MANAGEMENT SCIENCE
Vol. 22, No. 10, June, 1976
Printed in U.S.A.

AN ALGORITHM FOR NONLINEAR KNAPSACK PROBLEMS*

THOMAS L. MORIN† AND ROY E. MARSTEN‡

An algorithm which recursively generates the complete family of undominated feasible solutions to separable nonlinear multidimensional knapsack problems is developed by exploiting discontinuity preserving properties of the maximal convolution. The "curse of dimensionality", which is usually associated with dynamic programming algorithms, is successfully mitigated by reducing an M-dimensional dynamic program to a 1-dimensional dynamic program through the use of the imbedded state space approach. Computational experience with the algorithm on problems with as many as 10 state variables is also reported and several interesting extensions are discussed.

1. Introduction

Consider the following problem: find $x \in R^N$ so as to

$$\text{maximize} \quad \sum_{j=1}^{N} r_j(x_j), \tag{1}$$

$$\text{subject to} \quad \sum_{j=1}^{N} g_{ij}(x_j) \leqslant b_i, \qquad i = 1, 2, \ldots, M,$$

$$x_j \in S_j, \qquad j = 1, 2, \ldots, N,$$

where $(\forall j)\ S_j = \{0, 1, 2, \ldots, K_j\}$. For notational simplicity the maximum permissible value for each variable is taken to be the same, namely K, though this is by no means essential. We assume that $(\forall j)\ r_j : S_j \to R_+$ is nondecreasing with $r_j(0) = 0$, that $(\forall ij) \cdot g_{ij} : S_j \to R_+$ with $g_{ij}(0) = 0$, and $b = (b_1, \ldots, b_M) \geqslant 0$. WLOG we assume $r_j(k) = 0$ if $g_{ij}(k) = 0$ for $1 \leqslant i \leqslant M$.

We shall refer to (1) as the *separable nonlinear multidimensional knapsack problem* (NKP). The familiar (one-dimensional) knapsack problem and the multidimensional 0/1 knapsack problem are special cases for which $(\forall j)\ r_j(x_j) = c_j x_j$, $(\forall ij)\ g_{ij}(x_j) = a_{ij} x_j$ and K is taken as the smallest integer such that $(\forall j)\ g_{1j}(K+1) > b_1$, or $K = 1$, respectively. (A number of other variants of the knapsack problem, e.g. [8], [30], and [34], as well as variants of the "optimal distribution of effort problem" [19], [32], and other nonlinear integer programming problems [12], are also special cases of the (NKP).) These problems have received considerable attention because of their numerous applications to resource allocation, capital budgeting [6], [20], [24], [26], and cutting stock problems [15], [16], as well as to integer programming theory [5], [31]. Furthermore, a number of dynamic programming algorithms have been developed for the solution of special cases of the (NKP) [2], [15]–[18], [24], [31], [33]–[35]. In general, however, the usefulness of conventional dynamic programming algorithms in the solution of problems in which M is greater than 2 or 3 has been seriously limited by the "curse of dimensionality" [2]. We shall present an algorithm (M&MDP) for the solution of the (NKP) which successfully mitigates the dimensionality problem by exploiting properties of the maximal convolution [2], [9], [22], [28] through the use of the imbedded state space approach [23]. M&MDP recursively generates the complete family of undominated feasible solutions to the (NKP). Our approach has been

* Processed by Professor Arthur M. Geoffrion, Departmental Editor for Integer and Large-Scale Programming; received May 7, 1974, revised September 10, 1974, February 5, 1975. This paper was with the authors 3 months for revision.

† The Technological Institute, Northwestern University.
‡ Sloan School of Management, Massachusetts Institute of Technology.

motivated both by the theoretical work of Haymond [18] and by the computational success of both the Nemhauser and Ullman algorithm [24] and the Weingartner and Ness algorithm [33] in the solution of (linear 0/1) capital budgeting problems.

The M&MDP algorithm is developed in §2. The computational burden is seen to rest primarily on the task of detecting and eliminating solutions which are "inefficient". A computationally effective way of doing this is derived and the resulting computer implementation of the algorithm is discussed in §3. An illustrative numerical example problem is solved in §4. Computational experience with M&MDP is reported in §5, and some ideas on further extensions of the imbedded state space approach are presented in §6.

2. Development of the M&MDP Algorithm

Consider the following functional equations of a dynamic programming formulation of the (NKP). For any $\beta = (\beta_1, \ldots, \beta_M)$ in the state space

$$\Omega = \{\beta \in R^M \mid 0 \leqslant \beta \leqslant b\} \tag{2}$$

we have

$$f_n(\beta) = \max_{y \in Y(\beta)}\{v_n(y) + f_{n-1}(\beta - y)\} \tag{3}$$

for $n = 1, 2, \ldots, N$, with the boundary condition

$$f_0(\beta) = 0 \quad \forall \beta \geqslant 0, \tag{4}$$

where

$$f_n(\beta) = \max\{\textstyle\sum_{j=1}^n r_j(x_j) \mid \sum_{j=1}^n g_{ij}(x_j) \leqslant \beta_i \ \forall i \ \text{ and } x_j \in S_j, j = 1, 2, \ldots, n\}, \tag{5}$$

$$Y(\beta) = \{y \in R^M \mid 0 \leqslant y \leqslant \beta\}, \tag{6}$$

$$v_n(y) = \max_{x_n \in S_n}\{r_n(x_n) \mid g_{in}(x_n) \leqslant y_i \ \forall i\} \tag{7}$$

and vector inequalities are taken element-wise. The solution of the functional equations for $f_N(b)$ and the subsequent (policy) reconstruction process to determine $x^*(b)$ is straightforward. But if M is greater than 2 or 3, the dimensionality of the state space Ω may present a serious computational problem. However, we can effect a dramatic reduction in dimensionality by exploiting the discontinuity preserving properties of the "maximal convolution" defined by (3).

