JOURNAL OF MATHEMATICAL ANALYSIS AND APPLICATIONS 81, 524–544 (1981)

An Interactive Dynamic Programming Approach to Multicriteria Discrete Programming*

BERNARDO VILLARREAL

Instituto Technologico y de Estudios Superiores de Monterrey, Monterrey, Nuevo Leon, Mexico

AND

MARK H. KARWAN

Department of Industrial Engineering, State University of New York at Buffalo, Buffalo, New York 14260

Submitted by E. Stanley Lee

Several interactive schemes for solving multicriteria discrete programming problems are developed under a dynamic programming framework. It is assumed that the decision maker's preference structure satisfies the conditions of transitivity, monotonicity, and nonsatiation. Hybrid procedures are also structured by including branch and bound ideas into the recursions. Initial computational results are offered.

INTRODUCTION

The development of interactive procedures for solving multicriteria programming problems has received much attention. Among the many algorithms suggested are those due to Benayoum *et al.* [2], Geoffrion *et al.* [7], Dyer [4], Chankong and Haimes [3], Zionts [21], Zionts and Wallenius [22], Villarreal and Karwan [19], and Lee [8]. Even though the research done in this area has been very productive, only the schemes suggested by Zionts [21], Lee [8], and Villarreal *et al.* [19] can be applied to solve (mixed) integer programming problems. This paper deals with the development of initial theoretical results and methodologies for solving multicriteria discrete programming problems, under a dynamic programming approach. One can view these schemes as extensions of the recursions developed by Villarreal and Karwan [16–18] for determining the efficient set

* This research was supported in part by the Consejo Nacional de Ciencia y Tecnologia, Mexico.

0022-247X/81/060524–21$02.00/0

of solutions for multicriteria discrete programming problems. They can also be viewed as special applications of the general interactive procedure suggested by Mitten [12].

The problem of concern is that of finding the policy, $(x_1,..., x_N)$, among a set of alternative policies defined by the set of constraints

$$S \triangleq \left\{ \sum_{n=1}^{N} a^n \cdot x_n \leqslant b, \qquad x \in X, \right.$$

where $a^n = (a_{1n},..., a_{mn})'$ and X denotes a set of discrete values for x, such that it satisfies the decision maker's preferences over the values of a set of criteria or attributes defined by

$$G \triangleq \{R_1(x_1),..., R_N(x_N)\},$$

where $R_n(x_n) = (r_{1n}(x_n),..., r_{pn}(x_n))'$ denotes a p-dimensional vector of finite functions of x_n.

The general process to be developed consists of interactively selecting best subpolicies (or partial policies) at each (or after several) stage(s) for each state vector. Then, at the final stage (N), the decision maker selects the policy that best suits his preferences among those most preferred policies obtained for each state vector Y_N $(\leqslant b)$.

Let the operators $>$ and $\sim$ denote preference and indifference, respectively. It will be assumed that the decision maker's preference structure is such that these operators satisfy the following conditions.[1]

(1) Both operators are transitive, i.e.,

- (i) If $a > b$ and $b > c$ then $a > c$.
- (ii) If $a > b$ and $b \sim c$ then $a > c$.
- (iii) If $a \sim b$ and $b > c$ then $a > c$.
- (iv) If $a \sim b$ and $b \sim c$ then $a \sim c$.

(2) The operator $\sim$ is reflexive and symmetric, i.e., if $a \sim b$, a is preferred as much as b, then $b \sim a$.

(3) The operator $>$ is assymetric and irreflexive, i.e., if $a > b$ then $b \not> a$, or b is not as preferred as a.

(4) Both operators are complete, i.e., for all alternatives, a and b, possible, either $a > b$, $b > a$, or $a \sim b$ $(b \sim a)$.

(5) Consider the feasible policies a, b, and c. The relationship $(a \text{ and } c) > (b \text{ and } c)$ (or $(a \text{ and } c) \sim (b \text{ and } c)$) is satisfied if and only if $a > b$ $(a \sim b)$. This property will be regarded as the monotonicity property.

[1] See Fishburn [5].

1. Development of the General Procedure

It can be shown (see also Bellman [1] and Nemhauser [15]) that the set of constraints that define the various alternate policies available to the decision maker, S, can be equivalently imposed by the following set of constraints which are appealing for a dynamic programming recursive approach

$$S^D \triangleq \begin{array}{ll} Y_{n-1} = Y_n - a^n x_n & (n = 2, \ldots, N) \\ Y_N \leqslant b & \\ Y_0 \leqslant Y_1 - a^1 x_1 & \\ x \in X. & \end{array}$$

Therefore, the problem reduces to finding the decision maker's preferred policy with relation to the criteria set, G, and subject to the constraint set S^D. The following result will enable one to structure the general interactive scheme.

Theorem 1. *Consider the vector of resources Y_m available at stage m. The decision maker's preferred (partial) policy $(x_1^*, \ldots, x_m^*)$ must be a feasible policy such that $(x_1^*, \ldots, x_{m-1}^*)$ is the preferred partial policy for stage $(m-1)$ and the vector of resources $Y_{m-1} = Y_m - a^m x_m^*$.*

Proof. Let $(x_1^*, \ldots, x_{m-1}^*)$ be the preferred policy at stage $(m-1)$ for the vector of resources $Y_{m-1} = Y_m - a^m x_m^*$. Assume that $(x_1, \ldots, x_{m-1}, x_m^*)$ is the preferred policy at stage m, for vector of resources Y_m, with $(x_1, \ldots, x_{m-1}) \neq (x_1^*, \ldots, x_{m-1}^*)$. By the monotonicity assumption, if $(x_1, \ldots, x_{m-1}, x_m^*) > (x_1^*, \ldots, x_{m-1}^*, x_m^*)$ then $(x_1, \ldots, x_{m-1}) > (x_1^*, \ldots, x_{m-1}^*)$. But this is a contradiction of the initial assumption. Q.E.D.

