SIAM Review
Vol. 9, No. 2, April, 1967

INTEGER PROGRAMMING BY IMPLICIT ENUMERATION AND BALAS' METHOD*

ARTHUR M. GEOFFRION†

1. Introduction. In a recent comprehensive survey [2] of integer programming, M. L. Balinski expressed a belief that "... clever enumerative schemes ... are in certain situations, at least, the best known methods of solution." The adjective "clever" is undoubtedly meant to imply, of course, that one should do better than complete enumeration. This can be done by using strategies which lead, as the enumeration proceeds, to the generation of information that can be used to exclude large numbers of solutions from further consideration. The exclusion of solutions, which suggests the descriptive term *implicit enumeration* for such methods, can usually be accomplished in a variety of ways—for example, by exploiting the availability of a gradually improving bound on the optimal value of the objective function as better and better feasible solutions are found. It is unfortunate that the exclusion of solutions can sometimes give rise to very large information storage requirements in order to record which solutions have been excluded as well as enumerated, thereby adding another dimension to the need for "cleverness." A computationally successful algorithm must combine a sufficiently high "rate of exclusion" with sufficiently modest storage requirements.

Additional support for the viewpoint quoted above has accrued lately from the publication of Balas' "additive algorithm" [1] and some useful extensions proposed by Glover [5]. This and subsequent work suggests that it may be possible to achieve, for general bounded integer linear programs, both a high rate of exclusion and low storage requirements. Among the demonstrable advantages of "implicit enumeration" in general and Balas' approach in particular are the following:

(a) addition is the only arithmetic operation required (no systems of linear equations need be solved), thus eliminating roundoff problems;
(b) a feasible solution, hopefully a good one, is usually in store if the calculations are stopped prior to natural termination by exhaustion;
(c) it is easy to monitor the rate of implicit enumeration as the calculations proceed, thereby enabling informed stopping rules and opportunistic implementation;
(d) only minor modifications are necessary to handle a variety of nonlinear objective functions.

The aim of this paper is to provide, in expository fashion, the rudiments of a

* Received by the editors March 9, 1966, and in revised form October 3, 1966.

† Western Management Science Institute, University of California, Los Angeles, California. The author carried out this work as a consultant to The RAND Corporation; it was first published internally to RAND on July 19, 1965, and externally in February, 1966, as RM-4783-PR.

framework within which the reader can readily assimilate the works of Balas and his followers.

In the next section we motivate, prove, and discuss from a different point of view the flexible and economical version of the "back-track" procedure for integer programs given by Glover in [5]. Back-tracking is an old but potentially effective procedure for exhaustive search in many kinds of combinatorial problems. One of its earlier uses may well have been as a methodical way to thread a maze. It is rediscovered from time to time, and is known by different names in different fields.[1] The back-track idea provides the main foundation of Balas' algorithm, but his history-remembering scheme seems to use a good deal more storage capacity than that described by Glover. Using this scheme for implicit enumeration, in §3 we synthesize what is essentially a reformulation of Balas' algorithm.[2] A numerical example is given, and some variants are mentioned. The final section presents some new material on taking advantage of available prior information, and on advantages (c) and (d) above. It is of interest to note that a nonlinear extension of Balas' method is presented which applies to problems of target-assignment, redundancy allocation, and spare parts kit composition, among others. We conclude this introduction with a precise statement of the problem and a few preliminary definitions.

Any bounded integer linear programming problem can be written in the form:

$$\text{(P)} \qquad \text{Minimize} \quad cx \quad \text{subject to} \quad b + Ax \geqq 0, \qquad x_j = 0 \quad \text{or} \quad 1,$$

where c is an n-vector, b and 0 are m-vectors, A is an m by n matrix, and x is a binary n-vector to be chosen.[3] Any binary x will be called a *solution*. A solution that satisfies the constraints $b + Ax \geqq 0$ will be called a *feasible solution*, and a feasible solution that minimizes cx over all feasible solutions will be called an *optimal feasible solution*.

2. A back-tracking procedure for implicit enumeration.

Since there is a finite number 2^n of solutions, exhaustive enumeration provides a finite procedure for discovering an optimal feasible solution of (P). As indicated above, not all solutions are to be explicitly enumerated, of course, but rather implicitly enumerated by considering groups of solutions together. To explain how groups of solutions will be defined, we require the notion of a partial solution. A *partial solution* S is defined as an assignment of binary values to a subset of the n variables. Any

[1] One modern account in a general setting is given in [7].

[2] This reformulation is the basis for a computer routine written at The RAND Corporation that has been used to make a preliminary experimental assessment of computational efficiency [4].

