JOURNAL OF MATHEMATICAL ANALYSIS AND APPLICATIONS 26, 454–459 (1969)

A Dynamic Programming Solution to Integer Linear Programs

HAROLD GREENBERG

Department of Operations Analysis, Naval Postgraduate School, Monterey, California 93940

Submitted by Richard Bellman

I. INTRODUCTION

In this paper we present an algorithm for the solution to the integer linear programming problem: find $X = (x_1, \dots, x_n)$ that

minimizes

$$CX$$

when

$$AX = B$$

$$0 \leqslant x_j \leqslant b_j, \qquad x_j \text{ integer}, \qquad j = 1, \dots, n, \tag{1}$$

where C is an n-vector, B is an m-vector, A is an m by n matrix, and the b_j are integers. We assume that the components of C, B, and the elements of A are integers. We consider the case where some or all of the variables have upper bounds. Thus, the solution to (1) includes the case where the x_j are restricted to being either zero or one.

The continuous linear programming solution to (1) is found first. If the continuous solution is fractional, we develop a linear congruence that is added as a constraint. We then form an equivalent knapsack problem, which is solved by means of a dynamic programming enumeration.

The equivalent knapsack problem has been treated in [1] and [2] to round the non-integer values of the variables into an optimal integer solution after obtaining the continuous solution to (1). The method of [1] requires that the fractional parts of the coefficients, obtained in the continuous solution, attain a certain group property. The method of [2] is presented in a network context of considerable complexity. Neither of the methods achieves solution to the integer program (1) in all cases. In contrast, the method presented here solves the equivalent knapsack problem in a simple manner and then provides a solution to the integer program (1).

II. The Equivalent Knapsack Problem and Integer Solution

We use a bounded variable linear programming technique [3] to transform (1) to the equivalent problem:

minimize

$$d + \sum_{j \in H} c_j x_j$$

$$X_G + \sum_{j \in H} \alpha_j x_j = \alpha_0$$

$$0 \leqslant x_j \leqslant b_j, \qquad x_j \text{ integer}, \qquad j = 1, \ldots, n \tag{2}$$

where the vector $X_G = \{x_i \mid i \in G\}$, G is the set of indices of the basic variables, and H is the set of indices of the nonbasic variables. Further, some of the nonbasic variables may be at their upper bounds in the continuous solution of (1). For those variables we would have their corresponding c_j values as non-positive. We also have $c_j \geqslant 0$ for nonbasic variables that are at the zero value. For ease of computation, we make the transformation $x_j' = b_j - x_j$ for those nonbasic variables at their upper bound. Thus we may assume a form like (2) with all $c_j \geqslant 0$ and all nonbasic variables at zero values. The vectors α_j ($j = 0$ and $j \in H$) are column vectors. The components of α_0 are then nonnegative. If α_0 is all integer, then $x_j = 0$ ($j \in H$); $X_G = \alpha_0$ is an optimal solution (if any of the x_j, $j \in H$, are really the x_j' under the transformation, we naturally would have $x_j = b_j$).

If any of the α_0 are fractional, then the equivalent knapsack problem is [1]: find the x_j that

minimize

$$\sum_{j \in H} c_j x_j$$

when

$$\sum_{j \in H} \beta_j x_j = \beta_0 \qquad (\text{mod } 1)$$

$$0 \leqslant x_j \leqslant b_j, \qquad x_j \text{ integer}, \qquad j \in H \tag{3}$$

where the β_j are the columns of fractional parts of the α_j from (2).

To compute the solutions to (3), we form the knapsack function

$$F(\beta) = \min \left[\sum_{j \in H} c_j x_j \,\middle|\, \sum_{j \in H} \beta_j x_j = \beta \ (\text{mod } 1), x_j \leqslant b_j \right], \tag{4}$$

which may be written as the dynamic programming recursion

$$F(\beta) = \min_j [c_j + F(\beta - \beta_j)], \tag{5}$$

where the arguments of F are taken modulo 1.

The recursion in (5) may be solved as a simple enumeration by noting that

$$F(\beta_r) = c_r \tag{6}$$

where

$$c_r = \min_{j \in H} c_j \tag{7}$$

We then form $F(\beta - \beta_r)$ by replacing β by $\beta - \beta_r$ in (5), and we substitute the result for the $F(\beta - \beta_r)$ term on the right side of (5). We then can produce another immediate solution. We also take care that the upper bounds b_j are not violated while performing the enumeration.

We note that the enumeration process, using (7), produces increasing values of the objective function (3) while obtaining all possible integer solutions to the congruence in (3). Thus to solve the integer program give by (2), we stop the enumeration when $F(\beta_0)$ is calculated. We then check to see if the constraints in (2) are satisfied by the $x_j (j \in H)$ values found in the enumeration. If the constraints are not satisfied, we simply continue the enumeration until another $F(\beta_0)$ is produced. This entire procedure is contained in the following algorithm:

1. Suppose the indices in H are 1, 2,..., m, then list the values of the problem as

1	2	3	$\cdots$	m
c_1	c_2	c_3	$\cdots$	c_m
β_1	β_2	β_3		β_m
		$x_j = 0$		

Go to 2.

2. Given the list, find $c_r = \min c_j$ for all unmarked columns in all sections. If $\beta_r = \beta_0$, mark the column and go to 4. Otherwise, mark the column and go to 3.

