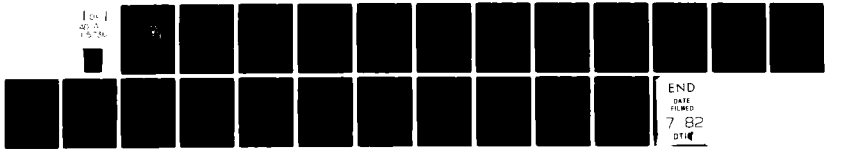
AD-A115 736 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 12/1
THE STRENGTH OF SURROGATE CONSTRAINTS FOR THE LINEAR ZERO-ONE I--ETC(U)
FEB 82 F R GIORDANO
UNCLASSIFIED NPS55-82-008 NL

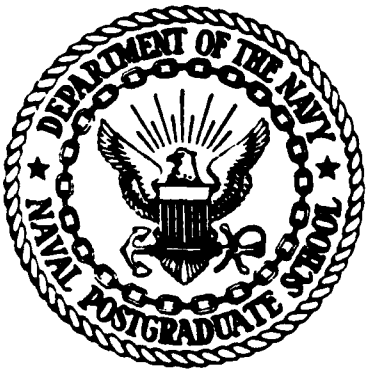


END
DATE FILMED
7 82
DTIC

NPS55-82-008

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THE STRENGTH OF SURROGATE CONSTRAINTS FOR THE LINEAR ZERO-ONE INTEGER PROGRAMMING PROBLEM

by

Frank R. Giordano

February 1982

Prepared for:
Naval Postgraduate School
Monterey, CA 93940

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

Rear Admiral J. J. Ekelund
Superintendent

D. A. Schrady
Acting Provost

This report was prepared by:

Frank R. Giordano
Adjunct Professor
Department of Mathematics

Reviewed by:

Kneale T. Marshall, Chairman
Department of Operations Research

Released by:

William M. Tolles
Dean of Research

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NPS55-82-008	2. GOVT ACCESSION NO. AD-A115 736	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE STRENGTH OF SURROGATE CONSTRAINTS FOR THE LINEAR ZERO-ONE INTEGER PROGRAMMING PROBLEM		5. TYPE OF REPORT & PERIOD COVERED Technical 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Frank R. Giordano		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE February 1982 13. NUMBER OF PAGES 19
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In this report the author discusses the strength of surrogate constraints in general and presents a hueristic procedure for iteratively constructing stronger surrogates beginning with the dual multiplier surrogate.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

1. BACKGROUND

1.1 DEFINITIONS

Given the binary linear integer programming problem

$$\text{Minimize } \{cx \mid Ax \leq b,\ x = (0,1)\} \tag{1}$$

where A is an m by n matrix, a surrogate constraint for problem (1) is defined as follows:

$$u(b-Ax) \geq 0, \quad u = (u_1, u_2, \ldots, u_m), \quad u \geq 0. \tag{2}$$

We will also find it convenient to define a corresponding one-constraint (knapsack) surrogate problem):

$$\text{Minimize } \{cx \mid u(b-Ax) \geq 0, \quad u \geq 0, \quad x = (0,1)\}. \tag{3}$$

1.2 PROPERTIES

Because the vector $u \geq 0$ implies that the solution set to inequality (2) contains the solution set to (1), we have the following properties:

1. If x* is a feasible solution to (1) it is also feasible to (2) and (3).
2. If the surrogate constraint (2) has no feasible solution, then neither does the original problem (1).

1.3 USES

Surrogate constraints are used in conjunction with implicit enumeration algorithms (e.g., Balas) in several ways. Each

vertex in an enumeration tree represents a restriction of problem (1). Problems (1), (2) and (3) can be written in explicit terms of the restriction being studied by substitution of the variables assigned values by the restriction. If it can be shown that a surrogate constraint corresponding to a vertex has no feasible solution (completion), then the vertex being studied can be fathomed by Property (2). If the vertex cannot be fathomed, Property (1) allows the surrogate to be appended to the constraint set as a valid constraint to be used with the implicit enumeration tests to identify promising variables for further exploration. Further, if the solution to Problem (3) corresponding to the vertex restriction is known, it provides an upper bound on the original problem (1) if it is feasible for (1) or a lower bound on the vertex restriction if it is infeasible for (1).