THEOREM 1. *For each $n = 1, 2, \ldots, N$, $v_n(y)$, $f_n(\beta)$, and $f_{n-1}(\beta - y)$ are nondecreasing step functions on Ω. Moreover, if the respective domain sets of points of discontinuity of f_n, v_n and f_{n-1} are denoted by F_n, V_n, and F_{n-1}, and $F_0 = \{0\}$ where 0 denotes an M-dimensional vector, then we have the following recurrence relation*

$$F_n \subseteq \{V_n \circledast F_{n-1}\} \subseteq \Omega, \qquad n = 1, \ldots, N, \tag{8}$$

where $V_n \circledast F_{n-1}$ denotes the set obtained by forming all sums of exactly one element of V_n and exactly one element of F_{n-1}.

PROOF. That $f_n(\beta)$ is a nondecreasing step function $\forall \beta \in \Omega$ follows from the fact that $v_n(y)$ is a nondecreasing step function $\forall y \in \Omega$ and the properties of the maximum operator by induction on n. The recurrence relation (8) can be established by first demonstrating the reverse containment of complements, i.e., $z \notin \{V_n \cup F_{n-1} \cup (V_n \circledast F_{n-1})\} \Rightarrow f_n(\beta)$ is continuous at $\beta = z$. This part of the proof parallels the (rather lengthy) proof of the one-dimensional case given by Haymond in [18], *mutatis mutandis*, and, hence, is omitted. Then, noting that the assumptions $r_j(0) = 0 \ \forall j$ and $g_{ij}(0) = 0 \ \forall ij$ when combined with the initial condition $F_0 = \{0\}$ on $(8) \Rightarrow V_n \cup F_{n-1} \subseteq (V_n \circledast F_{n-1})$ completes the proof. Q.E.D.

As an immediate consequence of the theorem we have

COROLLARY 1. *$\exists z \in F_N$ such that $f_N(b) = f_N(z)$.*

Therefore, we only have to calculate $f_n(\beta)$ for $\beta \in F_n$, and F_n can be determined recursively from V_n and F_{n-1} using (8). Furthermore, in this calculation we can usually eliminate certain elements of $\{V_n \circledast F_{n-1}\}$ as being either inefficient or infeasible, thereby reducing the cardinality of F_n. We have reduced an M-dimensional dynamic program defined on Ω to a one-dimensional dynamic program defined on the sequence of *imbedded state spaces* $F_0, F_1, \ldots, F_N \subseteq \Omega$.

The method of generating these successive imbedded state spaces will now be described. For each $n = 1, 2, \ldots, N$, the set V_n of points of discontinuity of v_n on Ω may be obtained by inspection. Clearly

$$V_n \subseteq \{\gamma^0, \gamma^1, \ldots, \gamma^K\} \tag{9}$$

where

$$\gamma^k = (g_{1n}(k), \ldots, g_{Mn}(k)), \qquad k = 0, 1, \ldots, K. \tag{10}$$

The point γ^k belongs to V_n unless $\gamma^k \notin \Omega$ or unless there exists a $k' \neq k$, $0 \leqslant k' \leqslant K$, such that

$$g_{in}(k') \leqslant g_{in}(k), \qquad i = 1, \ldots, M, \quad \text{and} \tag{11}$$

$$r_n(k') \geqslant r_n(k), \tag{12}$$

with at least one strict inequality among (11) and (12). When this is the case we say that $\gamma^{k'}$ *dominates* γ^k and that γ^k is *inefficient*. When $\gamma^k \notin \Omega$ we say that γ^k is *infeasible*. Letting $\bar{S}_n$ contain the indices of the undominated feasible points, then, we have

$$V_n = \{\gamma^k \mid k \in \bar{S}_n\}, \quad \text{and} \tag{13}$$

$$v_n(\gamma^k) = r_n(k) \quad \text{for } k \in \bar{S}_n. \tag{14}$$

Labelling the elements of F_{n-1} as $\{\beta^0, \beta^1, \ldots, \beta^P\}$, where $\beta^0 = 0$, gives

$$V_n \circledast F_{n-1} = \{\gamma^k + \beta^p \mid k \in \bar{S}_n, 0 \leqslant p \leqslant P\}. \tag{15}$$

If there exists $i \in \{1, 2, \ldots, M\}$ such that

$$g_{in}(k) + \beta_i^p > b_i \tag{16}$$

then $\gamma^k + \beta^p \notin \Omega$ and hence $\gamma^k + \beta^p \notin F_n$. As above, we say that $\gamma^k + \beta^p$ is infeasible if it falls outside of Ω.

