock, flyaway kit, project selection, etc. [2,3].

During the past decades, there has been an increasing realization on the practical needs to identify and to consider simultaneously several conflicting objectives [4]. This multiobjective knapsack problem (MOKP) can be expressed as [5]

$$\max z_k = \sum_{i=1}^{N} c_i^k x_i, \qquad k = 1, 2, \ldots, K, \tag{1}$$

I would like to thank Professor E. S. Lee at Department of Industrial and Manufacturing Systems Engineering, Kansas State University for his valuable help.

0898-1221/05/$ - see front matter Typeset by AMS-TEX
doi:10.1016/j.camwa.2004.07.025

$$\begin{aligned} &\text{s.t.} \sum_{i=1}^{N} w_i x_i \le W \qquad \text{(capacity constraint)}, \\ &x_i \text{ is integer}, \qquad i = 1, 2, \ldots, N. \end{aligned} \tag{1 (cont.)}$$

Note that the above expression is also classified as the unbounded knapsack problem [6] with one knapsack. If $x_i \in \{0, 1\}$ (i.e., $x_i = 1$, if the object i is selected; and $x_i = 0$, otherwise), the problem becomes a 0-1 KP [2]. Furthermore, the decision of these multiple objectives can be made by a team or by two groups of individual decision makers (DMs) [6].

In a hierarchical organization with multiple DMs, a multilevel knapsack problem (MLKP) can be formulated with decentralized planning. The simplest case of MLKP is the bilevel programming problem, where the top level DM has control over the vector $\mathbf{x}_1$ while the bottom level DM controls the vector $\mathbf{x}_2$. Let the performance functions of z_1 and z_2 for the two planners be linear and bounded, then the new bilevel knapsack problem can be represented as [7]

$$\max_{x1} z_1 = \mathbf{c}_{11}^{\top}\mathbf{x}_1 + \mathbf{c}_{12}^{\top}\mathbf{x}_2 \qquad \text{(upper level)}, \tag{2}$$

where $\mathbf{x}_2$ solves

$$\begin{aligned} &\max_{x2} z_2 = \mathbf{c}_{21}^{\top}\mathbf{x}_1 + \mathbf{c}_{22}^{\top}\mathbf{x}_2 \qquad \text{(lower level)}, \\ &\text{s.t.} \quad (\mathbf{x}_1, \mathbf{x}_2) \in \mathbf{X} = \left\{(\mathbf{x}_1, \mathbf{x}_2) \mid \mathbf{w}_1^{\top}\mathbf{x}_1 + \mathbf{w}_2^{\top}\mathbf{x}_2 \le W, \text{ and } \mathbf{x}_1 \text{ and } \mathbf{x}_2 \text{ are integers}\right\}, \end{aligned}$$

where $\mathbf{c}_{11}$, $\mathbf{c}_{12}$, $\mathbf{c}_{21}$, $\mathbf{c}_{22}$, $\mathbf{w}_1$, and $\mathbf{w}_2$ are vectors, and $\mathbf{X}$ represents the constraint region.

Equation (2) is a nested optimization model involving two problems, an upper one and a lower one [9]. Notice that if there exists no hierarchical control feature, equation (2) simplifies to the MOKP of equation (1). Thus, MOKP and MLKP are closely related. And the constraint regions in two equations are the same, i.e., $\mathbf{X} = \{x_i,\ i = 1, \ldots, N\} = \{\mathbf{x}_1, \mathbf{x}_2\}$.

The usual solution techniques dealing with KPs are dynamic programming (DP) and integer programming [3]. We will focus on DP in this study for the ease of extension to the fuzzy environment. For crisp MOKPs, recent approaches are concentrated on how to find an efficient solution in the DP structure. Cho and Kim [10] developed an improved interactive hybrid method to adjust DM preference information through a scaling constant among objectives. Klamroth and Wiecek [11] proposed DP-based approaches to obtain all the nondominated solutions. Most of other studies can be categorized as integer programming-based approaches. One example is the work of Salman *et al.* [12].

Because of the similar between MOKP and MLKP, these two problems will be imbedded in a common DP structure. We will first review the related literature of fuzzy MODP.

Bellman and Zadeh [13] suggested that fuzzy decisions could be considered as the confluence or intersection of goals and constraints. Esogbue and Bellman [14] made some extensions and introduced many applications. At the same time, Kacprzyk [15] offered a general view about multistage decision-making under fuzziness and derived a general structure for solving fuzzy DP problems. Kacprzyk and Esogbue [16] made a fairly comprehensive survey of the major developments and applications of fuzzy DP. However, the breadth of theory and applications of fuzzy MODP is still limited. This is especially true in the field of decentralized planning of hierarchical systems [17].

Due to the complexity of the multilevel programming (MLP) problems, there exist no efficient traditional techniques for obtaining the numerical solutions of a reasonable sized problem. Shih *et al.* [18] suggest a fuzzy approach for MLP to simplify the complex structure, and it was proven to be feasible and efficient. In addition, the suggested supervised search procedure can be easily extended to a k-level hierarchical system, and it is also a flexible structure for further expansion. Consequently, we would like to examine the possibility of unifying the level-wise (hierarchical) operation and stage-wise operation for multilevel DP in a fuzzy environment. Following an

extension of the structure of Kacprzyk [14], the new structure with interrelation among stages and objectives/levels can be simplified [18,19].

In the remaining sections, we will introduce the concept of capacity allocation for KP. The equations for fuzzy DP and MODM are introduced used to solve the MOKP and MLKP. A discussion on the well-known turnpike theorem with fuzzy approach is also carried out. The final section contains some conclusions and remarks.

2. DYNAMIC PROGRAMMING FOR KNAPSACK PROBLEMS

Dynamic programming is an effective algorithm for solving multistage decision problems. It utilizes a functional equation to circumvent the dimensional explosion for multistage processes [20] and has been applied to solve many real world problems such as optimal control, inventory control, advertising campaign, production planning, equipment replacement, resource allocation, etc. [21]. In this study, we only focus on the knapsack problem.

2.1. Formulation of Dynamic Programming

The knapsack problem paraphrases a general resource allocation model in which a single resource is assigned to a number of alternatives with the objectives of maximizing the total return [2]. Although many techniques could meet the requirement, this problem is ideally suitable for the DP approach [22]. Assume that there is a N-stage process from Stage 1 to Stage N. The total amount of the capacity W is given in the beginning, and we can then set the initial state s^0 equal to the capacity available W for allocation. The process will consume some amount of capacity at each stage, i.e., $s^{l-1} - s^l$ and $l = 1, 2, \ldots, N$. This consumed capacity will make a contribution to the system as an isolated return function or utility R^l or $c_i x_i$, $i = 1, 2, \ldots, N$, at each stage. Note that the number of activities x_i for capacity allocated is the same as the number of the stages here. Furthermore, the state variable s^j will represent the amount of the capacity remaining at Stage j, and it is ready for the allocation from Stage j to Stage N (the last stage). We use W^j as the consumed size before Stage j usually; thus, $0 \leq s^j \leq W - W^j$. In practice, we can leave some capacity unallocated, but we cannot utilize more than the initial amount of the capacity at any single stage. In addition, the state variable equation at each stage is kept as the same as in a DP structure. Meanwhile, the different policy control (variables) at each stage forms a feasible space in the DP structure. Following the computational procedure, the consumed capacity at any stage will yield a contribution for its objective, and these contributions will be accumulated to the total return (i.e., return function $g^l(s^{l-1})$ with the stage $l = 1, 2, \ldots, N$) in the end. Consequently, the DP formulation for a KP will be carried out by accumulated capacity returns R^l through various stages

$$\begin{aligned} g^l\left(s^{l-1}\right) &= \max_{kl}\left[R^l + g^{l+1}\left(s^l\right)\right], \qquad l = 1, 2, \ldots, N-1, \quad \text{and} \\ g^N\left(s^{N-1}\right) &= \max_{kN}\left[R^N\right], \end{aligned} \tag{3}$$

s.t.

$$\begin{aligned} u^l(k^l) &= s^{l-1} - s^l, \qquad l = 1, 2, \ldots, N \quad \text{(state variable equation)}, \\ k^l &\in \mathbf{F} \qquad \text{(feasible region for policy control variables)}, \\ s^l &\in \mathbf{S} \qquad \text{(capacity availability constraint)}, \\ s^0 &= W \qquad \text{(initial resource restriction)}, \\ & s^0 \text{ and } s^l \text{ are positive integers}, \qquad l = 1, 2, \ldots, N, \end{aligned}$$

where $\mathbf{F} = \{\mathbf{F}^l,\ l = 1, 2, \ldots, N\}$ is a feasible region for all possible policy control variables in all stages, and $\mathbf{S}$ is a resource availability set which includes budget, capacity, and other constraints

throughout the stages, i.e., $\mathbf{S} = \{(0 \leq s^l \leq W - W^l),\ l = 1, 2, \ldots, N\}$. In addition, the objective function $g^l(s^{l-1})$ is a recursive expression at each stage except the last Stage N.

