IMPROVED BOUNDS FOR A CLASS OF 0-1 INTEGER PROGRAMS USING COMBINED SURROGATE-LAGRANGEAN TECHNIQUES

Fred Glover, et al

Texas University at Austin

Prepared for:

Office of Naval Research

June 1975

CENTER FOR CYBERNETIC STUDIES

The University of Texas
Austin, Texas 78712

Research Report CCS 241

IMPROVED BOUNDS FOR A CLASS OF 0-1 INTEGER PROGRAMS USING COMBINED SURROGATE-LAGRANGEAN TECHNIQUES

by

Fred Glover*
G. Terry Ross

June, 1975

*University of Colorado

This research was partly supported by Project No. NR 047-021, ONR Contracts N00014-75-C-0616 and N00014-75-C-0569 with the Center for Cybernetic Studies, The University of Texas. Reproduction in whole or in part is permitted for any purpose of the United States Government.

CENTER FOR CYBERNETIC STUDIES

A. Charnes, Director
Business-Economics Building, 512
The University of Texas
Austin, Texas 78712
(512) 471-1821

ABSTRACT

A new method is described for calculating bounds for a special class of 0-1 integer programming problems. This method includes features of both Lagrangean and surrogate relaxation approaches. Certain binary knapsack calculations carried out in appropriate combinations together with an additivity principle provide the basis for computing the bounds.

Solving a linear programming relaxation of an integer programming problem is a commonly used method for computing bounds in branch and bound algorithms. When an integer program includes a subset of constraints with an exploitable mathematical structure, better bounds may be obtained by solving Lagrangean relaxations [3,6] or surrogate relaxations [7,8,9]. The constraints lacking special structure are assigned multipliers and carried into the objective function in the Lagrangean approach or used to form a summarizing constraint in the surrogate approach. These relaxations may improve the efficiency of a branch and bound algorithm when specialized algorithms are used to solve the relaxations and when the stronger bounds enhance the fathoming tests. The value of these relaxations over other relaxations depends in part on the values selected for the multipliers, and the determination of optimal multipliers give rise to "duality theories" of mathematical programming [5,9]. Although surrogate relaxations are stronger than Lagrangean relaxations (i.e. yield smaller duality gaps)[1] they may also be more difficult to solve. In general, the issue of strength versus ease of solution can be crucial to the design of an effective algorithm.

The purpose of this paper is to describe a new method for calculating bounds for a special class of 0-1 integer programming problems. This method has features of both the Lagrangean and surrograte approaches. Certain binary knapsack calculations carried out in appropriate combinations together with an additivity principle provide the basis for computing the bound. Our development is strongly motivated by the "augmented"

bounding construct proposed by Ross and Soland in their algorithm for the generalized assignment problem [10]. In that algorithm, bounds are computed by calculating an initial bound from a relaxation and augmenting this with penalties computed from the solution to binary knapsack problems. These penalties may be developed in the framework of Lagrangean relaxation, but the extension proposed here, which also makes use of surrogate analyses, gives stronger penalties for the more general class of problems we consider. At the same time, these new penalties can be computed in a highly efficient manner and may be used to augment the bound through appropriate binary knapsack calculations.

Our development is motivated not only by the success of Ross and Soland's use of binary knapsacks for accelerating the solution of generalized assignment problems, but also by the recent advance in the efficiency of binary knapsack procedures described in [2]. Consequently, the method proposed in this paper is designed to take advantage of techniques with demonstrated computational attractiveness, as well as to solve problems of substantially greater scope than those previously shown to be amenable to the "augmented" bounding construct.

Unclassified

Security Classification

DOCUMENT CONTROL DA: R & D

Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified

ORIGINATING ACTIVITY (Corporate author)
Center for Cybernetic Studies
The University of Texas

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE
Improved Bounds for a Class of 0-1 Integer Programs Using Combined Surrogate-Lagrangean Techniques

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

5. AUTHOR(S) (First name, middle initial, last name)
Fred Glover
G. Terry Ross

6. REPORT DATE	7a. TOTAL NO OF PAGES	7b. NO OF REFS
June 1975	24	10

8a. CONTRACT OR GRANT NO	9a. ORIGINATOR'S REPORT NUMBER(S)
N00014-75-C-0569, N00014-75-C-0616	Center for Cybernetic Studies Research Report CCS 241
b. PROJECT NO NR 047-021	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
c.	
d.	