1.4 STRENGTH OF SURROGATES

In order to choose $u \geq 0$, Glover [1965] defined surrogate $u^1(b-Ax) \geq 0$ to be stronger than $u^2(b-Ax) \geq 0$ if

$$\text{Min}\{cx \mid u^1(b-Ax) \geq 0,\ x = (0,1)\} > \text{Min}\{cx \mid u^2(b-Ax) \geq 0,\ x = (0,1)\},$$

$$\text{for } u^1,\ u^2 \geq 0. \qquad (4)$$

This definition states that if the corresponding surrogate knapsack problems (3) are resolved, the surrogate resulting in the more restrictive lower bound to problem (1) is stronger. Essentially, that surrogate eliminating more solutions (as measured by the objective function) is the stronger. This is intuitively appealing since by Property (1) the surrogate

cannot eliminate any feasible solutions to the original problem. Thus by this definition we should choose u as follows:

$$\underset{u\geq 0}{\text{Max}} \; \underset{x=(0,1)}{\text{Min}} \; \{cx \mid u(b-Ax) \geq 0\} \tag{5}$$

Unfortunately (5) is difficult to solve for general cases, although Glover [1965] has studied the two constraint case. An approximation to (5) can be made by relaxing the integer restriction on x, i.e., choose $u \geq 0$ satisfying

$$\underset{u\geq 0}{\text{Max}} \; \underset{0\leq x\leq 1}{\text{Min}} \; \{cx \mid u(b-Ax) \geq 0\} \tag{6}$$

Two aspects of the approximation suggested by (6) will be investigated. In this report we will investigate the strength of the solution to the approximation (6) as measured by definition (4). In a separate report we will investigate the speed with which (6) can be resolved in comparison with alternative approximations to problem (5), Giordano [1982].

2. THE DUAL MULTIPLIER SURROGATE

2.1 DEFINITION

The advantage of the relaxation to (6) is that it can be resolved optimally yielding $u^0 = v^0$ where v^0 are the dual variables to the linear programming (LP) relaxation of (1). Thus at any given vertex restriction, after substituting the

variables assigned values, the LP written in terms of the remaining free variables may be resolved and the optimal values of the dual variables used as weights to form a surrogate constraint. A surrogate so formed is called a dual multiplier surrogate.

2.2 PROPERTIES/USES

As with all surrogates, if it can be shown that the dual multiplier surrogate has no feasible solution then the vertex can be fathomed. This test can be strengthened by requiring that the solution to the surrogate constraint also improve the current lower bound on problem (1), Geoffrion [1969]. In resolving the corresponding LP for the dual variables the value of the free variables may be used to solve the LP relaxation of (3) directly. Note in this case when solving for the dual variables we are solving the LP relaxation of (1) corresponding to the vertex restriction. If the resulting values of the free variables are integer, problem (1) has been solved for that vertex and a new upper bound on the original problem has been obtained. If the values are fractional, then a lower bound for the vertex is obtained. If desired, heuristics may be applied to the fractional values to identify promising variables for branching.

2.3 COMPUTATIONAL ADVANTAGES

The dual multiplier surrogate has been widely used in conjunction with implicit enumeration algorithms and research

has been conducted on frequency of use, maximum number of constraints to carry forward, and related matters. It is interesting to note the effect of the use of the dual multiplier surrogate on the problem set studied, which includes twenty problems with up to 50 variables and up to 15 constraints. Nineteen of the problems required a total of 558.81 CPU seconds (CDC 6500) using a Balas Algorithm with heuristics. The same Balas Algorithm employing a dual multiplier surrogate generated every eight iterations and carrying a maximum of four surrogates forward solved the nineteen in 33.78 CPU seconds. Problem 20, consisting of 50 variables and 5 constraints, had not been solved optimally after 5631 CPU seconds using the Balas Algorithm but was solved optimally in 6.47 seconds when the surrogate was added. The results are summarized in Table 1.

2.4 OBSERVATIONS

The dual multiplier surrogate has been a very significant contribution to implicit enumeration. Nevertheless, there are disadvantages inherent in the dual multiplier surrogate when applied to large problems. A linear program must be solved at each vertex at which a surrogate constraint is to be formed. As problems with larger numbers of variables are considered, not only does the size of the corresponding LP's increase, but more importantly, the number of vertices grows exponentially. Since one of the primary advantages of the Balas Algorithm is that it is additive computationally, the