If the types of resources are more than one, we need more than one state variable to describe the resource utilization. However, we frequently take a weight capacity as the only one type of resources for simplicity. Moreover, it is a discrete resource allocation problem if the resource is discrete or integer [23,24]. Since the number of stage is finite, it can be categorized as a discrete (time) or a finite-stage deterministic DP problem.

2.2. Fuzzy Multiobjective Dynamic Programming

In a DP structure, the state variable equation will implicitly transfer a policy control variable and an input state to an output state at each stage, where the constraint is imposed on policy control variable and the goal is imposed on the output state. For a specific Stage j, an input state s^{j-1} will be applied to a policy control variable k^j, which is subjected to a fuzzy constraint $\mu_{Cj}(k^j)$, and proceeded to an output state s^j, on which fuzzy goal $\mu_{Gj}(s)$ is imposed, through some known cause/effect relationship as shown in Figure 1. Fuzzy decision will become an aggregation of the fuzzy constraint and fuzzy goal at a particular stage; however, the decision will be affected by the decisions of other stages. The optimality of fuzzy multistage decision making will express how well the subsequent fuzzy constraints and fuzzy goals are satisfied by the policy controls and states, respectively. Mathematically, the fuzzy decision in the dynamic environment can be represented as the following expression [15].

$$\mu\left(k^1, \ldots, k^{(N-1)} \mid s^0\right) = \mu_{C1}\left(k^1\right) \otimes \cdots \otimes \mu_{CN}\left(k^N\right) \otimes \mu_{GN}\left(s^N\right), \tag{4}$$

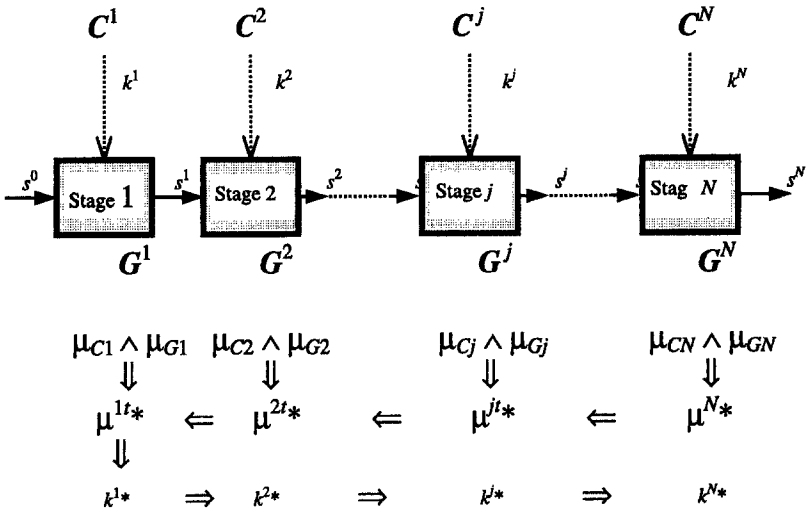

Figure 1. A Fuzzy approach for the structure of dynamic programming.

Note:

(1) An N-stage processing diagram is starting from Stage 1 to Stage N represented by each solid line with an arrow.
(2) State variable s^0 is the initial state and the input of Stage 1 as well; state variable s^1 is the output of Stage 1 and the input of Stage 2 as well; the rest are defined in the same order.
(3) Policy control variable k^1 represents the possible alternatives at Stage 1 and the rest are defined in the same order. In addition, the influence of the policy control variable on each stage is conveyed by the dotted line with an arrow.
(4) $\mathbf{C}^j$ and $\mathbf{G}^j$ are depicted as the constraint and goal, respectively, at Stage j.
(5) The constraint is imposed on policy control and the goal is imposed on output state at each stage. And the decision is the confluence of constraints and goals at each stage, and is accumulated backward. Thus, the total maximum satisfaction at that point μ^{jt*} will present the current accumulated degree of satisfaction from Stage $j = N$ backward to Stage j.

where $\otimes$ is a fuzzy operator. The problem is to find an optimal sequence of control variables $k^{1*}, \ldots, k^{N-1*}$, such as

$$\mu\left(k^{1*}, \ldots, k^{(N-1)*} \mid s^0\right) = \max_{k1,\ldots,kN} \left[\mu_{C1}\left(k^1\right) \otimes \cdots \otimes \mu_{CN}\left(k^N\right) \otimes \mu_{GN}\left(s^N\right)\right].$$

Furthermore, a group of recursive equations can be obtained for the backward iteration at a specified stage.

$$\begin{aligned} \mu_{Gh}\left(s^{(j-1)}\right) &= \max_{kj}\left[\mu_{Cj}\left(k^j\right) \otimes \mu_{G(j+1)}\left(s^j\right)\right], \qquad \text{and} \\ s^j &= g\left(s^{(j-1)}, k^j\right), \qquad j = 1, \ldots, N, \end{aligned} \tag{5}$$

where $\mu_{Gj}(s^{(j-1)})$ can be regarded as the fuzzy goal at Stage j induced by the fuzzy goal at Stage $j+1$.

Observe that the aggregation of fuzzy constraints and fuzzy objective functions will depend on max-min operation [13]. Other fuzzy operators would be also effective, e.g., compensatory operator [17], product operator [24].

After the structure is defined, an imbedded MODM problem will be processed in a fuzzy domain. The MOKP is defined as follows: W units of capacity are to be distributed among N activities to maximize K objectives. There are also given $T \times M$ data tables $f_{tm}(x)$ $(t = 1, 2, \ldots, N;$ $m = 1, 2, \ldots, K)$ representing the return which is realized by the j^{th} objective from an allocation of x units of capacity to the t^{th} activity $(x = 0, 1, 2, \ldots, W)$. The problem is to allocate W units of capacity to the N activities, such that the returns of K objectives from the activities are maximized simultaneously [25].

Despite the wide range of applications, the literature on the MOKP is still limited. Hence, we will survey the literature on MODM in this section, which is in a broad sense. Most of the fuzzy approaches for MODM problems will be classified as scalar-objective and generating methods which are the same categories of the approaches for MODP. Hussein and Abo-Sinna [26] present an approach for scalarizing the multiple objectives to be a synthetic objective so that the problem can be solved in a DP structure. Lai and Li [25] propose a method which makes marginal evaluation for each single objective and the method then works for global evaluation for multiple objectives by their relative important in DM's mind. And they further extend the approach including linguistic variables and the quantitative objectives [27]. It seems that the former is difficult to catch the decision information and the latter involves too much computation so far. It is worthy of making further development.

According to Salukvadze's consideration of ideal point [28] and the comment of Li and Haimes [29], we introduce the concept of (positive) ideal solution (PIS) and negative ideal solution (NIS) [30] to overcome the drawback. The PIS and NIS are used for defining the possible optimistic outcome and pessimistic outcome, respectively. Then the degree of satisfaction of DMs can be established in terms of fuzzy membership functions. The imbedded MODM problem at each stage is to seek a maximum degree of satisfaction through fuzzy operation under each control. Afterwards, these temporal results are accumulated in the structure of multistage decision-making process under fuzziness [15]. Note that we will see a slight difference between a resource allocation model [17] and the knapsack problem due to the KP has only one PIS/NIS over stages.

The process to solve the imbedded MODM problem is analogous to the concept of global criterion in MODM (see [31]), but the decision space of the former is discrete. Nevertheless, we can obtain the sufficient information in an efficient way so that the solution will be much meaningful and the procedure is straightforward.

It is noted that we could consider Bellman and Zadeh's max-min operation [13] for the imbedded fuzzy MODM problems as well as the stage-wise DP problem here. There is degree of satisfaction at each stage representing the compromise of MODM when the input state will be applied to the

policy control variable, which is subjected to some fuzzy constraints at that stage. Since it is unknown for the exact amount budget at any specific stage, the backward procedure of DP will help us enumerate all temporary optimal cases under available budget at each intermediate stage. When the process is reached to the first stage, the optimum of global evaluation is obtained. Then we will trace forward from Stage 1 to Stage N to pick up the optimal policy controls among stages and their correspondent optimal decisions among levels. These results will be the global optimum because all temporary optimal solutions are checked on the stage-wise and level-wise operations. Consequently, the MODM over various stages could be manipulated in the designed way.