10. DISTRIBUTION STATEMENT
This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Office of Naval Research (Code 434) Washington, D.C.

13. ABSTRACT

A new method is described for calculating bounds for a special class of 0-1 integer programming problems. This method includes features of both Lagrangean and surrogate relaxation approaches. Certain binary knapsack calculations carried out in appropriate combinations together with an additivity principle provide the basis for computing the bounds.

DD FORM 1 NOV 68 1473 (PAGE 1)

S/N 0101-807-6811

Unclassified
Security Classification

A-3140B

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Branch and Bound						
0-1 Integer Programming						
Lagrangean Relaxation						
Surrogate Relaxation						
Binary Knapsack Problems						

1. Notation and Problem Definition

The class of problems discussed in this paper has the following mathematical structure:

$$\text{minimize: } Z = \sum_{j \in J} d_j x_j \tag{2.1}$$

subject to

$$\sum_{j \in J} a_{ij} x_j \geq A_i \quad \text{for all } i \in I \tag{2.2}$$

$$\sum_{j \in R_k} b_{kj} x_j \geq B_k \quad \text{for all } k \in K \tag{2.3}$$

$$\sum_{j \in S_h} c_{hj} x_j \geq C_h \quad \text{for all } h \in H \tag{2.4}$$

$$0 \leq x_j \leq 1 \quad \text{for all } j \in J \tag{2.5}$$

$$x_j \text{ integer for all } j \in J \tag{2.6}$$

where $R_{k_1} \cap R_{k_2} = \emptyset$ for $k_1, k_2 \in K$ and $k_1 \neq k_2$

$S_{h_2} \cap S_{h_2} = \emptyset$ for $h_1, h_2 \in H$ and $h_1 \neq h_2$

In addition, the coefficients of (2.3) and (2.4) must satisfy a sign assumption:

For any $k \in K$, $h \in H$ such that $R_k \cap S_h \neq \emptyset$,

either $b_{kj} c_{hj} > 0$ for all $j \in R_k \cap S_h$

or $b_{kj} c_{hj} < 0$ for all $j \in R_k \cap S_h$.

In general, the sign assumption is satisfied as long as the coefficients b_{kj} associated with any set R_k are all the same sign (either all positive or all negative with zero coefficients excluded by restricting the membership of the sets R_k) and the same is true for the coefficients c_{hj} in any S_h. As a practical matter, such conditions are common in resource allocation problems. Other more complex sign conditions are also compatible with the sign assumption.

For notational convenience, it is assumed that $\bigcup_{h \in H} S_h \subset \bigcup_{k \in K} R_k$. If this assumption does not hold, a redundant constraint of the form $\sum_{J \in M} \delta_j x_j \geq \delta_o$ may be added. In this constraint, each δ_j is chosen to be either +1 or -1 to satisfy the sign assumption, δ_o equals the sum of the negative δ_j and $M \equiv \{j \mid j \in S_h$ for some $h \in H$ and $j \notin R_k$ for all $k \in K\}$.

The model (2.1)-(2.6) encompasses a number of important special cases of the distribution-assignment type. In particular, the generalized assignment problem arises when $I = \emptyset$, all $b_{kj} = 1$, all $B_k = 1$, each constraint (2.3) is an equality, all c_{hj} and C_h are negative and there is exactly one index j in each intersection $R_k \cap S_h$. More importantly, generalized transportation and assignment problems with additional side conditions, including certain multi-commodity distribution problems, may be represented in this format.

2. Calculating a Lower Bound

To provide insight into our method for computing a lower bound for (2.1) - (2.6), an interpretation of the bounding procedure proposed by Ross and Soland [10] in terms of Lagrangean relaxation is presented first.

Corresponding to the general integer programming problem:

$$
\begin{aligned}
(P) \qquad & \text{minimize } cx \\
& \text{subject to } Ax \geq b \\
& \qquad\quad Bx \geq d \\
& \qquad\quad x \geq 0 \\
& \qquad\quad x \text{ integer}
\end{aligned}
$$

the Lagrangean Relaxation of (P) can be defined [3,6] relative to $Ax \geq b$ and a non negative vector λ to be

$$
\begin{aligned}
(PR_\lambda) \qquad & \text{minimize } cx + \lambda (b-Ax) \\
& \text{subject to } Bx \geq d \\
& \qquad\quad x \geq 0 \\
& \qquad\quad x \text{ integer}
\end{aligned}
$$