3. Add a new section of columns to the list, if possible, as follows:

(a) Calculate $c_j' = c_r + c_j$, $\beta_j' = \beta_r + \beta_j \pmod 1$ for all values $j \in H$ (i.e., the values are taken from the list in step 1) where $x_j < b_j$ for $j \neq r$ and where $x_r + 1 < b_r$ for $j = r$ for the section containing the newly marked column.

(b) Add the columns headed by j in the new section with values c_j' and β_j'.

(c) Underneath the section added write the x_j values from the section containing the newly marked column. Increase x_r by one for the section. Go to 2.

(4) Take as a trial solution the values of the variables found below the section where $\beta_r = \beta_0$ appears, with x_r increased by one. See if the constraints in (2) are satisfied. If they are, the solution found is the optimal integer solution to (1). If not, go to 3.

This completes the algorithm. An integer solution is sure to be found because all integer solutions to the congruence in (3) are systematically produced in order of increasing objective function value.

Example. We take the problem given in [4].
minimize

$$5x_1 + 7x_2 + 10x_3 + 3x_4 + x_5$$

when

$$\begin{aligned} x_1 - 3x_2 + 5x_3 + x_4 - 4x_5 &\geqslant 2 \\ -2x_1 + 6x_2 - 3x_3 - 2x_4 + 2x_5 &\geqslant 0 \\ -x_2 + 2x_3 - x_4 - x_5 &\geqslant 1 \end{aligned}$$

$$0 \leqslant x_j \leqslant 1, \qquad x_j \text{ integer}, \qquad j = 1,\dots, 5.$$

We introduce surplus variables $x_6 \geqslant 0$, $x_7 \geqslant 0$, and $x_8 \geqslant 0$ (i.e., with ∞ as upper bounds) and find the continuous solution and equivalent problem:
minimize

$$9 + \tfrac{93}{9} x_1 + \tfrac{156}{9} x_4 + \tfrac{42}{9} x_5 + \tfrac{24}{9} x_7 + \tfrac{137}{9} x_8$$

when

$$\begin{aligned} -\tfrac{7}{9} x_1 - \tfrac{28}{9} x_4 + \tfrac{13}{9} x_5 + x_6 + \tfrac{1}{9} x_7 - \tfrac{7}{3} x_8 &= \tfrac{1}{3} \\ -\tfrac{2}{9} x_1 + x_3 - \tfrac{8}{9} x_4 - \tfrac{4}{9} x_5 - \tfrac{1}{9} x_7 - \tfrac{2}{3} x_8 &= \tfrac{2}{3} \\ -\tfrac{4}{9} x_1 + x_2 - \tfrac{7}{9} x_4 + \tfrac{1}{9} x_5 - \tfrac{2}{9} x_7 - \tfrac{1}{3} x_8 &= \tfrac{1}{3} \end{aligned}$$

$$0 \leqslant x_j \leqslant 1, \qquad x_j \text{ integer}, \qquad j = 1,\dots, 5$$

$$0 \leqslant x_j, \qquad j = 6, 7, 8.$$

We list the numerators for the problem (with 9 as denominator):

1	4	5	7	8
93	156	42	24	137
2	8	4	1	6
7	1	5	8	3
5	2	1	7	6
		*	*	
		$x_j = 0$		

(8)

where $\beta_0 = (\tfrac{1}{3}, \tfrac{2}{3}, \tfrac{1}{3})$ and the β_j are as listed.

We have $24 = \min c_j$, i.e., $c_r = 24$, with $r = 7$. We mark the 7 column in (8) and add the section

1	4	5	7	8
117	180	66	48	161
3	0	5	2	7
6	0	4	7	2
3	0	8	5	4
		*	*	
		$x_7 = 1$		

(9)

We have $\min c_j = 42$ from (8); we add the section:

1	4	8
135	198	179
6	3	1
3	6	8
6	3	7
	$x_5 = 1$	

(10)

We need not add a column headed by 7 in (10) because it would duplicate the 5 column in (9) (the use of either column would require $x_5 = 1$ and $x_7 = 1$). We do not add a 5 column in (10) since x_5 is at its upper bound in (10). We have $\min c_j = 48$ in (9). We add the section:

1	4	5	7	8
141	204	90	72	185
4	1	6	3	8
5	8	3	6	1
1	7	6	3	2
			*	
		$x_7 = 2$		

(11)

We have min $c_j = 66$ from (9). We add the section:

1	4	8
159	222	203
7	4	2
2	5	7
4	1	5
$x_7 = 1$, $x_5 = 1$		

(12)

The solution is now possible in the 7 column in (11). We have $x_7 = 3$. Substituting $x_7 = 3$ into the constraint equations, we obtain $x_6 = 0$, $x_3 = 1$, and $x_2 = 1$. Thus we have achieved the optimal solution with objective value $9 + \frac{72}{9} = 17$. Note that if x_6, x_3, or x_2 are not feasible, other solutions are possible; e.g., the 1 column of (9), the 4 column of (10), plus others if the enumeration is continued.

References

1. R. E. Gomory. On the relation between integer and non-integer solutions to linear programs. *Proc. Nat. Acad. Science* **53** (1965), 260–65.
2. J. F. Shapiro. Dynamic programming algorithms for the integer programming problem—I: the integer programming problem viewed as a knapsack type problem. *Operations Research* **16** (1968), 103–21.
3. G. B. Dantzig. "Linear Programming and Extensions." Princeton University Press, Princeton, New Jersey, 1963.
4. E. Balas. Discrete programming by the filter method. *Operations Research* **15** (1967), 915–57.