TABLE 1

COMPARISON OF SOLUTION TIMES USING A BALAS ALGORITHM WITH AND WITHOUT A DUAL MULTIPLIER SURROGATE

a	b	c	d	e	f	g
Problem number	Number of variables	Number of constraints	Optimum solution to the continuous version of the problem	Integer optimum solution	Solution time (CPU seconds, CDC 6500), Balas Algorithm without any surrogates	Solution time (CPU seconds, CDC 6500), Balas Algorithm with dual multiplier surrogate
1	5	5	537	370	0.10	0.12
2	6	10	4,134	3,800	0.14	0.19
3	6	4	1,882	1,800	0.07	0.12
4	8	2	2,772	2,600	0.07	0.07
5	10	10	9,896	9,850	0.28	0.43
6	14	3	37,407	35,777	0.56	0.32
7	15	10	4,127	4,015	0.76	0.99
8	15	15	-9.5	-10	5.48	1.28
9	20	10	6,155	6,120	6.28	0.83
10	25	2	167	148	0.35	0.39
11	28	10	12,462	12,400	107.26	3.57
12	28	2	142,019	141,278	5.68	2.39
13	28	2	131,637	130,883	11.14	3.85
14	28	2	99,647	95,677	9.58	2.91
15	28	2	122,505	119,337	1.13	0.77
16	28	2	100,433	98,796	15.97	4.13
17	28	2	131,355	130,623	12.68	2.13
18	31	5	61	61	0.47	0.35
19	39	5	10,672	10,618	380.81	8.94
20	50	5	17,007	16,537	*	6.47

* No solution after 5631 CPU seconds

necessity of solving the LP's should be investigated. Note that the necessity to solve the LP's makes the integer programming problem more sensitive to the number of constraints than is otherwise the case. Ideally one would like to build a surrogate with strength and computational advantages comparable to the dual multiplier surrogate but requiring less computation time.

3. THE CONSTRUCTION OF STRONGER SURROGATES

3.1 GEOMETRIC INTERPRETATION

Consider the minimization problem depicted in Figure 1 consisting of Constraints 1 and 2 and the Objective Function illustrated. Surrogate A is a nonnegative linear combination of Constraints 1 and 2. If the surrogate knapsack problem consisting of the Objective Function and Surrogate A is resolved, point 2 is the optimal solution. Note that point 2 is a lower bound on the original problem since it violates Constraint 2.

If a constraint stronger than Surrogate A is to be constructed, then point 2 must become infeasible for the new surrogate. This must be done such that point 1 remains infeasible. Figure 2 depicts such a surrogate. If the surrogate knapsack problem consisting of the Objective Function and Surrogate B is resolved, then point 3 is optimal. Since point 3 satisfies the original constraints, it is optimal for the original problem. In the general case one would expect