[3] A bounded problem is one for which an upper bound v_j is available for each variable. The substitution

$$x_j = \sum_{i=0}^{k} 2^i y_{ji} ,$$

where k is the smallest integer such that $v_j \leqq 2^{k+1} - 1$, y_{ji} binary, permits a binary representation for x_j . When this substitution is used, it is possible to take advantage of the special structure it induces on A and c.

variable not assigned a value by S is called *free*. We adopt the notational convention that the symbol j denotes $x_j = 1$ and the symbol $-j$ denotes $x_j = 0$. Hence, if $n = 5$ and $S = \{3, 5, -2\}$, then $x_3 = 1$, $x_5 = 1$, $x_2 = 0$, and x_1 and x_4 are free. It will be seen that the order in which the elements of S are written will be used to represent the order in which the elements are generated. A *completion* of a partial solution S is defined as a solution that is determined by S together with a binary specification of the values of the free variables. In the above example, there are four possible completions of S:

$$\begin{gathered}(0, 0, 1, 0, 1),\\ (0, 0, 1, 1, 1),\\ (1, 0, 1, 0, 1),\\ (1, 0, 1, 1, 1).\end{gathered}$$

Thus, a partial solution S with s elements, say, determines a set of 2^{n-s} different completions or solutions. When there are no free variables, there is only one completion of S, the trivial one determined by S itself.

Implicit enumeration involves generating a sequence of partial solutions and simultaneously considering all completions of each. As the calculations proceed, feasible solutions are discovered from time to time, and the best one yet found is kept in store as an incumbent. Now it may happen that for a given partial solution S we can determine a *best* feasible completion of S, i.e., a feasible completion that minimizes cx among all feasible completions of S. If such a best feasible completion is better than the best known feasible solution (assuming for simplicity that one is known), then it replaces the latter in store. Or we may be able to determine that S has no feasible completion better than the incumbent. In either case, we shall say that we can *fathom* S.[4] All completions of a fathomed S have been implicitly enumerated in the sense that they can be excluded from further consideration—except, of course, a best feasible completion of S that unseats the incumbent.

Leaving aside until the next section the important question of how one fathoms a given S, we now give a flexible procedure for generating a sequence $\langle S^\nu \rangle$ of partial solutions that is nonredundant and terminates only after all 2^n solutions have been (implicitly) enumerated. By *nonredundant*, we mean that no completion of a partial solution in the sequence ever duplicates a completion of a previous partial solution that was fathomed.

Start with $S^0 = \varnothing$, where $\varnothing$ indicates the empty set. If S^0 can be fathomed, we are finished—either there is no feasible solution, or there is one and the best feasible solution can be found. If S^0 cannot be fathomed, augment it by specifying a binary value for one additional free variable at a time, each time trying to fathom the resulting partial solution, until at some trial k_1, S^{k_1} is fathomed. Now to be sure of having enough information in the future to enable us to know when all 2^n solutions have been accounted for, we store S^{k_1}; and to be sure of having a non-

[4] To cover the case where there are no free variables, we shall agree that such an S is fathomed.

redundant sequence $\langle S^{\nu} \rangle$ from $\nu = k_1 + 1$ on, it is obviously necessary and sufficient to have in all future S^{ν} at least one element complementary to one in S^{k_1}. We may simultaneously accomplish the storage of S^{k_1} and heed the condition for the nonredundancy of S^{k_1+1}, at least, by taking S^{k_1+1} to be exactly S^{k_1} with its last element multiplied by -1 and *underlined.* The underline commemorates the fathoming of S^{k_1} (an example is presented below to make these ideas more concrete).

If S^{k_1+1} can be fathomed, then it is easy to see that all completions of S^{k_1} without its last element have been enumerated, and thus we can "forget" the fathoming of S^{k_1} and of S^{k_1+1} and "remember" only the fact that S^{k_1} without its last element has been fathomed. For example, if $k_1 = 3$ and $S^3 = \{3, 5, -2\}$ was fathomed and then $S^4 = \{3, 5, \underline{2}\}$ was fathomed, then all completions of $\{3, 5\}$ have been accounted for, since the completions of $\{3, 5, -2\}$ and $\{3, 5, 2\}$ dichotomize those of $\{3, 5\}$. Thus, fathoming S^3 and S^4 is equivalent to fathoming $\{3, 5\}$. Opportunities such as this to "telescope" some history lead to the economical storage requirements of the procedure we are now motivating. The same motive that directed our choice of S^{k_1+1} directs us, in this case, to choose S^{k_1+2} as S^{k_1} less its last element with its next to last element multiplied by -1 and underlined. In our hypothetical example, S^5 would be taken to be $\{3, -\underline{5}\}$. Note that S^5 contains an element (-5) complementary to one that appears in both previously fathomed partial solutions.

If, on the other hand, S^{k_1+1} cannot be fathomed, then one would augment it by specifying a binary value for one additional free variable at a time, each time trying to fathom the resulting partial solution, until at some later trial k_2, S^{k_2} is fathomed. Note that the sequence $S^{k_1+1}, \cdots, S^{k_2}$, is nonredundant because each contains the complement of an element of S^{k_1}. When S^{k_2} is fathomed, it, in addition to S^{k_1}, must be stored; and every succeeding S^{ν} must contain not only an element that is the complement of one in S^{k_1}, but also one that is the complement of an element of S^{k_2}. Both ends may be accomplished economically by taking S^{k_2+1} as S^{k_2} with its last element complemented and underlined. In our example, if S^4 could not be fathomed and S^5 were $\{3, 5, \underline{2}, 1\}$, say, that could be fathomed $(k_2 = 5)$, then S^6 would be taken to be $\{3, 5, \underline{2}, -\underline{1}\}$.