As depicted in Figure 1, the solution procedure of fuzzy approach for MODM over various stages for MOKPs is proposed as the following six steps for the backward recursion.

STEP 1. PREPARATION PHASE. Obtain reference information.

(1a) Construct a payoff table for each objective of the DM(s) based on the possible capacity allocated under policy control (variables).

(1b) Establish the degree of satisfaction for each objective in terms of fuzzy membership function through the distance between his/her PIS and NIS at each stage.

STEP 2. FORWARD PHASE. Seek compromise solutions among multiple objectives for possible capacity allocated at each stage.

Take minimum operation of the degree of satisfaction of multiple objectives as DM's decisions for possible budgets allocated at each stage.

STEP 3. Choose optimal decisions of MODM.

Take maximum operation of the possible outcomes, from Step 1, given by each specified capacity at each stage as its optimal decision of MODM.

STEP 4. BACKTRACKING PHASE. Process of the last stage N^{th} problem.

Record all possible noninferior outcomes under available budgets at the last stage as the current results from Step 2 and, for further backward calculation.

STEP 5. Process the previous stage $(N-1)^{\text{th}}$ problem.

The degree of satisfaction of Stage N is aggregated together with the previous Stage $(N-1)$s as a current accumulated degree of satisfaction. Take minimum operation of the results from Stage N and the recorded results from Stage $N-1$ (obtained from Step 2) under available budgets for these two successive stages. Then, take maximum operator of the possible outcomes for the combinations under policy control (variables) for these two stages. And the maximum degree of satisfaction is utilized as a current accumulated optimal decisions. Set $N = N-1$.

STEP 6. TERMINATION.

Under the condition of $N = 1$, the process is located at the beginning and terminated. Check the optimal solution for each stage under a specified policy, which is suggested as the group of best policies for the MOKP in the dynamic environment. Otherwise, go to Step 4.

The above procedure can divide into three phases: preparation, forward, and backtracking phases. The first one is to get decision information in each stage. The rest is the similar process as the crisp DP process. Observe that the forward process in DP structure could also be applicable to the above procedure.

Let us illustrate the proposed procedure through an example.

EXAMPLE 1. A bi-objective knapsack problem.

$$\begin{aligned}
&\max\ z_1 = 4x_1 + 7x_2 + 2x_3 + x_4 && \text{(objective 1)},\\
&\max\ z_2 = 2x_1 + x_2 + 4x_3 + 3x_4 && \text{(objective 2)},\\
&\text{s.t.}\quad 2x_1 + 2x_2 + x_3 + x_4 \le 13,\\
&\qquad x_1,\ x_2,\ x_3, \text{ and } x_4 \text{ are positive integers.}
\end{aligned}$$

Table 1. The degree of satisfaction between two objectives under the capacity available for x_1 at Stage 1.

Policy Number k^1	Capacity Available s^0	Activity Acquired x_1	Capacity Left	Return Functions R_1^1	Return Functions R_2^1	Fuzzy Goals μ_{R11}	Fuzzy Goals μ_{R21}	$\min(\mu_{R11}, \mu_{R21})$	$\mu^{1*} = \max - \min(\mu_{R11}, \mu_{R21})$	Remarks (Best Policy)
1	0	0	0	0.00#	0.00#	0.000	0.000	0.000	0.000	1
2	1	0	1	0.00#	0.00#	0.000	0.000	0.000	0.000	2
3	2	0	2	0.00	0.00	0.000	0.000	0.000		
4		1	0	4.00	2.00	0.091	0.038	0.038	0.038	4
5	3	0	3	0.00	0.00	0.000	0.000	0.000		
6		1	1	4.00	2.00	0.091	0.038	0.038	0.038	6
7	4	0	4	0.00	0.00	0.000	0.000	0.000		
8		1	2	4.00	2.00	0.091	0.038	0.038		
9		2	0	8.00	4.00	0.182	0.077	0.077	0.077	9
10	5	0	5	0.00	0.00	0.000	0.000	0.000		
11		1	3	4.00	2.00	0.091	0.038	0.038		
12		2	1	8.00	4.00	0.182	0.077	0.077	0.077	12
			(The middle part of the table is omitted.)							
37	11	0	11	0.00	0.00	0.000	0.000	0.000		
38		1	9	4.00	2.00	0.091	0.038	0.038		
39		2	7	8.00	4.00	0.182	0.077	0.077		
40		3	5	12.00	6.00	0.273	0.115	0.115		
41		4	3	16.00	8.00	0.364	0.154	0.154		
42		5	1	20.00	10.00	0.455	0.192	0.192	0.192	42
43	12	0	12	0.00	0.00	0.000	0.000	0.000		
44		1	10	4.00	2.00	0.091	0.038	0.038		
45		2	8	8.00	4.00	0.182	0.077	0.077		
46		3	6	12.00	6.00	0.273	0.115	0.115		
47		4	4	16.00	8.00	0.364	0.154	0.154		
48		5	2	20.00	10.00	0.455	0.192	0.192		
49		6	0	24.00&	12.00&	0.545	0.231	0.231	0.231	49
50	13	0	13	0.00	0.00	0.000	0.000	0.000		
51		1	11	4.00	2.00	0.091	0.038	0.038		
52		2	9	8.00	4.00	0.182	0.077	0.077		
53		3	7	12.00	6.00	0.273	0.115	0.115		
54		4	5	16.00	8.00	0.364	0.154	0.154		
55		5	3	20.00	10.00	0.455	0.192	0.192		
56		6	1	24.00&	12.00&	0.545	0.231	0.231	0.231	56

Note:

(1) The capacity allocated to the imbedded bi-objective problem at Stage 1 is designated by w^1, or x_1, and left with $s^0 - x_1$.

(2) At this stage, return function $R_1^1 = 4x_1$ is for the first objective and $R_2^1 = 2x_1$ is for the second objective.

(3) "&" and "#" indicate the PIS and the NIS of the objectives, respectively.

(4) μ^{1*} denotes the maximum satisfaction under a given capacity W^1 at Stage 1 only.

(5) The constraint for each policy and inside restriction is implicitly represented in the above table.

Table 2. The degree of satisfaction between two objectives under the capacity available for x_2 at Stage 2.

Policy Number k^2	Capacity Available s^1	Activity Acquired x_2	Capacity Left	Return Functions R_1^2	Return Functions R_2^2	Fuzzy Goals μ_{R12}	Fuzzy Goals μ_{R22}	$\min(\mu_{R12}, \mu_{R22})$	$\mu^{2*} = \max - \min(\mu_{R12}, \mu_{R22})$	Remarks (Best Policy)
1	0	0	0	0.00#	0.00#	0.000	0.000	0.000	0.000	1
2	1	0	1	0.00#	0.00#	0.000	0.000	0.000	0.000	2
3	2	0	2	0.00	0.00	0.000	0.000	0.000		
4		1	0	7.00	1.00	0.159	0.019	0.019	0.019	4
5	3	0	3	0.00	0.00	0.000	0.000	0.000		
6		1	1	7.00	1.00	0.159	0.019	0.019	0.019	6
7	4	0	4	0.00	0.00	0.000	0.000	0.000		
8		1	2	7.00	1.00	0.159	0.019	0.019		
9		2	0	14.00	2.00	0.318	0.038	0.038	0.038	9
10	5	0	5	0.00	0.00	0.000	0.000	0.000		
11		1	3	7.00	1.00	0.159	0.019	0.019		
12		2	1	14.00	2.00	0.318	0.038	0.038	0.038	12
			(The middle part of the table is omitted.)							
37	11	0	11	0.00	0.00	0.000	0.000	0.000		
38		1	9	7.00	1.00	0.159	0.019	0.019		
39		2	7	14.00	2.00	0.318	0.038	0.038		
40		3	5	21.00	3.00	0.477	0.058	0.058		
41		4	3	28.00	4.00	0.636	0.077	0.077		
42		5	1	35.00	5.00	0.795	0.096	0.096	0.096	42
43	12	0	12	0.00	0.00	0.000	0.000	0.000		
44		1	10	7.00	1.00	0.159	0.019	0.019		
45		2	8	14.00	2.00	0.318	0.038	0.038		
46		3	6	21.00	3.00	0.477	0.058	0.058		
47		4	4	28.00	4.00	0.636	0.077	0.077		
48		5	2	35.00	5.00	0.795	0.096	0.096		
49		6	0	42.00&	6.00&	0.955	0.115	0.115	0.115	49
50	13	0	13	0.00	0.00	0.000	0.000	0.000		
51		1	11	7.00	1.00	0.159	0.019	0.019		
52		2	9	14.00	2.00	0.318	0.038	0.038		
53		3	7	21.00	3.00	0.477	0.058	0.058		
54		4	5	28.00	4.00	0.636	0.077	0.077		
55		5	3	35.00	5.00	0.795	0.096	0.096		
56		6	1	42.00&	6.00&	0.955	0.115	0.115	0.115	56

Note:

(1) The capacity allocated to the imbedded bi-objective problem at Stage 1 is designated by W^2, or x_2, and left with $s^1 - x_2$.