Remark. This result assumes that the decision maker can identify a preferred partial policy and was not indifferent to choosing several alternate possibilities. In the case in which he is indeed indifferent to a set of partial policies for a given state vector of resources, then the theorem should be modified so as to allow for the possibility that the partial policy $(x_1^*, \ldots, x_{m-1}^*)$ is part of the set of indifferent policies. The proof of the modified theorem can be similarly constructed via contradiction. The proof follows because any partial policy not a member of this set will not be preferred to any other.

Using these results in a stagewise process leads one to consider only the best (preferred) policy(ies) at the final stage for each of the possible state vectors (resource vectors) of values. Thus, the decision maker would elect the desired or preferred policy among them. The general scheme of the process is presented below, but first, the following notation is introduced.

Let $\rho(Y_m)$ represent the set of criteria vectors associated with the most preferred partial policy(ies) at stage m for a given vector of resources Y_m. Also, let

$$\psi_m = \bigcup_{Y_m \leqslant b} \rho(Y_m).$$

Step 0. Let $\rho(Y_0)$ for all Y_0, and ψ_0 be empty sets. Set $m = 1$.

Step 1. Construct the set of partial policies at stage m employing x_m and $(x_1, ..., x_{m-1})$ such that $(x_1, ..., x_m) \in X$ and $(x_1, ..., x_{m-1}) \in \psi_{m-1}$.

Step 2. Delete all infeasible partial policies (those not satisfying the resource constraints).

Step 3. For each state vector (resource vector) $Y_m (\leqslant b)$ present the feasible (partial) policies to the decision maker for selection purposes, and form the set ψ_m.

Step 4. Set $m = m + 1$. If $m > N$ go to step 5. Otherwise, go to step 1.

Step 5. Present the set ψ_N to the decision maker for purposes of choosing the preferred policy.

Notice that steps 1 and 2 are equivalent to constructing the sets of feasible partial policies for each vector of resources Y_m $(\leqslant b)$ at each stage. The scheme can be modified to determine the preferred policy(ies) for any vector of resources Y_N $(\leqslant b)$. The change would consist of imposing the constraint set

$$\sum_{n=1}^{N} a^n x_n \leqslant Y_N$$

at step 5, and considering the feasible policies for selection purposes.

2. Nonsatiated Preference Procedures

In this section, it is assumed that the G is of the form

$$G = \{R^N(x_N) \Delta R^{N-1}(x_{N-1}) \Delta \cdots \Delta R^1(x_1)\},$$

where Δ denotes a vector of monotonic operators and the functions included in G satisfy the conditions of separability and monotonicity given by Mitten [11] and extended to a multicriteria framework in Villarreal and Karwan [16].

A particular case of interest is that in which the decision maker would like to have as much from each criterion as is feasible. Hence, if there exist two feasible policies, x and x^*, such that

$$R^N(x_N^*)\,\Delta \cdots \Delta R^1(x_1^*) \geqslant R^N(x_N)\,\Delta \cdots \Delta R^1(x_1)$$

with at least one strict inequality, the decision maker would prefer x^* to x. The following result is a direct consequence of this new assumption.

THEOREM 2. *Consider a vector of resources Y_m at stage m. The preferred (partial) policy $(x_1^*,..., x_m^*)$ is among the set of efficient*[2] *(partial) policies, $A^0(Y_m)$.*

Combining this result and Theorem 1 gives one the flexibility to set up various strategies that could be incorporated into the interactive process of Section 1. Before describing a possible scheme consider the following results.

Let $\hat{A}(Y_m)$ represent the set of policies such that $(x_1^*,..., x_l^*)$ is the preferred policy for $Y_l = Y_m - \sum_{j=l+1}^{m} a^j x_j^*$ and $(x_1^*,..., x_m^*)$ is an efficient policy for the vector of resources Y_m. A direct consequence of this definition is the following result.

LEMMA 1. *The set $\hat{A}(Y_m) \subseteq A^0(Y_m)$.*

Further, from the nonsatiation, transitivity, and monotonicity properties, the following result holds.

THEOREM 3. *Consider a vector of resources Y_m at stage m. The set $\rho(Y_m) \subseteq \hat{A}(Y_m)$.*

From this result, it follows that the decision maker's most preferred policy is among the set of efficient policies remaining at stage N for $Y_N = b$.

The success of various schemes that one may suggest depends upon the size of the sets of efficient policites for each state vector. These sets would be presented for selection purposes instead of the sets suggested in step 3 of the procedure of Section 1. One would want to have as small a set as possible. Obviously, a better set to present would be a subset of $A^0(Y_m)$, say, $\hat{A}(Y_m)$. This implies that prior sets of questions were posed to the decision maker in previous stages. Also, one may employ questioning sessions in a stagewise continuous or intermittent fashion. Even though the stagewise continuous interaction with the decision maker would lead to the smallest set, $\hat{A}(\cdot)$, possible in subsequent stages, it also implies a thorough questioning. Thus, it seems reasonable that a better strategy would be to question the decision maker intermittently. Possible rules could be devised using either a maximum

[2] An efficient policy, x, is such that there is no other feasible policy, $\tilde{x}$, satisfying the expression $R^N(\tilde{x}_N)\,\Delta \cdots \Delta R^1(\tilde{x}_1) \geqslant R^N(x_N)\,\Delta \cdots \Delta R^1(x_1)$ with at least one strict inequality.

number of efficient policies per state vector of values or a certain number of stages, or both. The outline of one procedure is described below. Let EFF and STAG denote numbers of efficient partial policies per state vector and stages, respectively. Let also CARD($\cdot$) denote the current number of efficient partial policies.