Continuing along these lines, one is led to the procedure of Fig. 1. In this figure, the contents of Steps 1 and 2 are deliberately unspecified in order to leave maximum flexibility in the design of an algorithm by having a general convergence proof. It is important to note that the mechanism by which Step 1 attempts to fathom a partial solution can be as weak or as powerful as desired—so long as S is truly fathomed when it purports to be. However, since the most powerful fathoming mechanism amounts to solving an integer program of the same form (albeit with fewer variables) as (P) itself, one typically uses simpler mechanisms that are considerably weaker.

THEOREM. *The procedure of Fig.* 1 *leads to a nonredundant sequence of partial trial solutions which terminates only when all* 2^n *solutions have been* (*implicitly*) *enumerated.*

A proof of an essentially equivalent theorem has been given by Glover [5].

His proof proceeds by induction on the number of variables. In the Appendix we give an independent proof that is illuminating from a different point of view.

If the mechanisms of Steps 1 and 2 are computationally finite (as they are in Balas' algorithm and certainly would be in any reasonable algorithm that uses the procedure of Fig. 1), then the theorem implies that Fig. 1 is computationally finite and that an optimal feasible solution of (P) is in store at termination if the collection of feasible solutions is not empty.

Remark. It is not difficult to see from the discussion of this section and the proof given in the Appendix that:

(a) S^0 can be *any* partial solution without underlines, rather than the empty one;

(b) S can be augmented at Step 2 by a collection of elements, rather than by a single element;

(c) the elements in any consecutive sequence of nonunderlined elements in S can be permuted arbitrarily at any time;

(d) S can be augmented at Step 2 by a collection of underlined elements (rather than by a nonunderlined element) when this does not exclude any feasible completions of S that are better than the incumbent.

Observation (d) can be viewed as the result of anticipating the outcome of the procedure of Fig. 1 when S is augmented by the complement of each element of such a collection in turn. Put another way, it amounts to deliberately augmenting S during one or more consecutive passes through Step 2 in such a way that the resulting partial solution can be fathomed because it is obvious that no feasible completion better than the incumbent exists. The explicit incorporation of ob-

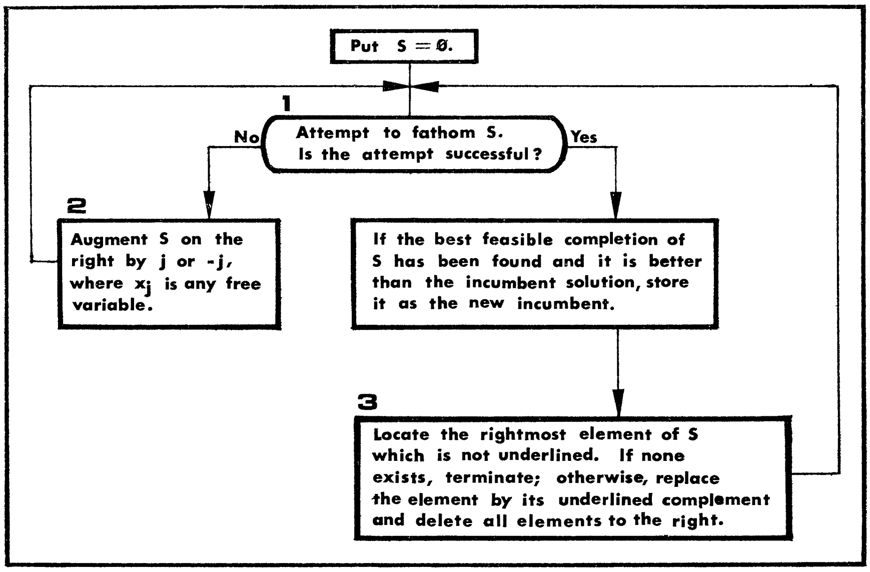
Put S = ∅.
1
Attempt to fathom S. Is the attempt successful?
No
Yes
2
Augment S on the right by j or -j, where x_j is any free variable.
If the best feasible completion of S has been found and it is better than the incumbent solution, store it as the new incumbent.
3
Locate the rightmost element of S which is not underlined. If none exists, terminate; otherwise, replace the element by its underlined complement and delete all elements to the right.

FIG. 1. *A back-tracking procedure for implicit enumeration*

servation (d) is the only substantive difference between Diagram 1 in [5, p. 884] and Fig. 1.

3. A particularization of the procedure based on Balas' algorithm. In §2 we presented and justified a flexible and economical back-tracking procedure for finding an optimal feasible solution of (P) by implicit enumeration. Details for the mechanisms of Steps 1 and 2 based on Balas' algorithm will now be derived to illustrate the implementation of the procedure.

At Step 1, the problem is to "fathom" the current partial solution S. Recall that S may be fathomed by doing either of the following:

(i) finding the best feasible completion of S,

(ii) determining that no feasible completion of S has a lower value of the objective function than the incumbent.

The general strategy will be to attempt to fathom S by taking each tack in turn by means of very simple computations.