If, on the other hand, $\exists k, k' \in \bar{S}_n$ and $0 \leqslant p, p' \leqslant P$ such that

$$g_{in}(k') + \beta_i^{p'} \leqslant g_{in}(k) + \beta_i^p, \qquad i = 1, \ldots, M, \quad \text{and} \tag{17}$$

$$r_n(k') + f_{n-1}(\beta^{p'}) \geqslant r_n(k) + f_{n-1}(\beta^p), \tag{18}$$

with at least one strict inequality among (17) and (18), then $\gamma^k + \beta^p$ is dominated by $\gamma^{k'} + \beta^{p'}$ and cannot be an element of F_n. By eliminating all infeasible and dominated points from $V_n \circledast F_{n-1}$ we obtain F_n. For each $\gamma^k + \beta^p \in F_n$, then, we have

$$f_n(\gamma^k + \beta^p) = r_n(k) + f_{n-1}(\beta^p). \tag{19}$$

The origin belongs to F_n for all $n = 0, 1, \ldots, N$ and will be denoted β^0. Notice that $0 \in \bar{S}_n$ for all $n = 1, \ldots, N$ and that

$$\gamma^0 = (g_{1n}(0), \ldots, g_{Mn}(0)) = (0, \ldots, 0) = \beta^0 \tag{20}$$

by our assumption that $(\forall ij)\ g_{ij}(0) = 0$. It follows that every element of F_{n-1}, unless dominated, is also an element of F_n since $\gamma^0 + \beta^p = \beta^p$. The following algorithm uses the feasibility and dominance tests to construct the successive imbedded state spaces and terminates with the complete family of undominated feasible solutions F_N.

M&MDP Algorithm

```
Step 1. Set $n = 0$, $F_0 = \{\beta^0\}$, and $f_0(\beta^0) = 0$.
Step 2. Set $n = n + 1$ and $k = 0$.
Step 3. If $n > N$, stop.
Step 4. Set $P = |F_{n-1}| - 1$ and label the points of $F_{n-1}$ as $\{\beta^0, \beta^1 \dots, \beta^P\}$.
Step 5. Set $F_n = F_{n-1}$.
Step 6. Set $k = k + 1$. If $k > K$, go to Step 2.
Step 7. If $k \notin \bar{S}_n$, go to Step 6.
Step 8. Set $p = 0$.
Step 9. If $\gamma^k + \beta^p \notin \Omega$, go to Step 13. (Feasibility test.)
Step 10. If $\gamma^k + \beta^p$ is dominated by some point already in $F_n$, go to Step 13.
Step 11. Set $F_n = F_n \cup \{\gamma^k + \beta^p\}$ and $f_n(\gamma^k + \beta^p) = r_n(k) + f_{n-1}(\beta^p)$.
Step 12. Set $F_n = F_n - \{$all points dominated by $\gamma^k + \beta^p\}$.
Step 13. Set $p = p + 1$. If $p > P$, go to Step 6. Otherwise, go to Step 9.
```

At Step 5, we are taking advantage of the fact that $\gamma^0 = \beta^0$ for all $1 \leqslant n \leqslant N$. Upon termination of this algorithm, we are in possession of F_N and of $f_N(\beta)$ for each $\beta \in F_N$. By Corollary 1, then, we may select $z \in F_N$ so that

$$f_N(z) = \max\{f_N(\beta) \mid \beta \in F_N\} = f_N(b). \tag{21}$$

The optimal values of $x_1, \dots, x_N$ that are associated with the point z can be reconstructed if we augment our algorithm with the following bookkeeping scheme. For each feasible, and as yet undominated, point $\gamma^k + \beta^p$ that is added to F_n at Step 11 we make an entry of the following type in a TRACE table:

$$\text{TRACE}(*, 1) = n, \quad \text{TRACE}(*, 2) = k, \quad \text{TRACE}(*, 3) = \rho, \tag{22}$$

where * denotes the current row of TRACE and ρ is the row number of the TRACE entry for β^p ($\rho = 1$ if $\beta^p = \beta^0$). This entry is kept even if the corresponding point is subsequently found to be dominated. Upon termination, we may reconstruct the x associated with any $\beta \in F_N$ (in particular z) as follows:

```
    Set $x(j) = 0$ for $j = 1, \dots, N$.
    Set $\rho$ = row number of the TRACE entry for $z$.
1.  $x(\text{TRACE}(\rho, 1)) = \text{TRACE}(\rho, 2)$.
    $\rho = \text{TRACE}(\rho, 3)$.
    If $\rho = 1$, stop.
    Otherwise, go to 1.
```

It is evident that the efficiency of the M&MDP algorithm depends very heavily on the method used at Steps 10 and 12 for determining if a new point dominates or is dominated by some point already in F_n. This will be explored in the next section.

3. Implementation

Let us first address the important question of how inefficient points are to be detected and eliminated. Our current implementation of the algorithm is based on the following two propositions. Suppose that we are at Step 10 of the algorithm and that $\gamma^k + \beta^p$ is the newly generated point under consideration.

PROPOSITION 1. *Any point in F_n which dominates $\gamma^k + \beta^p$ at Step* 10 *can be found between β^p and $\gamma^k + \beta^p$ on at least one dimension $i \in (1, 2, \ldots, M)$.*

PROOF. Let $\gamma^{k'} + \beta^{p'} \in F_n$. Since $\gamma^{k'} + \beta^{p'}$ was generated before $\gamma^k + \beta^p$, we know by construction that $k' \leqslant k$. Suppose that $\gamma^{k'} + \beta^{p'}$ dominates $\gamma^k + \beta^p$ so that (17) and (18) hold, with at least one strict inequality. We may assume that $\beta^p \neq \beta^{p'}$, since otherwise $\gamma^{k'}$ would dominate γ^k and that would contradict the definition of $\bar{S}_n$. We first show that $\gamma^{k'} + \beta^{p'}$ cannot dominate β^p. Suppose the contrary. Then

$$g_{in}(k') + \beta_i^{p'} \leqslant \beta_i^p, \qquad i = 1, \ldots, M. \tag{23}$$