Step 0. Let $\rho(Y_0)$ and $\hat{A}(Y_0)$ be empty sets, and CARD$(\hat{A}(Y_0)) = 0$ for all $Y_0 (\leqslant b)$. Set $m = l = 1$.

Step 1. For each state vector $Y_m (\leqslant b)$, construct the set of all possible partial policies with x_m and $(x_1, ..., x_{m-1}) \in \hat{A}(Y_{m-1} = Y_m - a^m x_m)$ such that $(x_1, ..., x_m) \in X$.

Step 2. Delete the infeasible partial policies for each associated vector of resources Y_m $(\leqslant b)$.

Step 3. For each vector of resources Y_m $(\leqslant b)$, obtain the set $\hat{A}(Y_m)$ via pairwise comparisons. If $l \neq$ STAG and CARD$(\hat{A}(y_m)) <$ EFF go to step 5. Otherwise, go to step 4.

Step 4. Present the set $\hat{A}(Y_m)$ to the decision maker for selection purposes. If $l =$ STAG set $l = 0$.

Step 5. If $m = N$ go to step 6. Otherwise set $m = m + 1$, $l = l + 1$, and go to step 1.

Step 6. Present the sets $\hat{A}(Y_N)$; $Y_N \leqslant b$, to the decision maker for selecting the policy he prefers the most.

This interactive scheme can also be modified to include bound ideas with the intention of reducing the sizes of the sets $\hat{A}(y_m)$.

3. An Interactive Hybrid Procedure

Bounding and fathoming criteria can be incorporated into the interactive procedure. Since the decision maker's preferred policy is among the set of efficient policies of the problem, one may employ sets of bounds such as those described in Villarreal and Karwan [17] to devise schemes aimed to eliminate partial policies not leading to efficient policies. They give the following definitions for the set of upper and lower bounds of an efficient set.

Definition 1. A set of upper bounds to the solution of a multicriteria programming problem is a set of points that satisfy the following conditions:

(1) Each element is either efficient or dominates at least one of the efficient solutions of the problem.

(2) Each efficient solution of the problem is dominated by at least one member of the set or is a member of the set.

DEFINITION 2. A set of lower bounds to the set of efficient solutions of a multicriteria programming problem is a set of points such that each element is either efficient or is dominated by at least one efficient solution of the problem.

Suppose that the solution to the following problem (to be denoted as the mth stage problem) is available at stage m.

$$H_m(Y_m) = v\text{-max}\{R^m(x_n)\,\Delta \cdots \Delta R^1(x_1)\},$$

$$\text{st: } \sum_{n=1}^{m} a^n \cdot x_n \leqslant Y_m, \qquad x_1, \ldots, x_m \in X.$$

Denote as the residual problem of this mth stage problem the one defined as follows.

$$v\text{-max}\{R^N(x_N)\,\Delta \cdots \Delta R^{m+1}(x_{m+1})\}$$

$$\text{st: } \sum_{n=m+1}^{N} a^n \cdot x_n \neq (b - Y_m), \qquad x_{m+1}, \ldots, x_N \in X.$$

The solution to this problem would be the best one could achieve efficient-wise with the remaining resources $(b - Y_m)$.

Let LB denote a set of lower bounds for the efficient set of solutions for the original problem (considering all the variables and the vector of resources b). Let also $UB_{m+1}(Y_m)$ denote the set of upper bounds for the set of efficient solutions of the residual problem for the previously given m-stage problem. Given these concepts, Villarreal and Karwan [17] prove the following result.

THEOREM 4. *Let an efficient subpolicy for the mth stage problem with vector of resources Y_m, say, x, be available. Let its p-dimensional return function values be denoted by $H_{m,x}$. If for every element $g_k \in H_{m,x} \oplus UB_{m+1}(Y_m)$ there exists an element $LB_j(k) \in LB$ such that*

$$g_k \leqslant LB_{j(k)} \tag{1}$$

with at least one strict inequality, then the subpolicy x cannot be part of an efficient or preferred policy.

$\oplus$ means that the operator Δ is performed with each member of the set $H_{m,x}$ and each member of $UB_{m+1}(Y_m)$. If $H_{m,x} = \{\binom{3}{2}, \binom{1}{4}\}$, $UB_{m+1}(Y_m) = \{\binom{6}{7}, \binom{5}{4}\}$, and $\Delta = \binom{+}{\cdot}$, then $\{H_{m,x} \oplus UB_{m+1}(Y_m)\} = \{\binom{9}{14}, \binom{8}{8}, \binom{7}{28}, \binom{6}{16}\}$.

Using this result and the previous concepts, Villarreal and Karwan [17] developed several fathoming schemes, and suggested various sets of bounds for an efficient set of solutions. Any of these schemes or sets of bounds may be incorporated into the previous scheme of Section 2 to structure hybrid procedures. The modifications required to include these ideas into the previous interactive scheme are the following.

(1) Include the determination of the set of lower bounds in step 0.

(2) Introduce in step 3 a counter of stages to decide when to compute sets of upper bounds and use a fathoming scheme. If the same counter is employed for deciding when to utilize questioning sessions, one would first use the fathoming scheme, and then, question the decision maker.

(3) Insert an additional step in which the computation of sets of upper bounds and the use of a fathoming scheme are performed. This would be executed whenever any of the rules is satisfied. If the number of stages is used for deciding when to question, and the critical number of reached, then the fathoming step is always performed prior to the questioning sessions.

(4) If desired, one may try to improve the set of lower bounds by computing new feasible solutions in the additional step just described.