An equivalent form of the objective function of (PR_λ) is minimize $(b + (c - \lambda A)x$. This alternate form has a special interpretation when λ is chosen as a vector of optimal dual multipliers associated with the constraints $Ax \geq b$ in the relaxation

$$
\begin{aligned}
(PR) \qquad & \text{minimize } cx \\
& \text{subject to } Ax \geq b \\
& \qquad\quad x \geq 0.
\end{aligned}
$$

Clearly the optimal solution to (PR) is a (possibly weak) lower bound on the optimal value for (P), and its value is given by λb (by duality). The relative cost factors of (PR) are given by the vector $(c - \lambda A)$. In general, one would expect that some of the constraints $B x \geq d$ would not be satisfied by the optimal solution to (PR). Thus, in this case, the Lagrangean relaxation may be interpreted as a problem of identifying the least costly modifications in the linear programming solution necessary to solve (P). That is, each relative cost factor $c_j - \lambda A_j$ may be interpreted as a penalty that accompanies changes in the value of x_j. The optimal value of (PR_λ) consists of a linear programming bound together with an accumulated penalty reflecting necessary changes in the linear programming solution. Using such an approach to determine a Lagrangean bound has particular appeal, whenever (PR) has a special mathematical structure.

The method just described for computing a bound features three important steps that must be performed. First, a linear relaxation that ignores some constraints must be solved. Second, using post optimality information, penalties associated with making changes in individual variable values must be determined. Finally, an aggregate penalty reflecting the composite changes necessary to satisfy the previously ignored constraints must be computed to add to the linear relaxation bound. When applied to the problem (2.1)-(2.6) these three steps can provide a strong bound provided the structure of the problem is exploited fully.

The initial step of our procedure is to solve the linear program given by (2.1) (2.2) (2.3) and (2.5). Given the optimal dual variables, w_i^*, associated with the constraints (2.2), a partial Lagrangean relaxation can be formed with the constraints (2.2) in the objective function. This relaxation has the form:

$$\text{minimize} \quad \sum_{j \in J} d_j w_j + d_o \tag{3.1}$$

$$\text{subject to} \quad \sum_{j \in R_k} b_{kj} x_j \geq B_k \text{ for all } k \in K \tag{2.3}$$

$$0 \leq x_j \leq 1 \text{ for all } j \in J \tag{2.5}$$

$$\text{where} \quad d_j = d_j - \sum_{i \in I} w_i^* a_{ij}$$

$$d_o = \sum_{i \in I} w_i^* A_i$$

This new problem consists simply of a collection of separable continuous knapsack problems each of the form

$$\text{minimize} \quad \sum_{j \in R_k} d_j x_j \tag{3.2}$$

$$\text{subject to} \quad \sum_{j \in R_k} b_{kj} x_j \geq B_k \tag{3.3}$$

$$0 \leq x_j \leq 1 \text{ for all } j \in R_k \tag{3.4}$$

The problems (3.2) (3.3) (3.4) are easily solved by standard techniques. It may be assumed that all $b_{kj} > 0$ (substituting the variable $x_j = 1 - x_j$ if $b_{kj} < 0$ to achieve positivity). The variables are then ordered in terms of ascending values of the ratio d_j/b_{kj}, and the x_j are assigned values by reference to their position in the ordering in the customary manner. If some $d_j < 0$, the associated variables x_j is set equal to 1 and remaining x_j

assigned values until (3.3) is satisfied. From duality theory, the optimal values for the variables in the knapsack problems (3.2) (3.3) (3.4), denoted x_j^*, substituted in either (3.1) or (2.1) give the same value. That is, if Z^* denotes the optimal value of the objective function for the problem (2.1) (2.2) (2.3) (2.5) then

$$Z^* = \sum_{j \in J} d_j x_j^* = \sum_{j \in J} d_j' x_j^* + d_o .$$

The solution x_j^* gives, therefore, a valid lower bound Z^* on the optimal value for Z in the integer program (2.1)-(2.6). Except for trivial problems, the values x_j^* will not satisfy some of the constraints (2.4) and (2.6), and as indicated earlier, this fact may be used to revise the bound.

In order to obtain a solution that satisfies (2.4) and (2.6), some changes must be made in the values of x_j^*. Making these changes will in turn increase the value of Z^* to better reflect the value of Z in the problem (2.1)-(2.6). The next steps in computing the revised bound are, therefore, to identify changes that decrease infeasibility and to ascertain the penalty associated with each such change.