Since $g_{in}(k') \geqslant 0$ we have

$$\beta_i^{p'} \leqslant \beta_i^p, \qquad i = 1, \ldots, M. \tag{24}$$

Now $\beta^{p'}$ cannot dominate β^p since both of these points belong to F_{n-1} and, since $\beta^{p'} \neq \beta^p$, at least one of the inequalities (24) must be strict. Consequently,

$$f_{n-1}(\beta^{p'}) < f_{n-1}(\beta^p). \tag{25}$$

Since $k' \leqslant k$ and r_n is nondecreasing, we have

$$r_n(k') \leqslant r_n(k). \tag{26}$$

Adding (25) and (26) yields

$$r_n(k') + f_{n-1}(\beta^{p'}) < r_n(k) + f_{n-1}(\beta^p) \tag{27}$$

which contradicts (18). Hence $\gamma^{k'} + \beta^{p'}$ cannot dominate β^p as well as $\gamma^k + \beta^p$.

Now (18) implies that

$$r_n(k') + f_{n-1}(\beta^{p'}) \geqslant f_{n-1}(\beta^p) \tag{28}$$

since $r_n(k) \geqslant 0$. Suppose that (23) holds. Then, since β^p is not dominated, we must have

$$g_{in}(k') + \beta_i^{p'} = \beta_i^p, \qquad i = 1, \ldots, M, \quad \text{and} \tag{29}$$

$$r_n(k') + f_{n-1}(\beta^{p'}) = f_{n-1}(\beta^p). \tag{30}$$

But then (18) implies that $r_n(k) = 0$ which means that $k \notin \bar{S}_n$ since $k > 0$ by construction. Hence (23) does not hold and there exists a $1 \leqslant i^* \leqslant M$ such that

$$\beta_{i^*}^p < g_{i^*n}(k') + \beta_{i^*}^{p'} \leqslant g_{i^*n}(k) + \beta_{i^*}^p \tag{31}$$

and the proof is complete. Q.E.D.

Now suppose that $\gamma^k + \beta^p$ is not dominated at Step 10. It is placed in F_n and any point that it dominates is discarded at Step 12.

PROPOSITION 2. *Any point in F_n which is dominated by $\gamma^k + \beta^p$ at Step* 12 *falls between β^p and $\gamma^k + \beta^p$ in objective function value.*

PROOF. Let $\gamma^{k'} + \beta^{p'} \in F_n$. If $\gamma^{k'} + \beta^{p'}$ is dominated by $\gamma^k + \beta^p$, then

$$g_{in}(k) + \beta_i^p \leqslant g_{in}(k') + \beta_i^{p'}, \qquad i = 1, \ldots, M, \quad \text{and} \tag{32}$$

$$r_n(k) + f_{n-1}(\beta^p) \geqslant r_n(k') + f_{n-1}(\beta^{p'}). \tag{33}$$

Since $g_{in}(k) \geqslant 0$, we have

$$\beta_i^p \leqslant g_{in}(k') + \beta_i^{p'}, \qquad i = 1, \ldots, M. \tag{34}$$

But $\gamma^{k'} + \beta^{p'}$ would not be in F_n if it were dominated by $\beta^p = \gamma^0 + \beta^p$, so we must have either

$$f_{n-1}(\beta^p) < r_n(k') + f_{n-1}(\beta^{p'}) \tag{35}$$

or else

$$f_{n-1}(\beta^p) = r_n(k') + f_{n-1}(\beta^{p'}), \quad \text{and} \tag{36}$$

$$\beta_i^p = g_{in}(k') + \beta_i^{p'}, \qquad i = 1, \dots, M. \tag{37}$$

It is clear from (32) that the latter case cannot arise unless $g_{in}(k) = 0$ for $1 \leqslant i \leqslant M$. But this means that $r_n(k) = 0$ and hence that $k \notin \bar{S}_n$ since $k > 0$ by construction. Combining (33) and (35) then,

$$f_{n-1}(\beta^p) < r_n(k') + f_{n-1}(\beta^{p'}) \leqslant r_n(k) + f_{n-1}(\beta^p) \tag{38}$$

and the proof is complete. Q.E.D.

We can take advantage of these two propositions by using a threaded list structure for each dimension of the state space and for the objective function. The point $\gamma^k + \beta^p$ is added to F_n at Step 11 of the algorithm by making an entry in the FLIST table.

$$\begin{aligned} \text{FLIST}(*, i) &= g_{in}(k) + \beta_i^p \quad \text{for } i = 1, 2, \dots, M, \\ &= g_{in}(k) + \text{FLIST}(q, i); \end{aligned} \tag{39}$$

$$\begin{aligned} \text{FLIST}(*, M+1) &= r_n(k) + f_{n-1}(\beta^p) \\ &= r_n(k) + \text{FLIST}(q, M+1); \end{aligned} \tag{40}$$

where * denotes the current row of the FLIST table and q is the row number of the entry for β^p. A second table, called the PERMUTE table, is maintained so that for each $i = 1, 2, \dots, M + 1$ and for any row t, we have either

$$\text{PERMUTE}(t, i) = 0 \tag{41}$$

which means that FLIST(t, i) is currently the maximum entry in column i of the FLIST table, or else

$$\text{FLIST}(t, i) \leqslant \text{FLIST}(\text{PERMUTE}(t, i), i). \tag{42}$$

The PERMUTE table is easy to update and it eliminates the sorting that would otherwise be required to detect dominance. In the course of updating column i of the PERMUTE table, we encounter all of the points currently in F_n that fall between β^p and $\gamma^k + \beta^p$ on dimension i. By Proposition 1, only such points can dominate $\gamma^k + \beta^p$. If $\gamma^k + \beta^p$ is not dominated by any of these points, for all $i = 1, 2, \dots, M$, then it can be added to F_n. Any points which *it* dominates, by Proposition 2, will be discovered when we update column $(M + 1)$ of PERMUTE. Thus, Propositions 1 and 2 greatly restrict the set of points against which the new point $\gamma^k + \beta^p$ must be tested. A listing of the complete FORTRAN program for the algorithm is available on request from the authors.