(2) At this stage, return function $R_1^2 = 7x_2$ is for the first objective and $R_2^2 = x_2$ is for the second objective.

(3) "&" and "#" indicate the PIS and the NIS of the objectives, respectively.

(4) μ^{2*} denotes the maximum satisfaction under a given capacity W^2 at Stage 2 only.

(5) The constraint for each policy and inside restriction is implicitly represented in the above table.

Table 3. The degree of satisfaction between two objectives under the capacity available for x_3 at Stage 3.

Policy Number k^3	Capacity Available s^2	Activity Acquired x_3	Capacity Left	Return Functions R_1^3	R_2^3	Fuzzy Goals μ_{R13}	μ_{R23}	$\min(\mu_{R13}, \mu_{R23})$	$\mu^{3*} = \max - \min(\mu_{R13}, \mu_{R23})$	Remarks (Best Policy)
1	0	0	0	0.00#	0.00#	0.000	0.000	0.000	0.000	1
2	1	0	1	0.00	0.00	0.000	0.000	0.000		
3		1	0	2.00	4.00	0.045	0.077	0.045	0.045	3
4	2	0	2	0.00	0.00	0.000	0.000	0.000		
5		1	1	2.00	4.00	0.045	0.077	0.045		
6		2	0	4.00	8.00	0.091	0.154	0.091	0.091	6
7	3	0	3	0.00	0.00	0.000	0.000	0.000		
8		1	2	2.00	4.00	0.045	0.077	0.045		
9		2	1	4.00	8.00	0.091	0.154	0.091		
10		3	0	6.00	12.00	0.136	0.231	0.136	0.136	10
			(The middle part of the table is omitted.)							
67	11	0	11	0.00	0.00	0.000	0.000	0.000		
68		1	10	2.00	4.00	0.045	0.077	0.045		
69		2	9	4.00	8.00	0.091	0.154	0.091		
70		3	8	6.00	12.00	0.136	0.231	0.136		
71		4	7	8.00	16.00	0.182	0.308	0.182		
72		5	6	10.00	20.00	0.227	0.385	0.227		
73		6	5	12.00	24.00	0.273	0.462	0.273		
74		7	4	14.00	28.00	0.318	0.538	0.318		
75		8	3	16.00	32.00	0.364	0.615	0.364		
76		9	2	18.00	36.00	0.409	0.692	0.409		
77		10	1	20.00	40.00	0.455	0.769	0.455		
78		11	0	22.00	44.00	0.500	0.846	0.500	0.500	78
79	12	0	12	0.00	0.00	0.000	0.000	0.000		
80		1	11	2.00	4.00	0.045	0.077	0.045		
81		2	10	4.00	8.00	0.091	0.154	0.091		
82		3	9	6.00	12.00	0.136	0.231	0.136		
83		4	8	8.00	16.00	0.182	0.308	0.182		
84		5	7	10.00	20.00	0.227	0.385	0.227		
85		6	6	12.00	24.00	0.273	0.462	0.273		
86		7	5	14.00	28.00	0.318	0.538	0.318		
87		8	4	16.00	32.00	0.364	0.615	0.364		
88		9	3	18.00	36.00	0.409	0.692	0.409		
89		10	2	20.00	40.00	0.455	0.769	0.455		
90		11	1	22.00	44.00	0.500	0.846	0.500		
91		12	0	24.00	48.00	0.545	0.923	0.545	0.545	91
92	13	0	13	0.00	0.00	0.000	0.000	0.000		
93		1	12	2.00	4.00	0.045	0.077	0.045		
94		2	11	4.00	8.00	0.091	0.154	0.091		
95		3	10	6.00	12.00	0.136	0.231	0.136		
96		4	9	8.00	16.00	0.182	0.308	0.182		
97		5	8	10.00	20.00	0.227	0.385	0.227		
98		6	7	12.00	24.00	0.273	0.462	0.273		

Table 3. (cont.)

Policy Number k^3	Capacity Available s^2	Activity Acquired x_3	Capacity Left	Return Functions R_1^3	Return Functions R_2^3	Fuzzy Goals μ_{R13}	Fuzzy Goals μ_{R23}	$\min(\mu_{R13}, \mu_{R23})$	$\mu^{3*} = \max - \min(\mu_{R13}, \mu_{R23})$	Remarks (Best Policy)
99		7	6	14.00	28.00	0.318	0.538	0.318		
100		8	5	16.00	32.00	0.364	0.615	0.364		
101		9	4	18.00	36.00	0.409	0.692	0.409		
102		10	3	20.00	40.00	0.455	0.769	0.455		
103		11	2	22.00	44.00	0.500	0.846	0.500		
104		12	1	24.00	48.00	0.545	0.923	0.545		
105		13	0	26.00&	52.00&	0.591	1.000	0.591	0.591	105

Note:

(1) The capacity allocated to the imbedded bi-objective problem at Stage 3 is designated by W^3, or x_3, and left with $s^2 - x_3$.

(2) At this stage, return function $R_1^3 = 2x_3$ is for the first objective and $R_2^3 = 4x_3$ is for the second objective.

(3) "&" and "#" indicate the PIS and the NIS of the objectives, respectively.

(4) μ^{3*} denotes the maximum satisfaction under a given capacity W^3 at Stage 3 only.

(5) The constraint for each policy and inside restriction is implicitly represented in the above table.

There is exactly one knapsack with 13 pounds available for the hiker team in the future. The decisions are controlled by the team with two different objectives, z_1 and z_2. The whole decision process will last for four periods, and shall be executed at four stages. For a given Stage l, total capacity s^{l-1} is allocated for investing Stage l and the rest of stages. However, only W^l amount of capacity is available for the imbedded bi-objective returns. The team can leave some units of the capacity unallocated, but they cannot spend more than the given capacity at any stage. The investments will yield the objective increases as $z_1^l(\mathbf{x}_i)$ and $z_2^l(\mathbf{x}_i)$ as the utilities or performance measures at each stage, $l = 1, 2, 3$, and 4. Therefore, the imbedded bi-objective problem will maximize its objectives at each stage, and can be described as the following expression under the initial state s^0.

For Stage 1,

$$\max R_1^1 = 4x_1 \quad \text{and} \quad \max R_2^1 = 2x_1.$$

For Stage 2,

$$\max R_1^2 = 7x_2 \quad \text{and} \quad \max R_2^2 = x_2.$$

For Stage 3,

$$\max R_1^3 = 2x_3 \quad \text{and} \quad \max R_2^3 = 4x_3.$$

And for Stage 4,

$$\max R_1^4 = x_4 \quad \text{and} \quad \max R_2^4 = 3x_2.$$

Here at each stage the imbedded bi-objective return function $\mathbf{R}^l = (R_1^l, R_2^l)$ is to give the optimal goals in DP structure under the input state and the control variable. In addition, the available budget W^l is allocated to the imbedded returns only at Stage l, where $W^l = s^{l-1} - s^l$, $l = 1, 2, 3$, and 4, under DP structure. Note that all above constraints can be represented by a set $\mathbf{Y}$.

All decision information is listed in Tables 1–4 for four different stages, and the constraint for the possible policies is also implicitly represented. Now we solve the problem through the proposed procedure.

STEP 1. According to the basic information in these tables, we will set up the degree of satisfaction by fuzzifying the objectives through their PIS and NIS of the possible capacity allocated

Table 4. The degree of satisfaction between two objectives under the capacity available for x_4 at Stage 4.