For notational convenience, let $H' \equiv \{h \in H \mid \sum_{j \in S_h} c_{hj} x_j < C_h\}$ and for each $h \in H'$ let $S_h^+ \equiv \{j \in S_h \mid c_{hj} < 0$ and $x_j^* > 1\}$ and let $S_h^- \equiv \{j \in S_h \mid c_{hj} < 0$ and $x_j^* > 0\}$. The set S_h^+ (S_h^-) comprises those variables that decrease infeasibility in the constraints (2.6) when they themselves are increased to 1 (decreased to 0). More importantly, the collection of modified solutions that would satisfy any single constraint $h \in H'$ is given by the set of feasible solutions to the constraints:

$$\sum_{j \in S_h^+} c_{hj}' x_j + \sum_{j \in S_h^-} c_{hj}' (1-x_j) \geq C_h' \qquad (3.5)$$

$$x_j = 0 \text{ or } 1 \qquad (3.6)$$

where $C'_h = C_h - \sum_{j \in S_h} c_{hj} x^*_j$

and

$$c'_{hj} = \begin{cases} (1-x^*_j)\ c_{hj} & \text{for } j \in S^+_h \\ -x^*_j\ c_{hj} & \text{for } j \in S^-_h \end{cases} .$$

For the purpose of determining a lower bound on the optimal solution to (2.1)-(2.6), we are particularly interested in finding a solution to (3.5) and (3.6) that optimizes the objective function

$$\text{minimize } z'_h = \sum_{j \in S^+_h} q_j x_j + \sum_{j \in S^-_h} p_j (1-x_j) \qquad (3.7)$$

That is, we seek to identify the least costly modified solution to (3.2) (3.3) (3.4) which satisfies (2.4) and (2.6), where q_j denotes in increase in z^* associated with enforcing $x_j = 1$ and p_j denotes the increase in z^* associated with enforcing $x_j = 0$. The penalties p_j and q_j can be computed easily using a naive post-optimality calculation on (3.2) (3.3) and (3.4) that will subsequently be illustrated by example.

The result that establishes the value of solving binary knapsack problems of the form (3.7) subject to (3.5) and (3.6) is the following:

Theorem - If z^*_h denote the optimal value of (3.7) subject to (3.5) and (3.6), then $z^* + \sum_{h \in H'} z^*_h$

is a valid lower bound on the optimum objective function value to the problem (2.1)-(2.6).

The theorem will establish that the penalties p_j and q_j are additive under the sign assumption. Thus an aggregate penalty can be determined to reflect the minimal cost of making changes necessary to satisfy (2.4) and (2.6) which were ignored in finding the optimal solution to the relaxation (2.1), (2.2), (2.3) and (2.5).

Before proving the theorem, an example will be given of the method for determining p_j and q_j that will clarify their nature and illustrate conditions that need to be established in order to demonstrate the validity of the theorem. Consider the continuous knapsack problem

$$\text{minimize} \qquad -9x_1 + 30x_2 + 20x_3 + 50x_4 + 45x_5$$

$$\text{subject to} \qquad 20x_1 + 30x_2 + 10x_3 + 20x_4 + 15x_5 \geq 55$$

$$0 \leq x_j \leq 1, \quad j=1,2,\ldots,5$$

which has the form of the knapsack problem (3.2)-(3.4). All $b_{kj} > 0$, and the variables are arranged by ascending value of the ratio of the objective function coefficient to the constraint coefficient. The optimal solution is $x_1^* = x_2^* = 1$, $x_3^* = 1/2$, $x_4^* = x_5^* = 0$. By the definitions of S_h^+ and S_h^-, p_j values are needed for at most the first three variables, and q_j values are needed for at most the last three variables. Each penalty is simply the change that occurs in the optimal value of the objective function when the variable is forced to 0 or to 1 as required. The following calculations illustrate the post optimality determination of these penalties.