The resulting hybrid preference procedure would use the concept of dominance, bounding and fathoming criteria, and questioning sessions to move towards the determination of the most preferred policy. The main problem of using this procedure is that of dimensionality. An increase in the number of constraints will lead to storage problems as well as to an increase in the number of questions required since the state vectors of values per stage increases. A desirable approach to alleviate the increases in the storage requirements is the imbedded state approach as suggested in Morin and Esogbue [14] and Villarreal and Karwan [16].

4. An Imbedded State Preference Procedure

The problem of large storage requirements caused by the dimensionality of the state vector Y_m can be alleviated in a fashion similar to that for finding the efficient set of policies for multicriteria discrete programming problems, i.e., by using the concept of resource-efficient policies defined in Villarreal and Karwan [16, 17].[4] This concept will let one obtain the sets of efficient policies for each vector of resources Y_m without having to identify each of

[4] A resource-efficient policy, x, is such that there is no other policy, x', such that $R^N(x'_N) \Delta \cdots \Delta R^1(x'_1) \geqslant R^N(x_N) \Delta \cdots \Delta R^1(x_1)$ and $\sum_{n=1}^N a^n x'_n \leqslant \sum_{n=1}^N a^n \cdot x_n$ with at least one strict inequality in the first expression.

them with the corresponding vector. From Theorem 6 of Villarreal and Karwan [16], one has that

$$A^0(Y_m) \in \phi^m, \qquad \text{for all } Y_m\ (\leqslant b),$$

where ϕ^m denotes the set of resource-efficient partial policies at stage m. This relationship and Lemma 1 imply that

$$\hat{A}(Y_m) \subseteq \phi^m, \qquad \text{for all } Y_m\ (\leqslant b).$$

Since one uses the sets $\hat{A}(Y_m)$ instead of $A^0(Y_m)$, the set of resource-efficient partial policies obtained at each stage will be a subset, say, $\hat{\phi}^m$, of ϕ^m. In order to include the resource efficiency concept into the working procedure, one must discard the rule used to decide when to use questioning sessions on the basis of the current number of efficient policies per state vector. One could instead use a similar rule based upon the cardinality of the sets $\hat{\phi}^m$. Let CARD(m) denote such number and MEFF represent a number such that if CARD(m) $\geqslant$ MEFF, questioning sessions would be called for. Before outlining a scheme that uses these concepts, the following helpful results are developed.

Let $F_{(m,x)}$ be a set such that, for a given $x \in X$, any element $z \in F_{(m,x)}$ is such that $z \in X$,

$$R^m(x_m)\,\Delta \cdots \Delta R^1(x_1) \not\geqslant R^m(z_m)\,\Delta \cdots \Delta R^1(z_1)$$

and

$$Ax \leqslant Az = Z.$$

LEMMA 2. *Let a solution $x \in X$ and its associated set $F_{(m,x)}$ be given. If for $z \in F_{(m,x)}$, $z \in X$, x is preferred to z, discard z from further consideration in the procedure.*

Proof. Suppose that $x \in A^0(Z)$. Then,

(1) Let $z \in A^0(Z)$. One can discard z from further consideration if it is not included in any other set of efficient policies, or it is not preferred in any of the sets in which it is included. In the first case, the result follows from Theorem 3. In the second case, one has that if $z \in A^0_{(R)}$ then

$$Ax \leqslant Az \leqslant R.$$

But,

(i) If $x \in A^0_{(R)}$, Theorem 3 applies.

(ii) If $x \notin A^0_{(R)}$, then there exists another policy, y, that dominates x, and so, $y > (x >)\, z$.

(2) Let $z \notin A^0(Z)$. Then, it would not be a member of any set of efficient policies since for any $L \leqslant Z$, it is infeasible, and for $L \geqslant Z$, there is a $y \in A^0(Z)$ that dominates it. Now, assume that $x \notin A^0(Z)$.

(i) Let $z \in A^0(Z)$. Obviously, there is a $y \in A^0(Z)$ that dominates x, and so, $y > (x >) z$.

(ii) Let $z \notin A^0(Z)$. Then, the previous case 2 applies. Q.E.D.

The point, x, will be called the generator of the set $F_{(m,x)}$. The previous scheme of Section 2 will now be adapted to incorporate the concept of resource efficiency. This is outlined as follows:

Step 0. Let ϕ^0 be empty and set CARD(0) = 0, and $m = l = 1$.

Step 1. Obtain the possible partial policies $(x_1,..., x_m)$ with x_m such that $(x_1,..., x_m) \in X$, and $(x_1,..., x_{m-1}) \in \hat{\phi}^{m-1}$.

Step 2. Delete the infeasible partial policies (those not satisfying the resource contraints).

Step 3. Obtain the set $\hat{\phi}^m$ via pairwise comparisons. If $l \neq$ STAG and CARD(m) = MEFF go to step 5. Otherwise, go to step 4.

Step 4. Obtain the sets $\hat{A}(Y_m)$ or $F_{(m,\cdot)}$. Present those with cardinality greater than one to the decision maker for selection purposes, and discard those that satisfy Theorem 3 or Lemma 2. If l = STAG, set $l = 0$.

Step 5. If $m = N$ go to step 6. Otherwise, set $m = m + 1$, $l = l + 1$, and go to step 1.

Step 6. For the required vector of resources Y_N ($\leqslant b$), present the set of efficient policies $\hat{A}(y_N)$ ($\subseteq \hat{\phi}^N$) to the decision maker for choosing his preferred policy.

Notice that whenever one fathoms or eliminates a policy x by preference, one would not consider it as part of any other set $\hat{A}(y_m)$ or $F_{(m,\cdot)}$. If there is a set $F_{(m,x)}$ then, one would not consider the whole set. Even though this outline does not include the use of bounding and fathoming criteria, one can easily modify it to include them as in Section 3.