Policy Number k^4	Capacity Available s^3	Activity Acquired x_4	Capacity Left	Return Functions R_1^4	Return Functions R_2^4	Fuzzy Goals μ_{R14}	Fuzzy Goals μ_{R24}	$\min(\mu_{R14}, \mu_{R24})$	$\mu^{4*} = \max - \min(\mu_{R14}, \mu_{R24})$	Remarks (Best Policy)
1	0	0	0	0.00#	0.00#	0.000	0.000	0.000	0.000	1
2	1	0	1	0.00	0.00	0.000	0.000	0.000		
3		1	0	1.00	3.00	0.023	0.058	0.023	0.023	3
4	2	0	2	0.00	0.00	0.000	0.000	0.000		
5		1	1	1.00	3.00	0.023	0.058	0.023		
6		2	0	2.00	6.00	0.045	0.115	0.045	0.045	6
7	3	0	3	0.00	0.00	0.000	0.000	0.000		
8		1	2	1.00	3.00	0.023	0.058	0.023		
9		2	1	2.00	6.00	0.045	0.115	0.045		
10		3	0	3.00	9.00	0.068	0.173	0.068	0.068	10
			(The middle part of the table is omitted.)							
67	11	0	11	0.00	0.00	0.000	0.000	0.000		
68		1	10	2.00	4.00	0.045	0.077	0.045		
69		2	9	4.00	8.00	0.091	0.154	0.091		
70		3	8	6.00	12.00	0.136	0.231	0.136		
71		4	7	8.00	16.00	0.182	0.308	0.182		
72		5	6	10.00	20.00	0.227	0.385	0.227		
73		6	5	12.00	24.00	0.273	0.462	0.273		
74		7	4	14.00	28.00	0.318	0.538	0.318		
75		8	3	16.00	32.00	0.364	0.615	0.364		
76		9	2	18.00	36.00	0.409	0.692	0.409		
77		10	1	20.00	40.00	0.455	0.769	0.455		
78		11	0	22.00	44.00	0.500	0.846	0.500	0.500	78
79	12	0	12	0.00	0.00	0.000	0.000	0.000		
80		1	11	2.00	4.00	0.045	0.077	0.045		
81		2	10	4.00	8.00	0.091	0.154	0.091		
82		3	9	6.00	12.00	0.136	0.231	0.136		
83		4	8	8.00	16.00	0.182	0.308	0.182		
84		5	7	10.00	20.00	0.227	0.385	0.227		
85		6	6	12.00	24.00	0.273	0.462	0.273		
86		7	5	14.00	28.00	0.318	0.538	0.318		
87		8	4	16.00	32.00	0.364	0.615	0.364		
88		9	3	18.00	36.00	0.409	0.692	0.409		
89		10	2	20.00	40.00	0.455	0.769	0.455		
90		11	1	22.00	44.00	0.500	0.846	0.500		
91		12	0	24.00	48.00	0.545	0.923	0.545	0.545	91
92	13	0	13	0.00	0.00	0.000	0.000	0.000		
93		1	12	2.00	4.00	0.045	0.077	0.045		
94		2	11	4.00	8.00	0.091	0.154	0.091		
95		3	10	6.00	12.00	0.136	0.231	0.136		
96		4	9	8.00	16.00	0.182	0.308	0.182		
97		5	8	10.00	20.00	0.227	0.385	0.227		
98		6	7	12.00	24.00	0.273	0.462	0.273		

Table 4. (cont.)

Policy Number k^4	Capacity Available s^3	Activity Acquired x_4	Capacity Left	Return Functions R_1^4	R_2^4	Fuzzy Goals μ_{R14}	μ_{R24}	$\min(\mu_{R14}, \mu_{R24})$	$\mu^{4*} = \max - \min(\mu_{R14}, \mu_{R24})$	Remarks (Best Policy)
99		7	6	14.00	28.00	0.318	0.538	0.318		
100		8	5	16.00	32.00	0.364	0.615	0.364		
101		9	4	18.00	36.00	0.409	0.692	0.409		
102		10	3	20.00	40.00	0.455	0.769	0.455		
103		11	2	22.00	44.00	0.500	0.846	0.500		
104		12	1	24.00	48.00	0.545	0.923	0.545		
105		13	0	26.00&	52.00&	0.591	1.000	0.591	0.591	105

Note:

(1) The capacity allocated to the imbedded bi-objective problem at Stage 4 is designated by W^4, or x_4, and left with $s^3 - x_4$.

(2) At this stage, return function $R_1^4 = x_4$ is for the first objective and $R_2^4 = 3x_4$ is for the second objective.

(3) "&" and "#" indicate the PIS and the NIS of the objectives, respectively.

(4) μ^{4*} denotes the maximum satisfaction under a given capacity W^4 at Stage 4 only.

(5) The constraint for each policy and inside restriction is implicitly represented in the above table.

Table 5. All possible noninferior outcomes under the total available capacity at Stage 4.

Available Capacity s^3	Activity x_4	Capacity Left	Maximized Satisfaction μ^{4*}	Optimal Policy k^4	Remarks
0	0	0	0.000	1	
1	1	0	0.023	3	
2	2	0	0.045	6	**
3	3	0	0.068	10	
4	4	0	0.091	15	
5	5	0	0.114	21	
6	6	0	0.136	28	
7	7	0	0.318	36	
8	8	0	0.364	45	
9	9	0	0.409	55	
10	10	0	0.455	66	
11	11	0	0.500	78	
12	12	0	0.545	91	
13	13	0	0.591	105	

Note:

(1) The available budget to the imbedded MODM is s^3, the input state.

(2) The optimal policy for Stage 4 corresponds to the maximum satisfaction from Table 4.

(3) ** (in Remarks) indicates the best policy to be chosen.

at each stage. Here PIS $z_1^+ = 44$ and $z_2^+ = 52$ are obtained from traditional DP process, and assuming NIS $z_1^- = z_2^- = 0$ for capacity unallocated.

STEP 2. The satisfactory solution of the imbedded bi-objective returns at each stage can be obtained through max-min operation, which means minimizing the satisfactory degree of two objectives, and then go to the next step.

Table 6 All possible noninferior outcomes under the total available capacity at Stage 3.

Available Capacity s^2	Activity x_3	Capacity Left	Degree of Satisfaction μ_{R13}	μ_{R23}	$\mu^{3*} = \min(\mu_{R13}, \mu_{R23})$	Max. Satisfaction from Stage 4 μ^{4*}	Accumulated Satisfaction μ^{3t*}	Optimal Policy k^3	Remarks
0	0	0	0.000	0.000	0.000	0.000	0.000		
1	1	0	0.045	0.077	0.045	0.000	0.000		
2	2	0	0.091	0.154	0.091	0.000	0.000		
3	3	0	0.136	0.231	0.136	0.000	0.000		
4	4	0	0.182	0.308	0.182	0.000	0.000		
5	5	0	0.227	0.385	0.227	0.000	0.000		
6	6	0	0.273	0.462	0.273	0.000	0.000		
7	7	0	0.318	0.538	0.318	0.000	0.000		
8	8	0	0.364	0.615	0.364	0.000	0.000		
9	9	0	0.409	0.692	0.409	0.000	0.000		
10	10	0	0.455	0.769	0.455	0.000	0.000		
11	11	0	0.500	0.846	0.500	0.000	0.000		
12	12	0	0.545	0.923	0.545	0.000	0.000		
13	13	0	0.591	1.000	0.591	0.000	0.000		
1	0	1	0.000	0.000	0.000	0.023			
2	1	1	0.045	0.077	0.045	0.023	0.023	5	
3	2	1	0.091	0.154	0.091	0.023	0.023	9	
4	3	1	0.136	0.231	0.136	0.023	0.023	14	
5	4	1	0.182	0.308	0.182	0.023	0.023	20	
6	5	1	0.227	0.385	0.227	0.023	0.023	27	
7	6	1	0.273	0.462	0.273	0.023	0.023	35	
8	7	1	0.318	0.538	0.318	0.023	0.023	44	
9	8	1	0.364	0.615	0.364	0.023	0.023	54	
10	9	1	0.409	0.692	0.409	0.023	0.023	65	
11	10	1	0.455	0.769	0.455	0.023	0.023	77	
12	11	1	0.500	0.846	0.500	0.023	0.023	90	
13	12	1	0.545	0.923	0.545	0.023	0.023	104	
2	0	2	0.000	0.000	0.000	0.045			
3	1	2	0.045	0.077	0.045	0.045	0.045	8	**
4	2	2	0.091	0.154	0.091	0.045	0.045	13	
5	3	2	0.136	0.231	0.136	0.045	0.045	19	
6	4	2	0.182	0.308	0.182	0.045	0.045	26	
7	5	2	0.227	0.385	0.227	0.045	0.045	34	
8	6	2	0.273	0.462	0.273	0.045	0.045	43	
9	7	2	0.318	0.538	0.318	0.045	0.045	53	
10	8	2	0.364	0.615	0.364	0.045	0.045	64	
11	9	2	0.409	0.692	0.409	0.045	0.045	76	
12	10	2	0.455	0.769	0.455	0.045	0.045	81	
13	11	2	0.500	0.846	0.500	0.045	0.045	94	
3	0	3	0.000	0.000	0.000	0.068			
4	1	3	0.045	0.077	0.045	0.068	0.045	12	
5	2	3	0.091	0.154	0.091	0.068	0.068	18	
6	3	3	0.136	0.231	0.136	0.068	0.068	25	
7	4	3	0.182	0.308	0.182	0.068	0.068	33	

Table 6. (cont.)