$$p_1 = -(-9) + 1/2(20) + 3/4(50) = 56\ 1/2$$
$$0 = -20 + 1/2(10) + 3/4(20)$$

$$p_2 = -30 + 1/2(20) + 1(50) + 1/3(45) = 45$$
$$0 = -30 + 1/2(10) + 1(20) + 1/3(15)$$

$$p_3 = -1/2(20) + 1/4(50) = 2\ 1/2$$
$$0 = -1/2(10) + 1/4(20)$$

$$q_3 = 1/2(20) - 1/6(30) = 5$$
$$0 = 1/2(10) - 1/6(30)$$

$$q_4 = 50 - 1/2(20) - 1/2(30) = 25$$
$$0 = 20 - 1/2(10) - 1/2(30)$$

$$q_5 = 45 - 1/2(20) - 1/3(30) = 25$$
$$0 = 15 - 1/2(10) - 1/3(30)$$

It should be noted that the foregoing penalties for setting variables to 0 and 1 are not independently additive. That is, the effect of summing several such penalties may not give a valid penalty for the corresponding assignment of values to the x_j variables. To illustrate, the penalty for setting $x_2 = 0$ and $x_4 = 1$ is 45 whereas the sum of p_2 and q_4 is 70. Nevertheless, it is desirable to use the foregoing "knapsack panalties" in some fashion because they are stronger than the corresponding Lagrangean penalties (which are respectively 49, 30, 0, 0, 10 and 15 for p_1, p_2, p_3, q_3, q_4 and q_5).[2] The following observation lays a foundation for using the knapsack penalties.

Lemma: The p_j and q_j penalties for a knapsack problem with all $b_{kj} > 0$ are additive as long as they are not mixed (i.e., as long as a given sum contains only one of these two types but not the other).

Proof: The lemma follows immediately by considering the form of the calculation that gives a valid penalty for simultaneously setting more than one x_j to 0 or setting more than x_j to 1. A sequential assignment of such values yields a piecewise linear concave function.

This lemma is significant because when addition is permissible it increases the differential between the knapsack penalties and the Langranean penalties. (In fact, it is easy to see that any given p_j or q_j will always dominate the corresponding Lagrangean penalty since the latter involves a single ratio calculation whereas the former can involve several ratios each progressively more costly.) The more terms that appear in a legitimate sum, the greater the difference between the two types of penalties. This is not meant to minimize the value of the Lagrangean penalties in the context of the theorem which are also valid in this application.

The difficulty in using the foregoing Lemma to take advantage of the knapsack penalties in a more general context relates to the signs of coefficients b_{kj}. In particular, if some b_{kj} in (3.3) are negative, necessitating a transformation to achieve positivity, then a p_j for a transformed variable will not necessarily be additive with a p_j associated with a variable which was not transformed. That is, for a variable with $b_{kj} < 0$, the p_j will have been computed as q_j and vice versa. However, under the sign assumption such potential conflicts are resolved as we now show.

Proof of the Theorem:

Penalties associated with any two variables $j_1, j_2 \in S_h$ such that $j_1 \in R_{k_1}$, $j_2 \in R_{k_2}$ and $k_1 \neq k_2$ are automaticly additive because of the separability of the problems (3.2)-(3.4). Thus, in order to invoke the Lemma, we must show that for all $j \in S_h \cap R_k$, the penalties p_j and q_j in (3.5) are of the same type in the sense of the Lemma.

The condition $b_{kj}c_{hj} > 0$ for all $j \in S_h \cap R_k$ assures that for $j \in S_h^+$, $b_{kj} > 0$ and for $j \in S_h^-$, $b_{kj} < 0$. Thus a penalty p_j incurred when x_j^* is set equal to 0 given that $b_{kj} < 0$ is of the q_j type for a knapsack problem with all $b_{kj} > 0$. Similarly, the condition $b_{kj}c_{hj} < 0$ for all $j \in S_h \cap R_k$ assures that for $j \in S_h^+$, $b_{kj} < 0$ and for $j \in S_h^-$, $b_{kj} > 0$. Thus a penalty q_j incurred when x_j^* is set equal to 1 given that $b_{kj} < 0$ is of the p_j type for a knapsack problem with all $b_{kj} > 0$. Thus, the Lemma applies to (3.7) and the penalties p_j and q_j of (3.7) are acceptably additive.

Next we show that for any set of additive (non-negative) penalties, the sum $\sum_{h \in H'} z_h^*$ is a valid increment to z^* and thus yields a bound on the optimal integer solution to (2.1)-(2.6). Since for any $h \in H'$ the constraint (2.4) is not satisfied by x_j^* it follows that some subset of the x_j^* for $j \in S_h^+$ must be set equal to 1 and/or some x_j^* for $j \in S_h^-$ must be set to 0. Moreover, since the penalties for changing the x_j^* values are non-negative, the total cost of such changes cannot be less than the optimal value of (3.7) which is the least cost assignment that permits (2.3) to be satisfied, leaving other x_j^* unchanged. Thus each z_h^* is a valid increment to z^*, and the fact that the sets S_h are disjoint assures that the sum $\sum_{h \in H'} z_h^*$ is a valid increment for z^*. This completes the proof.