4.1. *The Linear Utility Function Case*

A special case of interest is that in which it is assumed that the decision maker's utility function is linearly additive, i.e.,

$$\lambda\{R^N(x_N)\Delta,..., \Delta R^1(x_1)\},$$

where $\lambda \in R^p$ denotes a vector of preference weights assigned by the decision maker to the criteria set. One can show that the linear additive utility function satisfies the transitivity and monotonicity properties of the preference structure assumed previously. Assume that Δ is a vector of addition operators and

$$r_{ij}(x_j) = c_{ij} \cdot x_j \qquad (i = 1,..., p) \quad (j = 1,..., N).$$

Then,

$$\lambda\{R^N(x_N)\,\Delta,..., \Delta R^1(x_1)\} = \sum_{i=1}^{p} \sum_{j=1}^{N} \lambda_i c_{ij} x_j.$$

The exploration of the preference structure of the decision maker under an interactive framework would be accomplished by an exploration for the appropriate set of preference weights assigned to each criterion or attribute. Under the dynamic programming approach, this can be achieved in a stagewise manner using the responses of the decision maker to the peference questions posed to him during the procedure. For example, if one has the resource efficient policies $x, y, z \in R^N$, and $x > y$, $x \sim z$, then,

$$\lambda^t Cx > \lambda^t Cy$$

and

$$\lambda^t Cx = \lambda^t Cz,$$

where $\lambda^t = (\lambda_1,..., \lambda_p)$ denotes the true values of the vector of weights λ, and C is a $p \times N$ matrix of objective coefficients. By constructing these inequalities, one will be able to obtain approximate values for the true set of weights. Let λ^s denote the constraint set obtained from the responses of the decision maker. This set of constraints may be employed to eliminate (partial) resource-efficient policies that will not (lead or) be (to the) preferred (policy). This can be accomplished by solving the following subproblems. Let x and y be two resource-efficient policies with criteria Cx and Cy, respectively. The first subproblem is the following.

$$\begin{aligned} z^0 &= \min\{\lambda C(x - y)\},^5 \\ \text{st: } &\lambda \in \lambda^s, \\ &\lambda \geqslant 0. \end{aligned}$$

If $z^0 > 0$, one has $\lambda cx > \lambda cy$, and $x > y$. This is true since any set, λ, that satisfies the constraint set will also satisfy the relationship.

$$\lambda C(x - y) > z^0.$$

[5] Note that if $x(y)$ dominates $y(x)$ then $z^0 > 0$ $(z^1 > 0)$.

Since the true set of preference weights is contained in the set, λ^s, the previous relationship also holds for their values. In case that $z^0 \leqslant 0$, one would solve a second problem defined as

$$z^1 = \min\{\lambda C(y - z)\},^5$$
$$\text{st: } \lambda \in \lambda^s,$$
$$\lambda \geqslant 0.$$

Of course, if $z^1 > 0$ then $y > x$. If $z^0 \not> 0$ and $z^1 \not> 0$, one would assess the decision maker's preferences via a questioning session. The set, λ^s, may be updated after each session of questions.

5. An Alternative View of the Imbedded State Procedure

The previously outlined scheme has been regarded as a dynamic programming algorithm that uses bounding and fathoming ideas as well as preference questioning in its search towards the preferred policy. Recall that the use of bounds came about because of the fact that the decision maker's preferred policy (under nonsatiation) is among the set of efficient policies.

One could also interpret this scheme as an interactive branch and bound procedure. In this case, the branching and bounding strategies employed are based upon the works of Villarreal and Karwan [17], Marsten and Morin [10], and Morin and Marsten [13]. These are of different structure to those commonly used in standard branch and bound algorithms (see Garfinkel and Nemhauser [6], or Zionts [20], for example). In this scheme, branching and bounding are heavily controlled and dictated by the dynamic programming framework. Branching is always performed in the next (stage) variable of the problem. Thus, it leaves very little flexibility to decide where and in what variable to branch on. The computation of bounds can be devised in a very effective manner. Due to the dynamic programming structure, several types of bounds can be shared among all the nodes of the particular branching level. Various fathoming schemes (see Villarreal and Karwan [17]) that are based on these bounds can be structured for fathoming purposes at each of the nodes. A further modification of the scheme of Section 4 that can be interpreted ås an interactive implicit enumeration scheme is possible.

Let $F^*_{(m,x)}$ be a set of policies such that for $x \in X$ and any $z \in F^*_{(m,x)}$, the following relationships are satisfied.

$$Cz \not\geqslant Cx \qquad \text{or} \qquad Cz \leqslant Cx$$

and

$$Az \geqslant Ax,$$

with at least one inequality. Then, if $Cz \not\geqslant Cx$ and $Az \geqslant Ax$ are satisfied $z \in F_{(m,x)}$. If $Cz \leqslant Cx$ and $Az \geqslant Ax$, $z \notin \phi^m$ and must be eliminated. Using this last set of policies will enable on to determine to set of policies that are not part of the set of resource efficient policies as well as to construct the sets $F_{(m,x)}$ in the same step. It can be seen that

$$F_{(m,x)} \subseteq F^*_{(m,x)}.$$

Obviously, any policy $z \notin F_{(m,x)} \cap F^*_{(m,x)}$ but such that $z \in F^*_{(m,x)}$ will be eliminated by preference since it is a dominated solution. Since all the points that are dominated will be eliminated, the resulting set of (partial) policies at the end of stage m will correspond to $\hat{\phi}^m$. An algorithm based upon the use of the sets just described is outlined below.

Step 0. Let $\hat{\phi}^0$ be empty and set $m = 1$.

Step 1. Obtain the possible partial policies such that $(x_1,..., x_{m-1}) \in \hat{\phi}^{m-1}$ and $(x_1,..., x_m) \in X$.