Available Capacity s^2	Activity x_3	Capacity Left	Degree of Satisfaction μ_{R13}	μ_{R23}	$\mu^{3*} = \min(\mu_{R13}, \mu_{R23})$	Max. Satisfaction from Stage 4 μ^{4*}	Accumulated Satisfaction μ^{3t*}	Optimal Policy k^3	Remarks
8	5	3	0.227	0.385	0.227	0.068	0.068	42	
9	6	3	0.273	0.462	0.273	0.068	0.068	52	
10	7	3	0.318	0.538	0.318	0.068	0.068	63	
11	8	3	0.364	0.615	0.364	0.068	0.068	75	
12	9	3	0.409	0.692	0.409	0.068	0.068	82	
13	10	3	0.455	0.769	0.455	0.068	0.068	95	
(The middle part of the table is omitted.)									
11	0	11	0.000	0.000	0.000	0.500			
12	1	11	0.045	0.077	0.045	0.500	0.045	80	
13	2	11	0.091	0.154	0.091	0.500	0.091	94	
12	0	12	0.000	0.000	0.000	0.545			
13	1	12	0.045	0.077	0.045	0.545	0.045	93	
13	0	13	0.000	0.000	0.000	0.591	0.000	92	

Note:

(1) The available capacity to the imbedded MODM is s^2, the input state.

(2) The optimal policy for Stage 3 corresponds to the maximum satisfaction from Table 3.

(3) ** (in Remarks) indicates the best policy to be chosen.

STEP 3. The process is for maximizing the satisfactory degree among all possible cases of combinations under different inputs and states, and yet the given budget remains a constant constraint. Therefore, the individual degree of satisfaction, corresponding to the spending budgets, at different stages is listed in Tables 1–4 as well. This max-min operation search for an optimal solution is similar to the traditional fuzzy MODM. Note that the fuzzy constraint for the budget allocation in each stage has been incorporated into the outside policy control and the inside goal, and thus the term of fuzzy constraint in Step 1 will be implicitly expressed in the corresponding tables.

Table 1 shows the degree of satisfaction among two objectives under the capacity available for x_1 at Stage 1. To save some space, we omit the middle part, i.e., capacity available s^0 is among the interval of $[6, 10]$; however, it does not affect our understanding. Then we continue to calculate the degree of satisfaction from Stage 2 to Stage 4. The following Tables 2–4 show the degree of satisfaction among two objectives under the capacity available for x_2, x_3, and x_4 at Stages 2–4, respectively. To save some space, we also omit the middle part, i.e., capacity available is among the interval of $[6, 10]$.

STEP 4. After all individual problems are solved, i.e., the single stage problems, we will establish the backward relationship among stages through DP structure. To trace the capacity allocated, we start from the last stage, Stage 4, with the capacity remaining after the previous stage, Stage 3, then go back to the second stage, and to the first stage. In the manipulation, two maximum satisfactions under the input states are defined for backward accumulation of the degree of satisfaction so far, i.e., $\mu^{3t*} = \min(\mu^{4*}, \mu^{3*})$ at Stage 3, $\mu^{2t*} = \min(\mu^{3t*}, \mu^{2*})$ at Stage 2, and $\mu^{1t*} = \min(\mu^{2t*}, \mu^{1*})$ at Stage 1, as described in Steps 5 and 6. The overall optimal solution for satisfaction of the problem will be achieved after all, i.e., μ^{1t*}, then the corresponding policy at each stage is optimal solution.

The calculating processes are shown in Tables 5–8 stage by stage-wise backward. The result indicates that optimal degree of satisfaction for the given problem is 0.045, which is intersected

Table 7. All possible noninferior outcomes under the total available capacity at Stage 2.

Available Capacity s^1	Activity x_2	Capacity Left	Degree of Satisfaction μ_{R12}	μ_{R22}	$\mu^{2*} = \min(\mu_{R12}, \mu_{R22})$	Max. Satisfaction from Stage 3 μ^{3t*}	Accumulated Satisfaction μ^{2t*}	Optimal Policy k^2	Remarks
0	0	0	0.000	0.000	0.000	0.000	0.000		
2	1	0	0.159	0.019	0.019	0.000	0.000		
4	2	0	0.318	0.038	0.038	0.000	0.000		
6	3	0	0.477	0.058	0.058	0.000	0.000		
8	4	0	0.636	0.077	0.077	0.000	0.000		
10	5	0	0.795	0.096	0.096	0.000	0.000		
12	6	0	0.955	0.115	0.115	0.000	0.000		
1	0	1	0.000	0.000	0.000	0.000	0.000		
3	1	1	0.159	0.019	0.019	0.000	0.000		
5	2	1	0.318	0.038	0.038	0.000	0.000		
7	3	1	0.477	0.058	0.058	0.000	0.000		
9	4	1	0.636	0.077	0.077	0.000	0.000		
11	5	1	0.795	0.096	0.096	0.000	0.000		
13	6	1	0.955	0.115	0.115	0.000	0.000		
2	0	2	0.000	0.000	0.000	0.023	0.000		
4	1	2	0.159	0.019	0.019	0.023	0.019	8	
6	2	2	0.318	0.038	0.038	0.023	0.023	15	
8	3	2	0.477	0.058	0.058	0.023	0.023	24	
10	4	2	0.636	0.077	0.077	0.023	0.023	35	
12	5	2	0.795	0.096	0.096	0.023	0.023	48	
3	0	3	0.000	0.000	0.000	0.045	0.000		
5	1	3	0.159	0.019	0.019	0.045	0.019	11	
7	2	3	0.318	0.038	0.038	0.045	0.038	19	
9	3	3	0.477	0.058	0.058	0.045	0.045	29	**
11	4	3	0.636	0.077	0.077	0.045	0.045	41	
13	5	3	0.795	0.096	0.096	0.045	0.045	55	
4	0	4	0.000	0.000	0.000	0.045	0.000		
6	1	4	0.159	0.019	0.019	0.045	0.019	14	
8	2	4	0.318	0.038	0.038	0.045	0.038	23	
10	3	4	0.477	0.058	0.058	0.045	0.045	34	
12	4	4	0.636	0.077	0.077	0.045	0.045	47	
5	0	5	0.000	0.000	0.000	0.068	0.000		
7	1	5	0.159	0.019	0.019	0.068	0.019	18	
9	2	5	0.318	0.038	0.038	0.068	0.038	28	
11	3	5	0.477	0.058	0.058	0.068	0.058	40	
13	4	5	0.636	0.077	0.077	0.068	0.068	54	
			(The middle part of the table is omitted.)						
11	0	11	0.000	0.000	0.000	0.182	0.000		
13	1	11	0.159	0.019	0.019	0.182	0.019	51	
12	0	12	0.000	0.000	0.000	0.227	0.000		
13	0	13	0.000	0.000	0.000	0.273	0.000		

Note:

(1) The available budget to the imbedded MODM is s^1, the input state.

(2) The optimal policy for Stage 2 corresponds to the maximum satisfaction from Table 2.

(3) ** (in Remarks) indicates the best policy to be chosen.

Table 8. All possible noninferior outcomes under the total available capacity at Stage 1.

Available Capacity s^0	Activity x_1	Capacity Left	Degree of Satisfaction μ_{R11}	μ_{R21}	$\mu^{1*} = \min(\mu_{R11}, \mu_{R21})$	Max. Satisfaction from Stage 4 μ^{2t*}	Accumulated Satisfaction μ^{1t*}	Optimal Policy k^1	Remarks
13	0	13	0.000	0.000	0.000	0.182	0.000		
13	1	11	0.091	0.038	0.038	0.058	0.038		
13	2	9	0.182	0.077	0.077	0.045	0.045	39	**
13	3	7	0.273	0.115	0.115	0.038	0.038		
13	4	5	0.364	0.154	0.154	0.019	0.019		
13	5	3	0.455	0.192	0.192	0.000	0.000		
13	6	1	0.545	0.231	0.231	0.000	0.000		

Note:

(1) The available budget to the imbedded MODM is s^0, the input state.