Associated with S is a best (not necessarily unique or feasible) completion x^s of S. Constructing such a best completion is trivial—just take $x_j^s = 0$ or 1 for each free variable according as $c_j \geqq 0$ or <0. With the help of a simple change of variables we may assume without loss of generality that $c \geqq 0$, so that each free variable x_j^s may be taken to be 0. Observe that if x^s is feasible, then x^s is a best feasible completion of S and S is thereby fathomed. Since the computation of x^s is so easy, we shall test its feasibility as the first substep of Step 1 of Fig. 1. As the computations proceed, the value of the incumbent feasible solution gives a (hopefully good) upper bound $\bar{z}$ on the optimal value z^* of (P) that can be used to good advantage as indicated below. Until the first feasible solution has been found, we take $\bar{z} = \infty$, unless some better upper bound is known.

If the best completion x^s is not feasible, we do nothing further to find the best feasible completion. Instead, we attempt to determine that no feasible completion of S is better than the incumbent. If this is actually the case, then it must be impossible to complete S so as to eliminate all of the infeasibilities of x^s and yet improve upon $\bar{z}$. To demonstrate this impossibility, it is clearly sufficient to contemplate nonzero binary values only for the variables in

$$T^s \equiv \{j \text{ free: } cx^s + c_j < \bar{z} \text{ and } a_{ij} > 0 \text{ for some } i \text{ such that } y_i^s < 0\},$$

where $y^s = Ax^s + b$; for to give a value of 1 to some free variable not in T^s would either lead to a higher value than $\bar{z}$ or would not contribute to diminishing an infeasibility of x^s (we have made use of our assumption that $c \geqq 0$). Hence, if T^s is empty, then there could be no feasible completion of S that is better than the incumbent, and S is fathomed. It is also easy to see that the same conclusion holds if

$$y_i^s + \sum_{j \in T^s} \max\{0, a_{ij}\} < 0$$

for some i such that $y_i^s < 0$; for then there could be no way to select free variables so as to eliminate infeasibility. So much for Step 1.

For the augmentation mechanism of Step 2, one choice is to augment S by j_0 from T^s—where j_0 leaves the least amount of total infeasibility in the next x^s in the sense of making

$$\sum_{i=1}^{m} \min \{y_i^s + a_{ij}, 0\}$$

an algebraic maximum.

The above details for Steps 1 and 2 are summarized in Fig. 2.

3.1. An example. We have chosen to base our example on a problem taken from Balas [1] so that the interested reader may compare the present algorithm with the original one.

Let it be desired to minimize $5x_1 + 7x_2 + 10x_3 + 3x_4 + x_5$ over all binary $x_1, \cdots, x_5$ that satisfy:

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

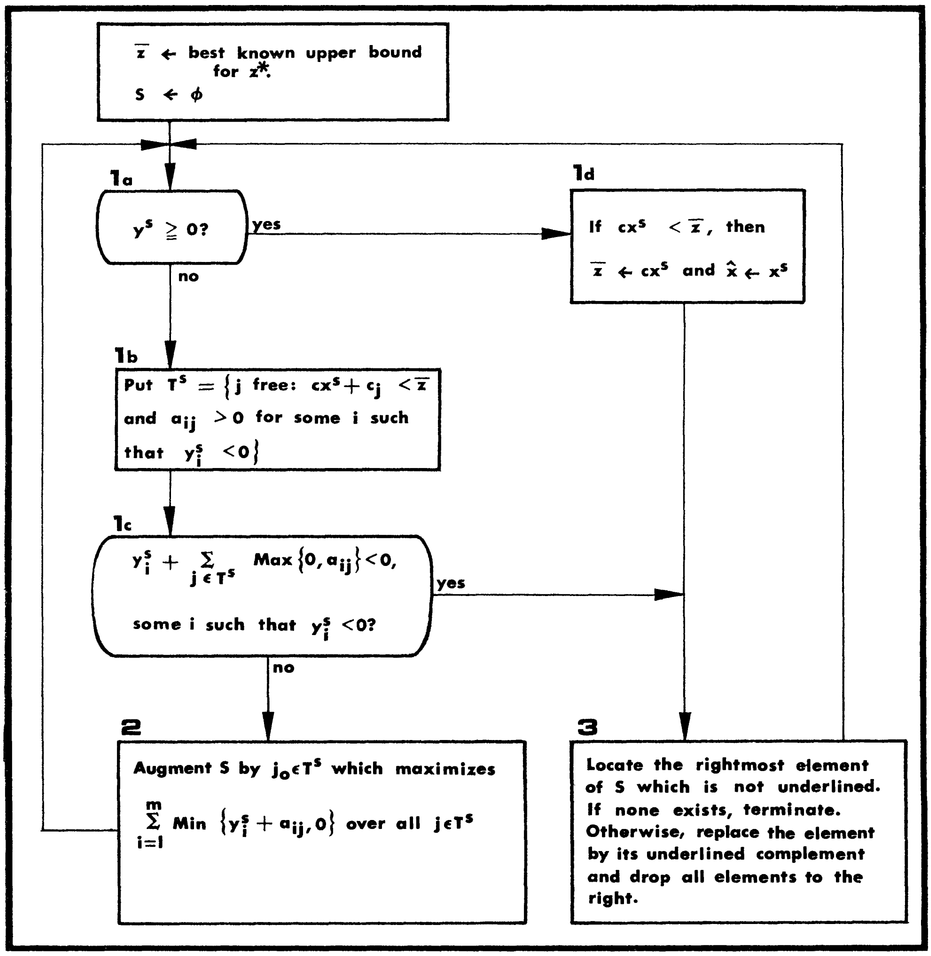
$\bar{z} \leftarrow$ best known upper bound for z^*.
$S \leftarrow \phi$
1a
$y^S \geqq 0$?
yes
no
1d
If $cx^S < \bar{z}$, then $\bar{z} \leftarrow cx^S$ and $\hat{x} \leftarrow x^S$
1b
Put $T^S = \{j$ free: $cx^S + c_j < \bar{z}$ and $a_{ij} > 0$ for some i such that $y_i^S < 0\}$
1c
$y_i^S + \sum_{j \in T^S} \text{Max}\{0, a_{ij}\} < 0$, some i such that $y_i^S < 0$?
yes
no
2
Augment S by $j_0 \in T^S$ which maximizes $\sum_{i=1}^{m} \text{Min}\{y_i^S + a_{ij}, 0\}$ over all $j \in T^S$
3
Locate the rightmost element of S which is not underlined. If none exists, terminate. Otherwise, replace the element by its underlined complement and drop all elements to the right.