Step 2. Delete the infeasible partial policies (those not satisfying the resource constraints).

Step 3. Obtain the sets $F^*_{(m,\cdot)}$. Present those with cardinality greater than one to the decision maker for choosing purposes, and discard those that are dominated or satisfy Lemma 2,. The resulting set is $\hat{\phi}^m$.

Step 4. If $m = N$ go to step 5. Otherwise, set $m = m + 1$ and go to step 1.

Step 5. Obtain the set of (partial) policies contained in $\hat{\phi}^N$ that are feasible for specific vector of resources y_N ($\leqslant b$). Present these to the decision maker for selection purposes.

Additional bounding and fathoming tests can be incorporated by including sets of lower and upper bounds in the scheme such as those described in Section 3. The changes to be made in the prior procedure are the following.

(1) Determine a set of lower bounds for the problem at step 0.

(2) Compute sets of upper bounds after step 2 and use them to eliminate policies employing Theorem 4.

(3) If desired, one may include another step to try to improve upon the set of lower bounds at each stage. This can be accomplished in the same step in which the set of upper bounds are computed.

6. Conclusions and Computational Results

The computational feasibility of the schemes previously outlined in prior sections depends upon the size of the sets $F_{(m,x)}$, $F^*_{(m,x)}$, or $\hat{A}_{(\cdot)}$. A large cardinality of these sets would imply that many questions would be posed to the decision maker. However, it also implies that the decrease of the sizes of subsequent sets (in later stages) will be of relevance. If a linear additive utility function is assumed, the performance of the procedure can be improved by the use of the subproblems suggested in Section 4.1. Their effectiveness is of great interest because one can use them to avoid questions to the decision maker, i.e., if any of the conditions given in Section 4.1 are met. one will be able to eliminate policies not leading to the preferred solution without asking the decision maker.

In order to obtain evidence about the computational possibility of the algorithm suggested, a sample of ten 0–1 bicriterion multidimensional knapsack problems was solved using the dynamic programming recursions to obtain sets of efficient solutions given in Villarreal and Karwan [17]. It was assumed that monotonicity, transitivy, and nonsatiation are characteristics of the decision maker's preference structure. All the problems consist of ten constraints, values of b equal to 0.75 times the sum of the associated row coefficients, and a density of 0.90. The size of the sets $F_{(m,\cdot)}$, $m = 1,..., 10$, was determined, as well as the specific points contained in such sets. The size is per stage, and it is assumed that no previous questions were asked. Given this information, the maximum and minimum number of questions that could be asked to the decision maker to eliminate resource-efficient policies were computed. The maximum number of points that would be eliminated corresponds to the minimum number of questions. This information is shown in Table I. Observe that the size of the minimum number of questions becomes relevant in later stages. This indicates the possibility for eliminating a significant number of elements of the sets via questioning sessions or (if a linear utility function is assumed) the subproblems suggested in Section 4.1. Table II illustrates the percentage mean and range of the number of (partial) policies that could be eliminated per stage. As previously pointed out, this becomes significant in later stages.

On the basis of the results of Tables I and II, it was decided to program the interactive scheme of Section 5 to solve multicriterion linear integer programming problems in which $X = \{x \mid 0 \leqslant x_n \leqslant k_n,\ x_n \text{ integer}, n = 1,..., N\}$. The scheme is employed (in all the results of this section) to interactively use the response of the decision maker to preference questions, to eliminate resource-efficient partial policies not leading to the preferred policy(ies). At the final stage, one will have a subset of efficient policies to present to the decision maker for selection purposes. The procedure is based on the use of the sets $F^*_{(m,\cdot)}$, and it is assumed that the decision maker's

TABLE I

Illustration of the Maximum Number of Questions and Solutions to Pose and Eliminate, Respectively

PROB	Stage number																				
	4			5			6			7			8			9			10		
No.	*M*	*m*	EF	*M*	*m*	EF	*M*	*m*	EF	*M*	*m*	EF	*M*	*m*	EF	*M*	*m*	EF	*M*	*m*	EF
1	4	3	16	9	6	32	48	22	64	200	63	124	341	88	173	403	118	206	1804	268	390
2	9	5	16	58	17	32	76	17	40	24	9	32	118	24	60	401	60	104	1221	111	163
3	3	2	16	28	10	32	95	24	62	294	47	88	418	59	105	845	103	155	2467	157	223
4	0	0	16	1	1	32	84	22	64	304	53	128	1459	136	233	—	—	—	—	—	—
5	0	0	12	1	1	19	5	4	36	55	20	68	181	45	111	730	100	177	2311	214	313
6	4	3	16	13	7	32	25	13	56	72	30	94	492	97	167	1012	146	246	1441	176	297
7	1	1	16	12	6	32	53	20	62	498	64	116	1434	127	194	—	—	—	—	—	—
8	2	2	16	14	8	32	44	21	61	166	49	106	783	111	176	1704	192	273	—	—	—
9	4	3	16	7	23	23	21	10	35	87	27	66	101	29	83	47	18	70	95	38	109
10	1	1	13	1	1	18	18	9	26	33	16	38	83	24	38	151	29	48	282	41	63

Note. *M*—Maximum number of questions; *m*—Minimum number of questions = number of points that could be eliminated; EF—Cardinality of resource efficient set of points.