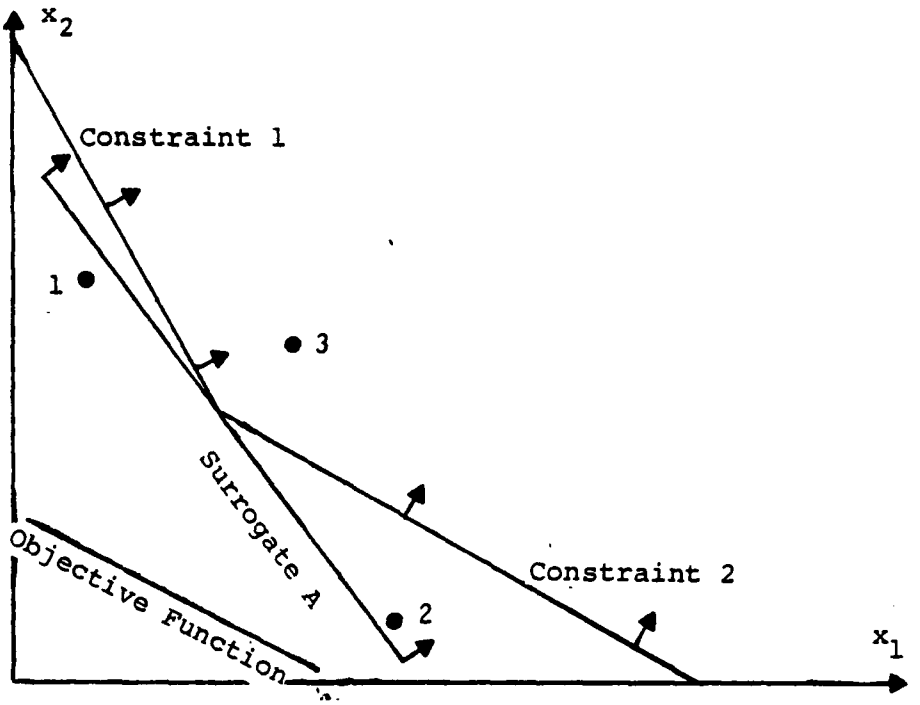


Figure 1. Surrogate A

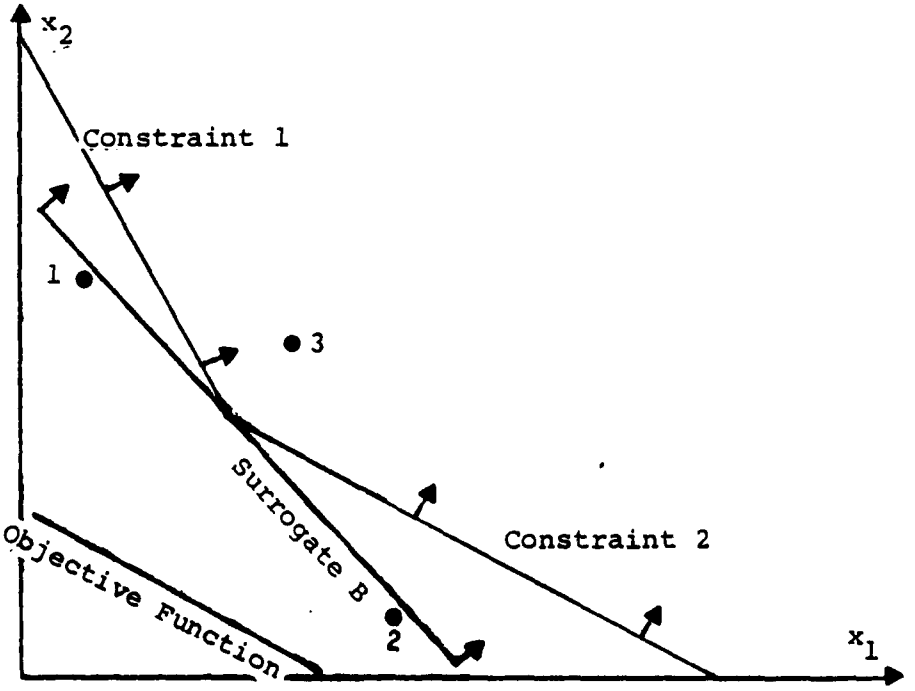


Figure 2. Surrogate B

to iterate to a best surrogate without solving the original problem. Thus we seek an iterative procedure which builds a new surrogate for which the optimal solution to the current surrogate becomes infeasible without admitting less optimal points.

It is interesting to note the limitations of the dual multiplier surrogate in such a procedure. Since the nonzero dual variables identify those constraints which are binding at the optimal solution to the continuous relaxation of problem (1), the dual multiplier surrogate must pass through the continuous optimum. Examination of Figure 3 reveals that a dual multiplier surrogate would not be able to eliminate point 1 since it is constrained to pass through point D and lie between Limit A and Limit B depending on the specific values of the dual variables. However, a general surrogate such as Surrogate C which is not constrained to pass through point D can eliminate point 1.

3.2 AN ALGEBRAIC PROCEDURE

Let us define for the current surrogate:

- x^*: the optimal solution to the current surrogate knapsack problem.
- S_0^*: the slack x^* allows in the current surrogate.
- u_i^*: weight of the i^{th} constraint in current surrogate.
- s_i^*: the slack x^* allows in the i^{th} constraint.