FIG. 2. *A simplified version of Balas' additive algorithm*

Summarized below are the calculations that arise from applying the procedure of Fig. 2 to this problem. The subscripts on y and T are suppressed.

Step		
	$\bar{z}$	$= \infty$
	S	$= \emptyset$
1a	y	$= (-2, 0, -1) \ngeq 0$
1b	T	$= \{1, 3, 4\}$
1c	i	$= 1: -2 + 7 \geqq 0$
	i	$= 3: -1 + 2 \geqq 0$
2	j	$= 1: -1 - 2 - 1 = -4$
	j	$= 3: -3 = \boxed{-3}$
	j	$= 4: -1 - 2 - 2 = -5$
	Hence,	
	S^1	$= \{3\}$.
1a	y	$= (3, -3, 1) \ngeq 0$
1b	T	$= \{2, 5\}$
1c	i	$= 2: 5 \geqq 0$
2	j	$= 2: \boxed{0}$
	j	$= 5: -1 - 1 = -2$
	Hence,	
	S^2	$= \{3, 2\}$.
	y	$= (0, 3, 0) \geqq 0$ (S^2 fathomed)
	cx^s	$= 17 < \infty$; $\bar{z} \leftarrow 17$; $\bar{x} \leftarrow (0, 1, 1, 0, 0)$
3	S^3	$= \{3, -\underline{2}\}$
1a	y	$= (3, -3, 1) \ngeq 0$
1b	T	$= \{5\}$
1c	i	$= 2: -3 + 2 < 0$ (S^3 fathomed)
3	S^4	$= \{-\underline{3}\}$
1a	y	$= (-2, 0, -1) \ngeq 0$
1b	T	$= \{1, 4\}$
1c	i	$1: -2 + 1 + 1 = 0$
	i	$= 3: -1 < 0$ (S^4 fathomed)
3	Terminate. An optimum solution is $(0, 1, 1, 0, 0)$, and the optimal value of the objective function is 17.	

3.2. Other possible particularizations. Minor variants of the above particularization of the procedure of Fig. 1 are obtained by changing the infeasibility measure used in Step 2 (e.g., one could measure infeasibility by the most infeasible constraint or by the number of infeasible constraints); or by modifying Step 2 to give some attention to choosing an augmentation element that has a relatively small c_j as well as relatively great ability to reduce infeasibility, in an attempt to find good feasible solutions.

Other variants of Step 2 are obtained by appealing to Remark (d) of §2—by looking first for augmentations of S that lead to easily fathomed partial solutions, and only failing that for augmentations that reduce infeasibility. For example, at the Step 2 following the creation of $S^1 = \{3\}$ in the numerical illustration above, it is clear by inspection of the second constraint that when $x_3 = 1$ one must also have $x_2 = 1$ if that constraint is to be satisfied. Thus if S^1 were augmented to $\{3, -2\}$, the resulting manifest infeasibility would be discovered at the ensuing Step 1c and the next partial solution would be $\{3, \underline{2}\}$. This outcome could easily have been anticipated, once it was discovered that $x_3 = 1$ implies $x_2 = 1$ for feasibility, by augmenting S^2 by $\underline{2}$ instead of by -2. It is not difficult to formalize tests for discovering such implications. One such test involves the computation of $[y_i^s + \sum_{j \in T^s} \max \{0, a_{ij}\} - | a_{ij} |]$ for each $j \in T^s$ and i such that $y_i^s < 0$; if a bracketed expression is found negative, then x_j must be 1 or 0 according as a_{ij} is positive or negative, and S can be augmented accordingly. Balas used a slightly weaker version of this test in his additive algorithm; when it is incorporated in Fig. 2, the resulting algorithm will in general lead to virtually the same sequence of partial trial solutions as the additive algorithm itself.

See Glover [5] for extensive discussion of various ideas useful in particularizing Steps 1 and 2, including the deliberate introduction of redundant "surrogate" constraints.