TABLE II

Percentages of Points to Eliminate Per Stage

Stage	Mean	Range
4	0.1264	0.0000–0.1875
5	0.1978	0.0312–0.5312
6	0.3141	0.1111–0.4250
7	0.4194	0.2941–0.5517
8	0.5306	0.3493–0.6546
9	0.5671	0.2571–0.7032
10	0.6211	0.3486–0.7040

utility function is linear and additive. The scheme includes the use of the subproblems suggested in Section 4.1 to simulate responses of the decision maker, and it also employs sets of lower and upper bounds for the set of efficient policies. The sets of lower bounds are composed of feasible solutions obtained by using the heuristic of Loulou and Michaelides [9] for 0–1 problems, and the sets of upper bounds are formed by setting the remaining variables to their upper bound (at each stage). All the problems are 0–1 bicriterion multidimensional knapsack problems. Table III illustrates the behavior of the sets of resource-efficient policies using the hybrid dynamic programming recursions developed in Villarreal and Karwan [17] and the interactive procedure with questioning sessions each three and four stages. A maximum of questions was prespecified. These are 15 and 25 questions in total. Obseve that the use of one more session resulted in more policies eliminated. Also, notice that at each of the stages at which questioning sessions are employed, the size of the set of resource-efficient policies becomes relatively small compared to what it would have been if no interactive sessions were used. This decrease is not as important in the early stages (3, 4, and 6) compared to that achieved in stages 8 and 9. No difference in the performance of the interactive algorithm is shown with the increase in the number of questions from 15 to 25.

As pointed out, in the last sample of problems, the preference problem used to simulate responses is employed only after 3 and 4 stages. In order to see the effect of using it more often, a sample of four problems was solved with the interactive procedure previously and the hybrid dynamic programming scheme of Villarreal and Karwan [17]. All the problems have four constraints, fifteen variables, and are 0–1 bicriterion multidimensional knapsack problems with b values of 0.50 the sum of the associated row coefficients. In eah problem, the interactive sessions are simulated at stages 3, 6, and 9 to 15. Table IV shows that the sets of resource efficient policies are decreased significantly after stage 9. In all the problems, the subsets of

TABLE III

Illustration of the Sizes of the Subsets of Resource Efficient Policies under Various Approaches

Problem number		Stage number 3	4	6	8	9	EFP	FATH	questions
1	IBB3	8	15	42	155	97	5	135	15
	IBB3	8	15	42	155	97	5	135	25
	IBB4	8	14	49	79	120	5	99	15
	HDP	8	14	44	93	113	5	—	—
2	IBB3	6	11	22	47	43	4	33	15
	IBB3	6	11	22	47	40	4	36	25
	IBB4	8	8	28	40	55	4	24	15
	HDP	6	8	23	42	50	4	—	—
3	IBB3	7	13	36	120	122	10	108	15
	IBB3	7	13	36	120	121	10	109	25
	IBB4	8	11	40	79	146	10	54	15
	HDP	7	13	44	110	167	10	—	—

Note. EFP—number of efficient points; FATH—number of points eliminated by preference; IBB3—interactive branch and bound with sessions each 3 stages; IBB4—interactive branch and bound with sessions each 4 stages; HDP—hybrid dynamic programming procedure.

resource-efficient policies surpassed a maximum level prespecified a priori of 2000 elements, before reaching stage 15, when the hybrid dynamic programming method is used. This illustrates that using the interactive procedure in a manner in which the preference problem is employed stagewise, will help to reduce the resource-efficient sets, keeping the storage requirements at reasonable levels. An important disadvantage is that the time necessary to solve the preference problems and to form the sets $F^*_{(m,\cdot)}$ is significant. The average time per stage (after stage 9) to carry out both tasks (for the sample of problems) is 31.98 cpu sec. The main component is the time spent constructing the sets $F^*_{(m,\cdot)}$, which corresponds to 88% of the total.

Table V shows further initial evidence that using 25 questions instead of 15, in an attempt to further decrease the multiplier space is not successful. This seems to indicate that for bicriterion problems, one does not need many questions to achieve a reasonable decrease of the multiplier space, or that the questions are associated with very similar policies and hence, most of the constraints added are redundant. All the problems are 0–1 bicriterion

TABLE IV

Comparison between the Interactive Procedure and the Hybrid Dynamic Programming Recursions

Problem no.		Stage No. 3	6	9	10	11	12	13	14	15	FATH	Solution time	F SET	Ave. time F time	LAMB	Total
1	IBB	8	34	97	154	220	267	348	328	349	728	202.26	1674	24.20	4.23	28.43
	HDP	8	64	398	761	1247	1332	**	**	**	—	—	—	—	—	—
2	IBB	8	47	154	262	267	261	370	*	*	594	220.00	1221	31.36	3.48	34.84
	HDP	8	62	329	581	945	1356	**	**	**	—	—	—	—	—	—
3	IBB	8	64	252	326	334	488	*	*	*	591	183.00	2334	37.79	5.78	43.58
	HDP	8	64	414	742	1180	1338	**	**	**	—	—	—	—	—	—
4	IBB	6	45	149	236	307	233	190	283	347	456	150.79	820	19.20	1.89	21.10
	HDP	8	64	395	646	839	900	793	1166	**	—	—	—	—	—	—

Note. IBB—interactive branch and bound; HDP—hybrid dynamic programming recursion; FATH—number of points eliminated; F SET—average time/stage to form sets $F^*_{(m,\cdot)}$; LAMP—average time-stage to solve preference problems; *—*time limit*; **-*maximum* size of resource-efficient set (2000) surpassed.

TABLE V

Illustration of the Sizes of the Subsets of Resource Efficient Policies for Two Numbers of Questions

Problem number		Stage number						FATH
		3*	6*	9*	10	11	12	
1	Q15	6	25	23	24	19	27	18
	Q25	6	25	23	24	19	27	18
2	Q15	5	27	95	168	281	462	52
	Q25	5	27	93	165	275	452	54
3	Q15	8	27	76	133	186	247	99
	Q25	8	27	76	133	186	247	99

Note. Q15—maximum of 15 questions; Q25—maximum of 25 questions; FATH—number of points eliminated; *—stages at which interactive sessions were carried out.