3, Considerations in Implementation

The second half of the proof of the theorem shows that other additive penalties may be used in place of the knapsack penalties. Furthermore, it may be observed that for penalties p_j and q_j that have a piecewise concave interaction, the knapsack problem (3.5)-(3.7) can be expanded to include 0-1 variables and constraints reflecting these interactions. For example, if p_{uv} is a lower bound on the penalty for setting both x_u and x_v equal to 0, then we may introduce a 0-1 variable x_{uv} with a coefficient $p_{uv} - (p_u + p_u)$ in (3.7) and a constraint $X_{uv} \geq x_u + x_v - 1$. Such techniques for handling interactions are easily generalizable, but they may not be extremely useful since they destroy the convenient knapsack structure.

It may be noted that problems containing imbedded networks and related special structures generally contain a set of constraints (2.3) with all b_{kj} equal to +1 or -1. The knapsack penalty calculations simplify substantially for such constraints, and in addition, the solutions x_j^* are integer valued. The findings of [4] suggest that there may be some virtue in this latter condition. One approach to take greater advantage of the continuous knapsack problems (3.2)-(3.4), when some x_j^* values are fractional is by employing several iterations of a cutting plane approach, continuing as long as worthwhile progress is made. Upon termination, the constraint (3.3) may be replaced by a surrogate constraint[3] generated from the dual multipliers of the original (3.3) and the currently binding cutting planes to give a new constraint (3.3'). If the cutting plane approach succeeds in obtaining an integer solution, the resulting continuous knapsack will admit this same solution as optimal thereby giving an improved value for Z^*. The validity of replacing (3.3) by (3.3') is assured by the following observation.

Remark: If the constraint (3.3) satisfies the sign assumption, the constraint (3.3') will also satisfy this assumption upon dropping any j from R_k for which b'_{kj} becomes 0.

Proof: Suppose all $b_{kj} > 0$ in (3.3). If the remark were false, some b'_{kj} would be negative. This implies that whereas $x_j = 1$ would assist the satisfaction of (3.3) it would hinder the satisfaction of (3.3'). No standard cutting plane produces such a contrary coefficient condition which could only be valid if superfluous.

It may also be observed that every index j in (3.3) will appear in (3.3') if (3.3) has a non-zero weight in forming the surrogate constraint (3.3'). There is, however, no disadvantage in dropping some j from R_k. Our remarks on this point also have a bearing on the general case in which $\bigcup_{h \in H} S_h \not\subset \bigcup_{k \in K} R_k$. Wherever a coefficient b_{kj} (or b'_{kj}) is equal to 0 (and j may be removed from R_k), the calculation of knapsack penalties for x_j independent of the problem (3.2)-(3.4). For any such j, there are two relevant cases:

$$p_j = -d'_j \text{ and } x^*_j = 1 \text{ if } d_j < 0.$$

$$q_j = d_j \text{ and } x^*_j = 0 \text{ if } d_j > 0.$$

(If $d'_j = 0$ x^*_j may be set equal to 0 or 1 and j may be excluded from the sets S^+_h and S^-_h.) Thus, when the knapsack calculation is irrelevant for some variables, the calculation of the p_j and q_j penalties corresponds to standard Lagrangean relexation methods. Further, when the optimal dual multiplier, w^*_k, associated with a constraint (2.3) is equal to 0, indicating the constraint is non-binding in the linear program (2.1) (2.2) (2.3) (2.5), the knapsack calculation may be irrelevant. In this case, the foregoing rule would be more efficient than setting up the knapsack problem.

The dual multiplier values of the linear program (2.1) (2.2) (2.3) and (2.5) make it clear how to utilize constraints (2.3) that are in equality form. If either of the two inequalities that compose the equation has implicitly a non-zero dual multiplier, then that inequality can be used as (3.3).