Other computational refinements are also possible. For example, the size of the lists of undominated solutions may be reduced by the use of fathoming criteria and relaxations [14]. For example, in the final stages, there may exist points in the F_n lists which could never lead to an optimal solution and, hence, could be fathomed. That is, we can eliminate any point $\beta^p \in F_{n-1}$ for which there exists a $\beta^{p'} \in F_{n-1}$ which satisfies the following fathoming criterion

$$f_{n-1}(\beta^{p'}) \geqslant f_{n-1}(\beta^p) + \textstyle\sum_{j=n}^{N} r_j(K). \tag{43}$$

In (43) $f_{n-1}(\beta^{p'})$ is a (crude) lower bound on $f_N(b)$ and $\sum_{j=n}^{N} r_j(k)$ is a (crude) upper bound on the residual problem from stage n to stage N. However, this type of refinement would not permit the generation of the complete family of undominated feasible solutions and was not incorporated in our current implementation. Such strategies are explored in detail in a subsequent paper [21].

For each point added to F_n at Step 11, we need a total of $2M + 6$ words of computer memory. If the point is later found to be dominated, then $2M + 4$ of these words can be re-used. The other two words make up the TRACE table entry. (The first column of the TRACE table (22) is replaced by N pointers, one to the first entry for each stage.) The total storage requirement for a given problem cannot be determined in advance. The worst possible case (no infeasibility or dominance) would have a requirement of $(2M + 6)(K + 1)^N$ words.

4. Example

The algorithm is demonstrated on one of Shapiro and Wagner's [31, pp. 336–337] simple test problems, with one additional constraint. This problem has $M = 2$, $N = 3$, $K = 3$ and $b = (8, 12)$. The values of the r_j and g_{ij} functions are tabulated as follows.

x_j	$r_1(x_1)$	$g_{11}(x_1)$	$g_{21}(x_1)$	$r_2(x_2)$	$g_{12}(x_2)$	$g_{22}(x_2)$	$r_3(x_3)$	$g_{13}(x_3)$	$g_{23}(x_3)$
0	0	0	0	0	0	0	0	0	0
1	2	1	2	3	2	3	$4\frac{1}{4}$	3	4
2	4	2	4	6	4	6	$6\frac{1}{4}$	6	8
3	$4\frac{3}{4}$	3	6	8	6	9	$6\frac{3}{4}$	9	12

Stage 1. $V_1 = \{(0, 0), (1, 2), (2, 4), (3, 6)\}$, $F_0 = \{(0, 0)\}$, $V_1 \circledast F_0 = \{(0, 0),$ $(1, 2), (2, 4), (3, 6)\}$. None of these points is infeasible or dominated, so $F_1 = \{(0, 0),$ $(1, 2), (2, 4), (3, 6)\}$. At the end of this stage, we would have

FLIST

0	0	0
1	2	2
2	4	4
3	6	$4\frac{3}{4}$

PERMUTE

2	2	2	1
3	3	3	2
4	4	4	3
0	0	0	4

TRACE

0	0	0
1	1	1
1	2	1
1	3	1

The fourth column appended to the PERMUTE table points to the corresponding entry in the TRACE table.

Stage 2. $V_2 = \{(0, 0), (2, 3), (4, 6), (6, 9)\}$, $V_2 \circledast F_1 = \{(0, 0), (1, 2), (2, 4),$ $(3, 6), (2, 3), (3, 5), (4, 7), (5, 9), (4, 6), (5, 8), (6, 10), (7, 12), (6, 9), (7, 11),$ $(8, 13), (9, 15)\}$. The points (8, 13) and (9, 15) are infeasible since $b = (8, 12)$. The point (3, 5) with return 5 dominates the point (3, 6) with return $4\frac{3}{4}$. Similarly (5, 8) with return 8 dominates (5, 9) with return $7\frac{3}{4}$, as well as (6, 9) with return 8. Finally, the point (6, 10) with return 10 dominates the point (7, 11) with return 10. Therefore, we have $F_2 = \{(0, 0), (1, 2), (2, 4), (2, 3), (3, 5), (4, 7), (4, 6), (5, 8), (6, 10), (7, 12)\}$. At the end of the second stage, the tables would appear as:

FLIST

0	0	0
1	2	2
2	4	4
X	X	X
2	3	3
3	5	5
4	7	7
6	10	10
4	6	6
5	8	8
7	12	$10\frac{3}{4}$

PERMUTE

2	2	2	1
3	5	5	2
5	6	6	3
X	X	X	X
6	3	3	5
7	9	9	6
9	10	10	7
11	11	11	11
10	7	7	9
8	8	8	10
0	0	0	12

TRACE

0	0	0
1	1	1
1	2	1
1	3	1
2	1	1
2	1	2
2	1	3
2	1	4
2	2	1
2	2	2
2	2	3
2	2	4

The X's in the FLIST and PERMUTE tables mark re-useable spaces which have not yet been filled.