(2) The optimal policy for Stage 1 corresponds to the maximum satisfaction from Table 1.

(3) ** (in Remarks) indicates the best policy to be chosen.

accumulatedly by 0.045 at Stage 4, 0.045 at Stage 3, 0.058 at Stage 2, and 0.077 at Stage 1, respectively. The optimal proposal for capacity allocated is $(x_1, x_2, x_3, x_4) = (2, 3, 1, 2)$ with $z_1 = 33$ and $z_2 = 17$. Furthermore, the procedure of fuzzy DP will eliminate unpromising cases, i.e., keep DMs at any stage satisfactory. Compared to the crisp solution with two objectives: $z_1 = 44$ with $(x_1, x_2, x_3, x_4) = (0, 6, 1, 0)$, and $z_2 = 52$ with $(x_1, x_2, x_3, x_4) = (0, 0, 13, 0)$, the suggested proposal is conducted stage-wise in a well-balanced way.

Table 5 shows all possible noninferior outcomes under the total available capacity at Stage 4. Then we continue to process backward to Stages 3, 2, and 1, accordingly, as illustrated in Tables 6–8. To save some space, we also omit the middle part, i.e., capacity left under the range of $[6, 10]$ in Tables 6 and 7.

2.3. Fuzzy Multilevel Dynamic Programming

Multilevel programming (MLP) tries to simulated the decision making process in a hierarchical organization or a multilevel system with multiple executors. The system explicitly assigns each decision unit a unique objective, a set of decision variables, and a set of common constraints that will affect all decision units [32]. The basic concept of MLP technique is that an upper-level DM sets his or her goal and/or decisions and then asks each subordinate level of the organization for its optimum which is calculated independently. Lower-level DM' decisions are then submitted and modified by the upper-level DM with consideration of the overall benefits for the organization. The process is continued until a satisfactory solution is reached. This decision-making process is extremely practical for such decentralized systems as agriculture, government policy, economic systems, finance, warfare, transportation and network designs, and is suitable for conflict resolution [8,9,32].

In the past decades, there have been many approaches to solving MLP or hierarchical optimization problems. Traditional programming tools, such as decomposition principle (in linear programming), goal programming, multiobjective programming, or game theory, cannot meet the common features of multilevel systems. Thus, many heuristic methodologies have been proposed to solve MLP during the last three decades. Most methods for linear cases are based on concepts of extreme point search and transformation approach. The former is to search for a compromise vertex by simplex algorithm based on adjusting higher level control variables, but it is rather inefficient, especially for large size problems. The transformation approach involves transforming lower-level programming problems to be the constraints of the higher level

by Karush-Kuhn-Tucker (KKT) conditions or other functions [8]. Because nonlinear terms will appear in constraints, auxiliary problems become complicated and sometimes unmanageable. Since most existing methods are computationally inefficient, Shih *et al.* [18] use the concepts of tolerance membership functions and multiobjective optimization to develop a fuzzy approach for solving MLP.

The auxiliary problem of fuzzy approach for equation (2) can be expressed as follows:

$$\begin{aligned}
&\max \lambda, \\
&\text{s.t. } \mathbf{A}_1\mathbf{x}_1 + \mathbf{A}_2\mathbf{x}_2 \leq \mathbf{b}, \\
&\frac{[z_1(\mathbf{x}) - z_1^-]}{[z_1^+ - z_1^-]} \geq \lambda, \\
&\frac{[(\mathbf{x}_1^U + \mathbf{p}_1) - \mathbf{x}_1)]}{\mathbf{p}_1} \geq \lambda \mathbf{I}, \\
&\frac{[\mathbf{x}_1 - (\mathbf{x}_1^U - \mathbf{p}_2)]}{\mathbf{p}_2} \geq \lambda \mathbf{I}, \\
&\frac{[z_2(\mathbf{x}) - z_2^-]}{[z_2^+ - z_2^-]} \geq \lambda, \\
&\lambda \in [0, 1], \\
&\mathbf{x}_1, \mathbf{x}_2 \geq 0,
\end{aligned} \tag{6}$$

where $\mathbf{I}$ is a column vector with the same dimension as $\mathbf{x}_1$ and all its elements are 1. z_1^+ and z_1^- represent the PIS and NIS of z_1, respectively, so as the z_2. In addition, $\mathbf{p}_1$ and $\mathbf{p}_2$ are the tolerances of LHS and RHS of $\mathbf{x}_1$, respectively.

The above equation can be solved through any mathematical programming code and the procedure, a supervised search procedure, which has been proven to be efficient. Besides, it will not increase the complexity of the original problem and is flexible for extension [18]. Thus, the procedure can be easily applied to knapsack problems in a top-down process.

As the decision will be made over time, it is a general form of MLP in which decisions may be changed over various stages. The central part of the problem is an imbedded MLP problem under a DP structure, so that the resources may be allocated over time. Note that most problems will concentrate on discrete (time) resource-allocation processes for easy comprehension, and thus the term of "stages" will replace the term of "time" for a clear definition. In fuzzy MODM for resource allocation, Esogbue and Bellman [14] have offered a fuzzy mathematical model of the resultant fuzzy allocation process by decomposing the system into three levels, and process these hierarchical levels by the concept of stage of DP. However, we consider the stage as time horizon instead of this simple approach for dealing with a more complex system. Moreover, MLP over various stages is suitable for long range planning, especially good for production planning, inventory control, social-economic policy development, and resource allocation in planning, programming and budget system.

To attack the knapsack problem, the solving procedure for MLKP is similar to the MOKP's, but the difference only exists in its imbedded system. Compared to the procedure of MOKP, we make some modifications as follows.

STEP 1. PREPARATION PHASE. Obtain reference information.

(1b) Establish the degree of satisfaction of each DM in terms of fuzzy membership function by the distance between his/her PIS and NIS at each stage. The establishment will search for a restricted decision space controlled by the multiple level DMs.

STEP 2. FORWARD PHASE. Seek compromise solutions among multiple levels for possible budgets allocated at each stage.

Take minimum operation of the multiple levels' objectives and controlled decisions as their modified decision for possible budgets allocated at each stage.

STEP 3. Choose optimal decisions of MLP.

Take maximum operation of the possible outcomes from Step 1 given by each specified budget at each stage as the optimal decision of MLP. The establishment will search for a restricted decision space controlled by the multiple level DMs.

The rest of the steps are the same as described in Section 2.2, and a satisfactory solution is obtained with the maximum degree of satisfaction.

Now, we shall see that through an example.

EXAMPLE 2. A bilevel knapsack problem.

$$\begin{aligned}
&\max_{x1,x2} z_1 = 4x_1 + 7x_2 + 2x_3 + x_4 && \text{(upper level)},\\
&\max_{x3,x4} z_2 = 2x_1 + x_2 + 4x_3 + 3x_4 && \text{(lower level)},\\
&\text{s.t. } 2x_1 + 2x_2 + x_3 + x_4 \leq 13,\\
&\quad x_1,\ x_2,\ x_3, \text{ and } x_4 \text{ are positive integers.}
\end{aligned}$$

Similar to the procedure shown in Example 1, we first establish payoff tables, then the decision information is transformed in terms of fuzzy membership function. And these data are joined into the DP structure. To demonstrate the difference between MOKP and MLKP, we assume that the lower level DM asks for more resources, i.e., $x_3 \geq 3$, whose action will embedded into the decision process.

Following the modified procedure, the result indicates that optimal degree of satisfaction for the given problem is 0.038, that is accumulated of 0.068 at Stage 4, 0.091 at Stage 3, 0.038 (or 0.058) at Stage 2, and 0.077 (or 0.038) at Stage 1, respectively. Because alternative optimal solutions existed, there are two optimal proposals for capacity allocation, which are $(x_1, x_2, x_3, x_4) = (2, 2, 2, 3)$ and $(1, 3, 2, 3)$, respectively. And the objective will be $z_1 = 29$ and $z_2 = 23$, $z_1 = 32$ and $z_2 = 22$ accordingly. In comparison with the solution of MOKP, the degree of satisfaction for MLKPs is less due to a restricted space. However, their objective functions of MLKP and MOKP cannot be compared with each other, and all are noninferior. In addition, we do not list the calculation process, but the interested readers can refer to Tables 1–8 of Example 1 with a limited decision space.