TABLE VI

Comparison of the Sizes of the Subsets of Resource Efficient Policies for the Same Problem with Different Set of Preference Weights

Problem number		M1	M2	Stage number			FATH
				3	6	9	
$b = 0.50$	1	0.50	0.50	6	25	23	18
		0.25	0.75	6	23	19	23
	2	0.50	0.50	5	27	95	52
		0.25	0.75	5	31	90	72
	3	0.50	0.50	8	27	76	99
		0.25	0.75	8	29	81	101
$b = 0.75$	4	0.50	0.50	8	37	112	38
		0.25	0.75	8	37	112	38
	5	0.50	0.50	5	13	32	21
		0.25	0.75	5	12	32	18
	6	0.50	0.50	8	35	87	46
		0.25	0.75	8	32	85	46

Note. FATH—number of points eliminated; M1—value of first preference weight; M2—value of second preference weight.

multidimensional knapsack problems with four constraints, twelve variables, 90% density, and b values of 0.50 the sum of the associated row coefficients.

Table VI shows that different results can be achieved using the interactive procedure for solving the same problem, when different utility functions are considered. The problems solved have the same characteristics of those of Table V with b values of 0.50 and 0.75 the sum of the associated row coefficients.

In this paper, it has been shown that multicriteria discrete programming problems can be solved by imbedding an interactive mode in the dynamic programing framework, provided that several properties of preference structures are satisfied. It was illustrated that using interactive sessions helps to decrease the sizes of the sets of resource-efficient policies. It was also seen that using the sets $F^*_{(m,\cdot)}$ in the interactive procedure, leads to a significant amount of time spent in constructing them. Finally, it is pointed out that several degrees of success could be achieved in reducing the sizes of the sets of resource-efficient policies when solving the same problem with different utility functions (see Table VI).

Further analysis and research should focus on the behavior of the multiplier space with respect to variations in the number of questions posed to the decision maker, and on the generation of sets more effective than the sets $F^*_{(m,\cdot)}$, where construction requires a significant portion of the time required for solving the total problem.

REFERENCES

1. 1. R. E. BELLMAN, "Dynamic Programming," Princeton Univ. Press, Princeton, N. J., 1957.
2. R. BENAYOUM, J. MONTGOLFIER, J. TERGNY, AND O. LARITCHEV, Linear programming with multiple objectives: STEP method (STEM)," *Math. Programming* **1** (1971).
3. V. CHANKONG AND Y. Y. HAIMES, The interactive worth trade-off (ISWT) method of multiobjective decision making, paper presented at the Conference on Multicriteria Problem Solving: Theory, Methodology, and Practice, Buffalo, N. Y., 1977.
4. J. S. DYER, Interactive goal programming, *Management Sci.* **19** (1972).
5. P. C. FISHBURN, "Utility Theory for Decision Making," Wiley, New York, 1970.
6. R. S. GARFINKEL AND G. L. NEMHAUSER, "Integer Programming," Wiley, New York, 1972.
7. A. M. GEOFFRION, J. S. DYER, AND A. FEINBERG, An interactive approach for multicriterion optimization with an application to the operations of an academic department, *Management Sci.* **19** (1972).
8. M. S. LEE, Interactive integer goal programming: Methods and applications, paper presented at the Conference on Multiple Criteria Problem Solving: Theory, Methodology, and Practice, Buffalo, N. Y., 1977.
9. R. LOULOU AND E. MICHAELIDES, "New Greedy-like Heuristic for the Multidimensional 0–1 Knapsack Problem," Working Paper, McGill University, 1977.
10. R. E. MARSTEN AND T. L. MORIN, A hybrid approach to discrete mathematical programming, *Math. Programming* **14** (1978).

11. L. G. MITTEN, Composition principles for synthesis of optimal multistage processes. *Oper. Res.* **12** (1964).
12. L. G. MITTEN, Preference order dynamic programming, *Management Sci.* **21** (1974).
13. T. L. MORIN AND R. E. MARSTEN, An algorithm for nonlinear knapsack problems, *Management Sci.* **22** (1976).
14. T. L. MORIN AND A. M. O. ESOGBUE, The imbedded state space approach to reducing dimensionality in dynamic programs of higher dimensions, *J. Math. Anal. Appl.* **48** (1974).
15. G. L. NEMHAUSER, "Introduction to Dynamic Programming," Wiley, New York, 1966.
16. B. VILLARREAL AND M. H. KARWAN, Dynamic programming approaches for multicriteria integer programming, paper presented at the TIMS/ORSA Joint National Meeting at New York, May 1978.
17. B. VILLARREAL AND M. H. KARWAN, Multicriteria integer linear programming: A hybrid dynamic programming recursive approach, paper presented at the TIME/ORSA Joint National Meeting at New Orleans, May 1979.
18. B. VILLARREAL AND M. H. KARWAN, "Multicriteria Integer Linear Programming: Some Extensions," Working Paper No. 78–5, State University of New York at Buffalo.
19. B. VILLARREAL, M. H. KARWAN, AND S. ZIONTS, "An Interactive Branch and Bound Procedure for Multicriteria Integer Linear Programming," to appear in the Proceedings of the Third Conference on Multiple Criteria Decision Making-Theory and Application, Königswinter, West Germany, August 20–24, 1979.
20. S. ZIONTS, "Linear and Integer Programming," Prentice–Hall, Englewood Cliffs, N. J., 1974.
21. S. ZIONTS, Integer linear programming with multiple objectives, *Ann. Discrete Math.* **1** (1977).
22. S. ZIONTS AND J. WALLENIUS, An interactive programming methodfor solving the multiple criteria problem, *Management Sci.* **22** (1976).