Stage 3. $V_3 = \{(0, 0), (3, 4), (6, 8), (9, 12)\}$. $V_3 \circledast F_2$ contains 40 points, of which 18 are infeasible and 7 are dominated. This leaves F_3 with 15 points: $F_3 = \{(0, 0), (1, 2), (2, 4), (3, 4), (2, 3), (3, 5), (4, 7), (6, 10), (4, 6), (5, 8), (5, 7), (6, 9), (7, 11), (7, 10), (8, 12)\}$. At the end of the final stage, the tables look like:

FLIST

0	0	0
1	2	2
2	4	4
3	4	$4\frac{1}{4}$
2	3	3
3	5	5
4	7	7
6	10	10
X	X	X
X	X	X
X	X	X
4	6	$6\frac{1}{4}$
5	8	$8\frac{1}{4}$
5	7	$7\frac{1}{4}$
6	9	$9\frac{1}{4}$
7	11	$11\frac{1}{4}$
7	10	$10\frac{1}{4}$
8	12	$12\frac{1}{4}$

PERMUTE

2	2	2	1
3	5	5	2
5	4	4	3
6	6	6	13
4	3	3	5
7	12	12	6
12	14	14	7
15	17	17	11
X	X	X	X
X	X	X	X
X	X	X	X
13	7	7	14
14	15	15	15
8	13	13	16
16	8	8	17
17	18	18	18
18	16	16	19
0	0	0	20

TRACE

0	0	0
1	1	1
1	2	1
1	3	1
2	1	1
2	1	2
2	1	3
2	1	4
2	2	1
2	2	2
2	2	3
2	2	4
3	1	1
3	1	2
3	1	3
3	1	5
3	1	6
3	1	7
3	1	9
3	1	10

The zero in the third column of the PERMUTE table indicates that the maximum return is achieved at $z = (8, 12)$ with $f_3(8, 12) = f_3(b) = 12\frac{1}{4}$. To reconstruct the optimal x^* solution, we go to the TRACE entry for z in row 20. Here we find that $x_3^* = 1$. Proceeding to row 10 of the TRACE table, we find $x_2^* = 2$. Finally, row 2 yields $x_1^* = 1$. All other variables (none in this case) are zero and $x^* = (1, 2, 1)$.

Notice that although the state space for this problem contains $9 \times 13 = 117$ points, the sets F_1, F_2, and F_3 contain only 4, 10, and 15 points, respectively. Only 194 words of memory would be required, as opposed to the "worst possible" value of 640.

5. Computational Experience

Computational experience on a number of linear and nonlinear knapsack problems is summarized in Table 1. The execution times reported in this table are the CPU times for a FORTRAN IV code implemented on Northwestern University's CDC6400. The sources of the problems are as follows: problem 1 is Cord's problem [6]; problems 2 and 3 are Petersen's [26] problems 1 and 2; problems 4–8 are Petersen's problems 3–7 with reduced b vectors. Problems 9–11 have been constructed from Petersen's data on problems with 10 constraints by grouping the decision variables in terms of values of $g_{1j}(1)$ in the intervals $[0, 10)$, $[10, 100)$, and $[100, \infty)$, respectively, and setting the b vector large enough that any one of the decision variables could be positive ($= 1$). Similarly, problems 12–14 have been constructed from Petersen's data on problems with 5 constraints as well as his data on problems with 10 constraints (the latter by constructing two problems with 5 constraints from a 10-constraint problem by using the same objective function and constraints 1–5 and constraints 6–10, respectively) by grouping the decision variable in terms of values of $g_{1j}(1)$ (or $g_{6j}(1)$) in the intervals $[0, 10)$, $[10, 100)$ and $[100, \infty)$, respectively, and setting the b vector large enough so that any one of the decision variables could be positive ($= 1$).

Problems 15–23 were constructed using the data from problem 10 with concave (15–17), linear (18–20), and convex (21–23) objective functions and convex (15, 18, 21), linear (16, 19, 22), and concave (17, 20, 23) constraints. Problems 24–27 are problem 10 with the decision variables sequenced in the original order; in nonincreasing order of the ratio $\alpha_j = \max_i(g_{ij}(1)/b_i)$, a heuristic for sequencing the decision variable consuming the largest amount of resources first; in nondecreasing order of α_j; and in nondecreasing order of the ratio $r_j(1)/\alpha_j$, a heuristic for sequencing the "best" decision variables last; respectively. Fina''y, problems 28–36 are problem 10 with 9, 8, . . . , 1 constraints, respectively.

Inspection of Table 1 reveals that the performance of the M&MDP algorithm is directly related to the length of the list of efficient solutions; the smaller the list lengths the shorter the execution times. In fact, this factor is much more significant than the dimensionality of the state space. Shorter list lengths are encountered on problems in which (i) the components of the b vector are "small" relative to the constraint requirements of the decision variables, and (ii) the range of variation in constraint requirements is small. (Relative to these criteria, problem 1 and problems 9–36 were "easy" problems and problems 2–8 were "hard" problems.) For problems satisfying these criteria, M&MDP is even competitive in terms of computational efficiency with the best of the current (linear) integer programming codes on linear problems—see [13], for example. However, if in a linear problem, the b vector is such that approximately half of the decision variables were positive in an optimal solution, one of the good (linear) integer programming codes should probably be used since both the storage requirements and computational times of M&MDP might prove to be excessive. Alternatively on hard linear (or nonlinear) problems we can abandon the concept of generating the complete family of undominated feasible solutions and use either (1) a hybrid approach employing fathoming criteria as discussed in §3 to find the set of optimal solutions or (2) a conventional DP approach based upon the functional equation (3) together with a coarse grid approximation model to obtain an