3. CAPACITY ALLOCATION

There is a famous rule named turnpike theorem dominating the resource to be allocated in knapsack problems [21]. It is said that the object with the largest value of c_i/w_i, where $i = 1, 2, \ldots, N$, is the best object to be chosen, where c_i and w_i are the utility and the capacity (or weight) of object j in equation (1). For a large-capacity KP, the solving procedure starts from putting the object with the largest value in the knapsack, then putting the object with the second-largest value in the knapsack, and continue in this fashion until the object with the best remaining value will overfill the knapsack. Thus, the capacity will be allocated in an efficient way in the crisp domain.

In a fuzzy domain, the decision could be viewed as the intersection of fuzzy constraints and fuzzy objective functions [13]. And for a multistage decision process, the same operation can be processed recursively. Thus, the capacity can be allocated through the same operation level-wise and stage-wise in a fuzzy environment. In addition, the fuzzy membership function is established under how much capacity is allocated with respect to the possible total amount of capacity, and under how much return with respect to the total accumulated return of its objective function. Note that the fuzzy membership function represents the degree of satisfaction of the decision or the objective in DMs' mind.

After the insides of two approaches are discussed, we can see that the turnpike theorem is not applicable to KP in the fuzzy domain due to the ration of c_i/w_i is not the concern of fuzzy operation. Fuzzy decision tries to get the degree of satisfaction as much as possible, and eliminate undesirable cases with zero value. Thus, the capacity will be allocated level-wise and stage-wise in a balanced way, i.e., everyone or every stage might get some, instead of the variable with the largest value of c_i/w_i to be chosen in a first priority. Furthermore, compensatory operation might relieve the situation mentioned above, i.e., the undesirable cases would be less, in fuzzy decision making [17], and the argument is still valid.

4. CONCLUSIONS AND REMARKS

In this study, we have investigated multiobjective and multilevel knapsack problems in a fuzzy environment. An efficient algorithm is developed, and a solution procedure has been proposed through Microsoft Excel as well. Although the size of the examples under scrutinizing is small, large size problems are expected to solve through the same procedure.

Since different decision variables can be controlled by separated different decision units (or DMs) in a multilevel system, its decision space will be more restricted than that of the MODM problems'. In general, its solution will no better than that of the MODM's. However, both are efficient solutions.

In a general resource allocation problem, the number of decision variables will not always equal the number of the stages, but the situation will happen in solving KPs. Consequently, we could think that the KP is a special case of the resource allocation problems.

Notwithstanding only bilevel problems are illustrated, multiple-level dynamic programming problems can be solved through the same algorithm. Please see the details of the search algorithm in [17] for simplifying multilevel structures.

The efficiency of our fuzzy approach is dependent on the scheme of dynamic programming with a loose structure. We have not discussed the computational problem in this study, and interested readers might check some algorithms, e.g., [10], in the literature. And looking for a short-cut algorithm in DP structure will be the future direction.

In this study we have not involved the integer programming-based algorithms for MOKPs and MLKPs. Readers can go through the contents of Shih and Lee [33] seeing a case of multilevel minimum-cost flow problem. This would be another direction for further investigation.

REFERENCES

1. R.K. Ahuja, T.L. Magnanti and J.B. Orlin, *Networks Flows: Theory, Algorithms, and Applications*, Prentice-Hall, Englewood Cliffs, NJ, (1993).
2. S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*, John-Wiley, Chichester, West Sussex, (1990).
3. H.M. Salkin, The knapsack problem: A survey, *Naval Research Logistics Quarterly* **22** (1), 127–144, (1975).
4. A. Goicoechea, D.R. Hanson and L. Duckstein, *Multiobjective Decision Analysis with Engineering and Business Application*, John Wiley, New York, (1982).
5. M. Visee, J. Teghem, M. Pirlot and E.L. Ulungu, Two-phrases method and branch and bound procedures to solve the bi-objective knapsack problem, *J. of Global Optimization* **12**, 139–155, (1998).
6. R. Andonov, V. Poirriez and S. Rajopadhye, Unbounded knapsack problem: Dynamic programming revisited, *European J. of Operational Research* **123**, 394–407, (2000).
7. T. Erlebach, H. Kellerer and U. Pferschy, Approximating multiobjective knapsack problems, *Management Science* **48** (12), 1603–1612, (2002).
8. U.P. Wen and S.T. Hsu, Linear bi-level programming problems—A review, *J. of Operational Research Society* **42**, 125–133, (1991).
9. O. Ben-Ayed, Bilevel linear programming, *Computers and Operations Research* **20**, 485–501, (1993).
10. K.I. Cho and S.H. Kim, An improved interactive hybrid method for the linear multi-objective knapsack problem, *Computers and Operations Research* **24** (11), 991–1003, (1997).
11. K. Klamroth and M.M. Wieck, Dynamic programming approaches to the multiple criteria knapsack problem, *Naval Research Logistics* **47**, 57–76, (2000).
12. F.S. Salman, J.R. Kalagnanam, S. Murthy and A. Davenport, Cooperative strategies for solving the bicriteria sparse multiple knapsack problem, *J. of Heuristics* **8**, 215–239, (2002).

13. R.E. Bellman and L.A. Zadeh, Decision making in a fuzzy environment, *Management Science* **17**, B141–164, (1970).
14. A.O. Esogbue and R.E. Bellman, Fuzzy dynamic programming and its extensions, In *TIMS Studies in the Management Sciences*, Vol. 20, (Edited by H.-J. Zimmermann, L.A. Zadeh and B.R. Gaines), pp. 147–167, (1984).
15. J. Kacprzyk, *Multistage Decision-Making under Fuzziness: Theory and Applications*, Verlag TUV Rheinland, Köln, (1983).
16. J. Kacprzyk and A.O. Esogbue, Fuzzy dynamic programming: Main development and applications, *Fuzzy Sets and Systems* **81** (1), 31–45, (1996).
17. H.S. Shih and E.S. Lee, Discrete multi-level programming in a dynamic environment, In *Dynamic Aspects in Fuzzy Decision Making Volume 73, Studies in Fuzziness and Soft Computing*, (Edited by Y. Yoshida), pp. 79–98, Physica-Verlag, Heidelberg, (2001).
18. H.S. Shih, Y.J. Lai and E.S. Lee, Fuzzy approach for multi-level mathematical programming problems, *Computers and Operations Research* **23** (1), 73–91, (1996).
19. E.S. Lee, Fuzzy multiple level programming, *Applied Mathematics and Computation* **120**, 79–90, (2001).
20. R.E. Bellman and E.S. Lee, History and development of dynamic programming, *Control Systems Magazine* **14** (4), 24–28, (1984).
21. E.V. Denardo, *Dynamic Programming: Models and Applications*, Prentice-Hall, Englewood Cliffs, NJ, (1982).
22. R.E. Bellman and E.S. Lee, Functional equations in dynamic programming, *Aequationes Mathematicae* **17** (1), 1–18, (1978).
23. H.A. Taha, *Operations Research: An Introduction*, 7th edition, Pearson Education, Upper Saddle River, NJ, (2003).
24. C.L. Chen, C.Y. Chang and D.Y. Sun, Solving multi-objective dynamic optimization problems with fuzzy satisfying method, *Optimal Control Applications and Methods* **24**, 279–296, (2003).
25. K.K. Lai and L. Li, A dynamic approach to multi-objective resource allocation, *European J. of Operational Research* **117** (2), 293–309, (1999).
26. M.L. Hussein and M.A. Abo-Sinna, A fuzzy dynamic approach to the multicriterion resource allocation problem, *Fuzzy Sets and Systems* **69** (2), 115–124, (1995).
27. L. Li and K.K. Lai, Fuzzy dynamic programming approach to hybrid multiobjective multistage decision-making problems, *Fuzzy Sets and Systems* **117** (1), 13–25, (2001).
28. M.E. Salukvadze, An approach to the solution of the vector optimization problem of dynamic systems, *J. of Optimization Theory and Applications* **38** (3), 409–422, (1982).
29. D. Li and Y.Y. Haimes, Multiobjective dynamic programming: The state of the art, *Control-Theory and Advanced Technology* **5** (4), 471–483, (1989).
30. C.L. Hwang and K. Yoon, *Multiple Attribute Decision Making: Methods and Applications*, Springer-Verlag, Berlin, (1981).
31. C.L. Hwang and A.S.M. Masud, *Multiple Objectives Decision Making: Methods and Applications*, Springer-Verlag, Berlin, (1979).
32. G. Anandalingam and T.L. Friesz, Editors, Hierarchical optimization, *Annals of Operational Research* **34**, (1992).
33. H.S. Shih and E.S. Lee, Fuzzy multi-level minimum-cost flow problems, *Fuzzy Sets and Systems* **107** (2), 159–176, (1999).