4. Extensions to More General Problems

The proof of the theorem in section 3 does not rest on the requirements that the constraints (2.4) and hence (3.6) are single inequality knapsack constraints. The constraints (2.4) may in fact represent a collection of disjoint decomposable matrix inequalities. In this case, the sign assumption must hold in the same inequality sense for all components of c_{hj} (with a relaxation of the strict positivity or strict non-negativity). Thus, the sign assumption becomes for all components c^i_{hj} of c_{hj}:

$$\text{either } b_{kh}c^i_{hj} \geq 0 \text{ for all } j \in R_k \cap S_h$$

$$\text{or } b_{kj}c^i_{hj} \leq 0 \text{ for all } j \in R_k \cap S_h.$$

The necessary changes in other definitions and statements are analagous.

Systems of decomposable non negative or non-positive matrix inequalities (or equations) may satisfy the extended sign assumption when (2.3) has corresponding non-negativity or non-positivity properties. The application of the Theorem to matrix systems may be extremely useful if the matrix constraints have special structure that facilitates their solution as 0-1 integer programs. On the other hand, if it is difficult to obtain an integer solution to (3.5) (3.6) (3.7) an alternative is to use a few interactions of a cutting plane procedure to obtain a lower bound on z^*_h.

Surrogate constraints can again be used to transform more general structures into the structure discussed in preceeding sections. If, for example, (2.3) is a system of matrix inequalities (with the b_{kj} representing column vectors) then the penalty calculations can be applied to a

surrogate for these inequalities. The surrogate constraint can be generated from the dual multipliers of the linear program (2.1) (2.2) (2.3) (2.5) or by alternative methods. The sign assumption stated in the natural way for a matrix system (2.3) carries over to the surrogate constraints, and tne remarks concerning (2.3) in its initial form are applicable to the surrogate. This use of surrogate constraints to accomodate more general forms of (2.3) is essential because the additivity principle identified in the Lemma does not manifest itself in analagous calculations for matrices.

NOTES

1. A certain subclass of surrogate relaxations in which the integer restriction is disregarded or made irrelevant is actually equivalent to a form of Lagrangean, relaxation, as noted in [6,9].

2. For a continuous knapsack problem of the form

$$\text{minimize} \quad \sum_{j=1}^{n} v_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} w_j x_j \geq W$$

$$0 \leq x_j \leq 1$$

the Lagrangean penalty associated with making a change in the optimal value of any variable x_j is $v_j - \lambda w_j$ where $\lambda = v_r/w_r$ and r is the least integer such that $\sum_{j=1}^{r} w_j \geq W$. Thus the Lagrangean penalties are simply the reduced cost factors of the continuous knapsack problem. It is possible, with an additional bit of circumlocation, to view the stronger knapsack penalties proposed in [10] as "deferred" or "conditional" Lagrangean penalties. Independently of these taxonomic contrivances the key principle governing the stronger penalties, as expressed in the Lemma, is the source of their value in the present setting.

3. We remark that this form of surrogate constraint is one of class originally proposed in [7] and not one of the weaker type proposed in [8] , which corresponds to a Lagrangean relaxation.

BIBLIOGRAPHY

[1] Balas, E., "Discrete Programming By the Filter Method," Operations Research, 19, (1967), pp. 915-957.

[2] Barr, R. S. and G. Terry Ross, "A Linked List Data Structure for a Binary Knapsack Algorithm," Research Report CS-232 Center for Cybernetic Studies, The University of Texas, Austin, Texas (August, 1975)

[3] Fisher, M. L., W. D. Northup and J. F. Shapiro, "Using Duality to Solve Discrete Optimization Problems," Working Paper OR 000-74 Operations Research Center, M.I.T. (1974).

[4] Geoffrion, A. M., "An Improved Implicit Enumeration Approach for Integer Programming," Operations Research 17 (1969) 437-454.

[5] Geoffrion, A. M., "Duality in Non-linear Programming," SIAM Review 13 (1971) 1-37.

[6] Geoffrion, A. M., "Lagrangean Relaxation for Integer Programming," Mathematical Programming Study 2, 2, (1974), pp 82-114.

[7] Glover, Fred, "A Multiphase Dual Algorithm for the Zero-One Integer Programming Problem," Operations Research, 13, (1965), pp. 879-919.

[8] Glover, Fred, "Surrogate Constraints", Operations Research, 16, (1968), pp. 741-749.

[9] Glover, Fred, "Surrogate Constraint Duality in Mathematical Programming," Operations Research, 23, (1975), pp 434-451.

[10] Ross, G. Terry and R. M. Soland, "A Branch and Bound Algorithm for the Generalized Assignment Problem," Mathematical Programming, 8 (1975), pp. 91-103.