TABLE 1
Summary of Computational Experience

Problem Number	Problem Size			Execution Time (sec.)	Final List Length	No. of Positive Variables in Optimal Solution
	N	M	K			
1	25	2	1	1.394	167	3
2	6	10	1	0.350	22	3
3	10	10	1	3.026	427	5
4	15	10	1	21.869	391	9
5	20	10	1	42.254	526	7
6	28	10	1	78.398	814	6
7	39	5	1	57.427	1043	4
8	50	5	1	192.140	2053	4
9	9	10	1	0.386	10	1
10	28	10	1	1.815	55	4
11	7	10	1	0.330	7	1
12	34	5	1	29.698	540	2
13	50	5	1	22.527	571	5
14	10	5	1	0.316	10	2
15	28	10	5	2.937	55	4
16	28	10	5	4.024	76	4
17	28	10	5	8.758	132	3
18	28	10	5	2.976	56	4
19	28	10	5	4.031	87	3
20	28	10	5	8.460	115	2
21	28	10	5	3.238	69	3
22	28	10	5	4.058	85	3
23	28	10	5	7.333	110	3
24	28	10	1	1.966	55	4
25	28	10	1	3.839	55	4
26	28	10	1	2.254	55	4
27	28	10	1	5.994	55	4
28	28	9	1	1.680	55	4
29	28	8	1	1.493	53	4
30	28	7	1	1.342	52	4
31	28	6	1	1.189	44	4
32	28	5	1	1.084	44	4
33	28	4	1	1.000	44	4
34	28	3	1	0.835	28	4
35	28	2	1	0.653	18	4
36	28	1	1	0.585	13	4

approximate solution to the (NKP). However, the latter approach becomes computationally intractable for large M ($\geqslant 4$) and, with the notable exception of [11], the effects of refining grid size have not been studied rigorously. (Notice that if the b vector is such that most of the decision variables in an optimal solution were positive, we could use M&MDP to solve efficiently the "dual", or "complementary", problem of which decision variables should be at zero level, i.e., excluded—see Pandit [25] or Weingartner and Ness [33].) However, even on "hard" linear problems, M&MDP has the valuable advantage of providing not only the optimal solution to the original problem in the final list, but also providing the optimal solution to all problems with smaller b vectors. This built-in sensitivity analysis is not available with the usual integer programming codes. Furthermore, M&MDP has the valuable capability of being able to provide exact solutions to knapsack problems in which the return functions or constraints or both are nonlinear as in problems 15–23.

Inspection of Table 1 also reveals that the computational time was dependent upon the input sequence of the decision variables. (A similar observation was made in [33].) In all of the problems, except for problems 24–27, the execution time reported was the smallest of (i) the execution time for the problem with the decision variables sequenced in nonincreasing order of the ratio, $\alpha_j = \max_i(g_{ij}(1)/b_i)$, a heuristic for sequencing the decision variables which "eat most", in terms of resources, first, and (ii) the execution time for the problem with the decision variables sequenced in nonincreasing order of the ratio $r_j(1)/\alpha_j$, a heuristic for sequencing the "best" decision variables first. (In all problems except for problem 3 and problems 5–8, the "best" first heuristic was found to be superior.) The dependence of the computational time on the input sequence is demonstrated in problem 10 and problems 24–27 of the table. These are all problem 10 but with different input sequencing of decision variables, as previously explained.

Finally, inspection of Table 1 reveals that the computational time is also related to the dimensionality of the state space. However, in M&MDP the computational time does not increase exponentially with the number of state variables as is usually the case with conventional dynamic programming algorithms. For example, the decrease in execution time observed in problems 10 and 28–36 is seen to be even less than linear. Furthermore, since the optimal solution did not change in problems 10 and 28–36, a major portion of the decrease in computational time was probably attributable to the fact that decreasing the number of constraints in this problem had the effect of shortening the list of efficient points.

6. Conclusion

We have presented and tested an imbedded state space dynamic programming algorithm for separable, nonlinear, multidimensional knapsack problems. In contrast to the usual limitation of DP to 2 or 3 dimensional state spaces, our results include problems with 10 dimensional state spaces. This increase in the range of applicability of DP is made possible by the imbedded state space approach.

It appears that the imbedded state space approach can also be applied to the solution of wider classes of problems in which the return functions are discontinuous and, in particular, where they are (or can be transformed to, as in the case treated herein) step functions—see [23], for example. The class of problems to which the imbedded state space approach is applicable appears to include not only separable additive returns, but also separable multiplicative returns as encountered in certain reliability problems [1], [2]—see [27] for example, and R & D capital budgeting problems, as well as separable logical returns, as those involving the infix operator $\bigvee$ ("disjunction"), which are encountered in fuzzy decision problems [4].

Finally, we note that relaxations and fathoming criteria (such as (43)) can be incorporated into M&MDP to produce a hybrid (DP-branch and bound) algorithm. Preliminary results with this hybrid algorithm have been quite encouraging [21].

Notice that M&MDP could be extended to handle knapsack problems involving multiple objective functions, and that the loading problem (optimally loading several knapsacks) can be formulated as a special case of the (NKP).

References

1. Bellman, R. E. and Dreyfus, S. E., "Dynamic Programming and the Reliability of Multicomponent Devices," *Operations Research*, Vol. 6 (1958), pp. 200–206.
2. ——— and ———, *Applied Dynamic Programming*, Princeton University Press, Princeton, N. J., 1962.
3. ——— and Karush, W., "Mathematical Programming and the Maximum Transform," *J. SIAM*, Vol. 10 (1962), pp. 550–567.
4. ——— and Zadeh, L. A., "Decision-Making in a Fuzzy Environment," *Management Science*, Vol. 17 (1970), pp. B-141–B-164.