4. Further remarks.

Several further remarks that enhance the usefulness and efficiency of the present approach are now presented. The content of §§4.2–4.4 appears to be new.

4.1. Finding alternative optimal solutions. The procedure of Fig. 1 finds exactly one optimal solution of (P) when the constraints are consistent. To find alternative optimal solutions, the procedure can be restarted (see below) with the following changes:

(a) put S^0 equal to a permutation of the known optimal solution;

(b) in part (ii) of the definition of fathoming (see §3), replace "a lower" by "at least as low a";

(c) in the step immediately after the positive branch of Step 1, replace "better" by "as good as" and print out each new incumbent.

In terms of Fig. 2, the initial $\bar{z}$ should be taken as z^*, and (b) is effectively accomplished by slightly modifying the definition of T^s as follows:

$$T^s = \{j \text{ free: } cx^s + c_j \leqq \bar{z} \quad \text{and} \quad a_{ij} > 0 \quad \text{for some} \quad i \quad \text{such that} \quad y_i^s < 0\}.$$

It should be noted, however, that not all alternative optima are necessarily found when some $c_j = 0$, for then any corresponding variable that appears with the value 0 in an optimal feasible solution can also be assigned the value 1 without destroying optimality if the resulting solution is still feasible.

4.2. Using prior information to make a better start. When a feasible solution is known a priori, $\bar{z}$ can be put equal to its value, and by Remark (a) of §2, S^0 can be initially taken as a permutation of this solution rather than as the empty set. If the feasible solution is a good one, as it is likely to be if it is produced by insight into the problem or by the solution to a very similar problem or by a heuristic method, then convergence should be improved.

Other times, it may seem likely a priori that certain variables take on certain values at an optimum solution. S^0 can be taken as a permutation of these values.

Since the earlier elements of S^0 will be complemented during the course of the calculations after the later elements are complemented, by the nature of the basic enumerative scheme it seems reasonable to choose the permutation that ranks the vaaribles in decreasing order of certainty as to their values. For example, it may be deemed "very likely" a priori that $x_7 = 1$ at an optimum, and "fairly likely" that $x_2 = 0$. Then S^0 should be taken as $\{7, -2\}$ rather than as $\{-2, 7\}$. The same strategy for choosing a permutation can be applied when S^0 is a known feasible solution.

4.3. Measuring the rate of implicit enumeration. The present procedure, thanks to its essentially enumerative nature and the scorekeeping method used, has the following useful property: at any iteration it is obvious by inspection of the current partial solution (including the underlines) exactly how many and which solutions have been implicitly enumerated so far (see the italicized assertion in the proof in the Appendix). For example, if $n = 24$ and the calculations are stopped when $S = \{5, 4, -\underline{2}, \underline{3}, 9\}$, then all $2^{(24-3)}$ completions of $\{5, 4, 2\}$ have been accounted for, as have all $2^{(24-4)}$ completions of $\{5, 4, -2, -3\}$. Thus, it is known precisely which $2^{21} + 2^{20}$ of the 2^{24} solutions have been enumerated, and the current incumbent is an optimal feasible solution of (P) with the additional restrictions: $[x_5 = 1, x_4 = 1]$ and either $[x_2 = 1]$ or $[x_2 = 0, x_3 = 0]$.

The main usefulness of the ability to calculate easily the number of implicitly enumerated solutions is in determining when to terminate the calculations if an optimal feasible solution has not been found in a reasonable length of computing time. It is easy to show that the fraction of all 2^n solutions implicitly enumerated up to the current iteration is $\sum_i (\frac{1}{2})^{p_i}$, where the sum is taken over all underlined elements of the current partial solution and p_i is the position of the ith underlined element counting from the left. In the above example, $(\frac{1}{2})^3 + (\frac{1}{2})^4 = \frac{3}{16}$ of the 2^{24} solutions have been implicitly enumerated. Thus one can keep a running tally of the progress of the calculations, and by extrapolation obtain an estimate of the additional time required for termination. Fig. 3 illustrates typical behavior observed in many of the problems run by Freeman [4]. This device also provides a potent tool for adaptive implementation of the procedure with a repertoire of possible methods for attempting to fathom or augment partial solutions. By sampling the rate of implicit enumeration over time intervals with the different methods, estimates of their current relative efficiencies can be obtained and used to establish (perhaps time-varying) priorities for their use.

4.4 Nonlinear objective functions. The underlying enumerative scheme of Fig. 1 is not in any way dependent on the linear nature of (P). In the linear case, however, reasonably efficient details for Steps 1 and 2 appear to be available. It is natural to seek extensions to integer nonlinear problems. To illustrate how this can be done we shall indicate how Fig. 2 can be modified to handle a wide class of nonlinear objective functions.