5. BRADLEY, G., "Transformation of Integer Programs to Knapsack Functions," *Discrete Mathematics*, Vol. 1 (1971), pp. 29–45.
6. CORD, J., "A Method for Allocating Funds to Investment Projects When Returns are Subject to Uncertainty," *Management Science*, Vol. 11 (1964), pp. 335–341.
7. DANTZIG, G., "Discrete-Variable Extremum Problems," *Operations Research*, Vol. 5 (1957), pp. 266–277.
8. FAALAND, B., "Solution of the Value-Independent Knapsack Problem by Partitioning," *Operations Research*, Vol. 21 (1973), pp. 333–337.
9. FENCHEL, W., "Convex Cones, Sets and Functions," *Mimeographed Lecture Notes*, Princeton University, Princeton, N. J., 1951.
10. FOX, B., "Discrete Optimization Via Marginal Analysis," *Management Science*, Vol. 13 (1966), pp. 210–216.
11. ———, "Discretizing Dynamic Programs," *J. Optimization Theory and Applications*, Vol. 11 (1973), pp. 228–234.
12. GARFINKEL, R. S. AND NEMHAUSER, G. L., *Integer Programming*, Wiley-Interscience, New York, N. Y., 1972.
13. GEOFFRION, A. M., "An Improved Implicit Enumeration Approach for Integer Programming," *Operations Research*, Vol. 17 (1969), pp. 437–454.
14. ——— AND MARSTEN, R. E., "Integer Programming Algorithms: A Framework and State-of-the-Art Survey," *Management Science*, Vol. 18 (1972), pp. 465–491.
15. GILMORE, P. C. AND GOMORY, R. E., "Multi-Stage Cutting Stock Problems of Two or More Dimensions," *Operations Research*, Vol. 13 (1965), pp. 94–120.
16. ——— AND ———, "The Theory and Computation of Knapsack Functions," *Operations Research*, Vol. 14 (1966), pp. 1045–1074.
17. GREENBERG, H., "An Algorithm for the Computation of Knapsack Functions," *Journal of Mathematical Analysis and Applications*, Vol. 26 (1969), pp. 159–162.
18. HAYMOND, R. E., "Discontinuities in the Optimal Return in Dynamic Programming," *Journal of Mathematical Analysis and Applications*, Vol. 30 (1970), pp. 639–644.
19. KARUSH, W., "A General Algorithm for the Optimal Distribution of Effort," *Management Science*, Vol. 9 (1962), pp. 50–72.
20. LORIE, J. N. AND Savage, L. J., "Three Problems in Rationing Capital," *Journal of Business*, Vol. 28 (1955), pp. 229–239.
21. MARSTEN, R. E. AND MORIN, T. L., "A Hybrid Approach to Discrete Mathematical Programming," Sloan School of Management, Working Paper 838–76, March, 1976.
22. MOREAU, J. J., "Weak and Strong Solutions to Dual Problems," in *Contributions to Nonlinear Functional Analysis*, Zarantonello, E. H. (ed.), pp. 181–214, Academic Press, New York, 1971.
23. MORIN, T. L. AND ESOGBUE, A. M. O., "The Imbedded State Space Approach to Reducing Dimensionality in Dynamic Programs of Higher Dimensions," *Journal of Mathematical Analysis and Applications*, Vol. 48 (1974), pp. 801–810.
24. NEMHAUSER, G. L. AND ULLMAN, Z., "Discrete Dynamic Programming and Capital Allocation," *Management Science*, Vol. 15 (1969), pp. 494–505.
25. PANDIT, S. N. N., "The Loading Problem," *Operations Research*, Vol. 10 (1962), pp. 639–646.
26. PETERSEN, C. C., "Computational Experience with Variants of the Balas Algorithm Applied to the Selection of R & D Projects," *Management Science*, Vol. 13 (1967), pp. 736–750.
27. PROSCHAN, F. AND BRAY, T. A., "Optimal Redundancy Under Multiple Constraints," *Operations Research*, Vol. 13 (1965), pp. 800–814.
28. ROCKAFELLAR, R. T., *Convex Analysis*, Princeton University Press, Princeton, N. J., 1970.
29. SALKIN, H. H. AND DE KLUYVER, C. A., "The Knapsack Problem: A Survey," *Naval Research Logistics Quarterly*, Vol. 22 (1975), pp. 127–144.
30. SCHWARTZ R. E. AND DYM, C. L., "An Integer Maximization Problem," *Operations Research*, Vol. 19 (1971), pp. 548–550.
31. SHAPIRO, J. F. AND WAGNER, H. M., "A Finite Renewal Algorithm for the Knapsack and Turnpike Models," *Operations Research*, Vol. 15 (1967), pp. 319–341.
32. WAGNER, H. M., *Introduction to Operations Research*, Prentice-Hall, Englewood Cliffs, N. J., 1969.
33. WEINGARTNER, H. M. AND NESS, D. N., "Methods for the Solution of the Multi-Dimensional 0/1 Knapsack Problem," *Operations Research*, Vol. 15 (1967), pp. 83–103.
34. WEINSTEIN, I. J. AND YU, O. S., "Comment on an Integer Maximization Problem," *Operations Research*, Vol. 21 (1973), pp. 648–650.
35. YORMARK, J. S., "Accelerating Greenberg's Method for the Computation of Knapsack Functions," presented at the Joint ORSA/TIMS/AIIE National Meeting, Atlantic City, N. J., November 1972.