Let it be desired to minimize a nonlinear function $f(x)$ subject to $b + Ax \geqq 0$ and x binary. Assume that for any given partial solution S it is possible to find easily a best completion x^s of S. That is, we must be able to minimize $f(x)$ over

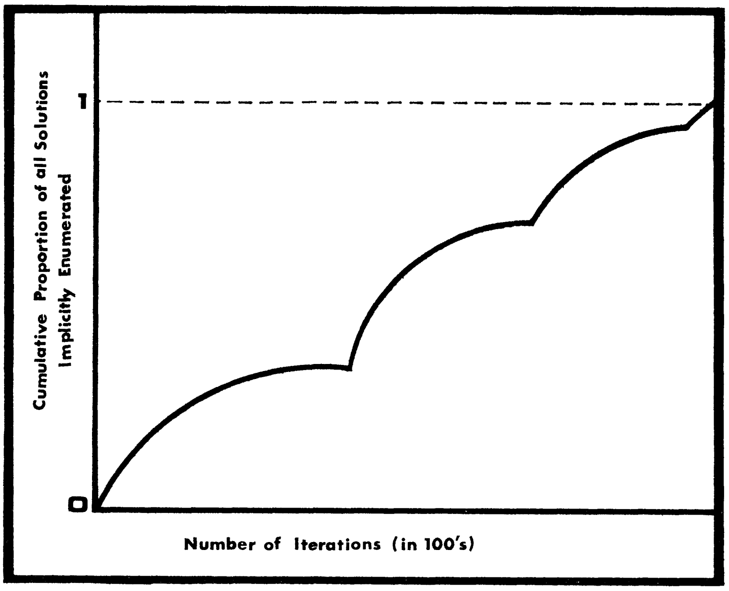

FIG. 3

the free binary variables while the variables in S are held at their assigned values. It can be shown that the procedure of Fig. 2 remains valid if we make the following modifications: (a) x^s defined as above; (b) T^s defined as $\{j \text{ free}: f(x^{s/j}) < \bar{z}$ and $\tilde{a}_{ij} > 0$ for some i such that $y_i^s < 0\}$, where $x^{s/j}$ is just x^s with the jth free variable complemented, and $\tilde{a}_{ij}$ is defined to be a_{ij} if $x_j^s = 0$ and $-a_{ij}$ if $x_j^s = 1$; (c) $\tilde{a}_{ij}$ replaces a_{ij} in Steps 1c and 2; and (d) $f(x^s)$ replaces cx^s in Step 1d. Similar modifications permit the ideas in §3.2 to be used.

Typical examples of nonlinear objective functions to which this extension applies are: (a) $\sum_k v_k \prod_j (p_{jk})^{x_j}$, where $v_k \geqq 0$ and $0 \leqq p_{jk} \leqq 1$, and (b) $-\prod_j (1 - p_j^{x_j})$, where $0 \leqq p_j \leqq 1$. Binary representation of bounded integers causes no difficulty. Objective functions of variety (a) arise in target-assignment problems (see, e.g., [3]), while those of variety (b) occur in optimal redundancy allocation problems (see, e.g., [6]). Optimum spare parts kit problems and others also fall within the domain of applicability. The present nonlinear extension may be a useful alternative for moderate-sized problems to the approximate methods developed in [3] and [6], and to methods of dynamic programming or "marginal analysis" genre often proposed for solving nonlinear integer programs, especially when there are more than a few constraints.

Appendix. The proof below involves induction on the sequence $\langle S^\nu \rangle$ of partial solutions. To clarify the consideration of various cases in the proof, we refer to Fig. 1A, an expanded but equivalent version of Fig. 1.

Proof. If $S^0 = \emptyset$ can be fathomed, the theorem is obviously true. Hence, we may assume that $\emptyset$ cannot be fathomed.

To show that $\langle S^\nu \rangle$ is nonredundant, we shall show that, if $S^1, \cdots, S^\nu$ are non-

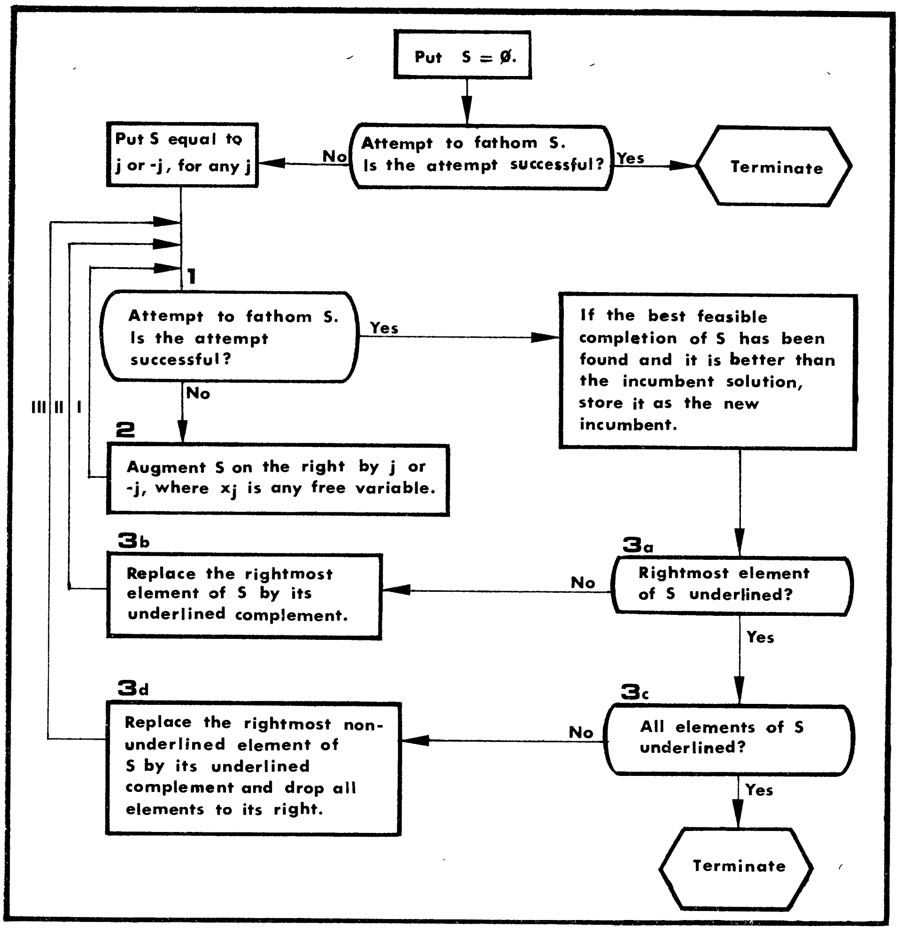
Put S = ∅.
Attempt to fathom S. Is the attempt successful?
No
Yes
Put S equal to j or -j, for any j
Terminate
1
Attempt to fathom S. Is the attempt successful?
Yes
If the best feasible completion of S has been found and it is better than the incumbent solution, store it as the new incumbent.
No
III II I
2
Augment S on the right by j or -j, where xj is any free variable.
3b
Replace the rightmost element of S by its underlined complement.
3a
No
Rightmost element of S underlined?
Yes
3d
Replace the rightmost non-underlined element of S by its underlined complement and drop all elements to its right.
3c
No
All elements of S underlined?
Yes
Terminate

FIG. 1A. *An expanded version of Fig.* 1

redundant, then $S^{\nu+1}$ cannot be redundant; i.e., that $S^{\nu+1}$ must include the complement of at least one element from each of the partial solutions fathomed prior to $S^{\nu+1}$. There are three pathways by which $S^{\nu+1}$ can be determined from S^{ν}; they are labeled I, II, and III in Fig. 1A. If pathway I is taken, the desired conclusion follows from $S^{\nu} \subset S^{\nu+1}$. If pathway II is taken, the desired conclusion follows from the evident fact that S^{ν} must have been determined from $S^{\nu-1}$ by pathway I and hence $S^{\nu-1} \subset S^{\nu+1}$. To establish the desired conclusion for pathway III, we observe from Fig. 1A that (i) the element complemented in Step 3d was contained in every partial solution since it was originally introduced (and hence in every fathomed partial solution since that time), and that (ii) $S^{\nu+1}$ less its last element is not redundant with respect to the partial solutions (if any) fathomed up to the time that the deleted element was introduced simply because $S^{\nu+1}$ less its last element coincides with the actual partial solution at that time.

It remains to show that $\langle S^{\nu}\rangle$ terminates only when all 2^n solutions have been (implicitly) enumerated. Clearly $\langle S'' \rangle$ terminates only if a partial solution consisting of all underlined elements is fathomed. From our earlier remarks concerning the "telescoping of history," we see that the proof would be at hand if we could show that *every partial solution has the following property: each underlined element implies that all completions of that portion up to and including the complement of the underlined element have been enumerated.* Now underlined elements have two possible origins: Steps 3b and 3d. Any underlined element created at Step 3b obviously has the asserted property. To see that the same is true of underlined elements created at Step 3d, consider the first time Step 3d is encountered. Then all underlined elements of the corresponding partial solution S must have been created at Step 3b; since S was just fathomed, therefore, by "telescoping" it follows that all completions of S less its right-most consecutive underlined elements have been enumerated, i.e., this deleted partial solution has been fathomed. Thus, the new partial solution generated at the first execution of Step 3d has the desired property. A similar argument holds for each subsequent execution of 3d. The proof is now complete.

REFERENCES

[1] E. Balas, *An additive algorithm for solving linear programs with zero-one variables*, Operations Res., 13 (1965), pp. 517–546.

[2] M. L. Balinski, *Integer programming: methods, uses, computation*, Management Sci., 12 (1965), pp. 253–313.

[3] S. I. Firstman, *An approximating algorithm for an optimum aim-points problem*, Naval Res. Logist. Quart., 7 (1960), pp. 151–167.

[4] R. J. Freeman, *Computational experience with the Balas integer programming algorithm*, P-3241, The RAND Corporation, Santa Monica, California, 1965.

[5] F. Glover, *A multiphase-dual algorithm for the zero-one integer programming problem*, Operations Res., 13 (1965), pp. 879–919.

[6] F. Proshan and T. A. Bray, *Optimal redundancy under multiple constraints*, Ibid., 13 (1965), pp. 800–814.

[7] R. J. Walker, *An enumerative technique for a class of combinatorial problems*, Combinatorial Analysis, Proceedings of Symposia in Applied Mathematics, vol. 10, R. Bellman and M. Hall, Jr., eds., American Mathematical Society, Providence, Rhode Island, 1960, pp. 91–94.