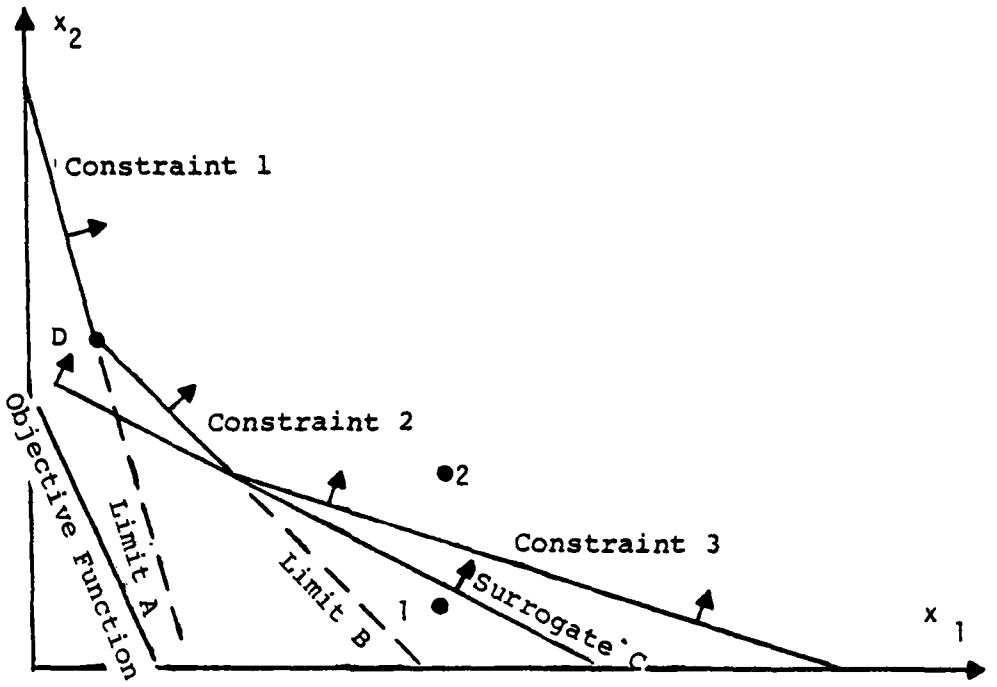


Figure 3. Limits of the Dual Multiplier Surrogate

Similarly, for the new surrogate let:

u_i': The weight of the i^{th} constraint in the new surrogate.

S_0': The slack x* would allow in the new surrogate.

By definition

$$S_0^* = \sum_i u_i^* s_i^*, \quad \text{and}$$

$$S_0' = \sum_i u_1' s_i^*.$$

Let

$$u_1' = u_i^* - \alpha_i \theta \tag{7}$$

where $\alpha_i = f(s_i^*)$ and θ is a parameter to insure that the optimal solution to the current surrogate becomes infeasible for the new surrogate. The purpose of α_i is to increase the weight of the constraints which are presently violated. Substituting, we have

$$S_0' = \sum_i (u_i^* - \alpha_i \theta) s_i^* = S_0^* - \theta \sum_i \alpha_i^* s_i^*.$$

To insure x* becomes infeasible for the new surrogate, choose

$$\theta = (S_0^* / \sum_i \alpha_i s_i^*) + \varepsilon \quad \text{for} \quad \varepsilon > 0. \tag{8}$$

3.3 DISCUSSION

We must insure that all weights remain nonegative to retain the properties of surrogate constraints. For example, unless care is taken, some weights which are currently zero may become negative as weights are changed.

Several possibilities for α_i exist; one is:

$$\alpha_i = \begin{cases} 1 & \text{if } s_i^* > 0. \\ 0 & \text{if } s_i^* = 0. \\ -1 & \text{if } s_i^* < 0. \end{cases} \tag{9}$$

Using this procedure, weighting the new surrogate with previous surrogates in some fashion may be desirable to prevent abrupt changes.

Another choice is

$$\alpha_i = s_i^*. \tag{10}$$

The idea here is to increase the weight of the violated constraints while reducing the weight of the satisfied constraints roughly in proportion to the degree of respective violation or satisfaction. If this procedure is followed, one might consider normalizing the constraints in some fashion so that a unit of slack in each constraint means essentially the same thing.

The value of ε is essentially heuristic. It may initially be large and then progressively reduced as optimal values for the surrogate knapsack problems fail to improve. A termination procedure based on the number of such reductions may be devised.

Finally, if the new surrogate is weighted with the previous surrogate, the size of the coefficients of the surrogates should be controlled to avoid compromising the weighting scheme.

4. EXPERIMENTATION

4.1 METHODOLOGY

Since the algebraic procedure suggested above is heuristic in nature, a procedure is needed to measure the effect of the various schemes and values for the parameters inherent in the process. If none of the optimum solutions to the current surrogate knapsack problem are feasible with respect to the

original constraints, then the optimal solution value for the original problem, if one exists, must be greater than the optimal solution value for the current surrogate knapsack problem by an amount at least as great as the least common multiple of the objective function vector. Thus if the dual multiplier surrogate is the initial surrogate, one may converge to the optimal solution to problem (1) by formulating and solving surrogate knapsack problems, using the solution to the current surrogate as a lower bound. While in the stated form the algorithm is not practical for solving large problems, it has been useful for experimentation since the effectiveness of various schemes for formulating surrogates can be measured roughly by the number of surrogate knapsack problems resolved in the process.

4.2 DETERMINATION OF α_i

The initial procedure tested was to form a surrogate by simply adding the constraints violated by the solution to the current surrogate. The procedure worked reasonably well for those problems where a few constraints dominated. However, oscillation in the constraints forming subsequent surrogates indicated the desirability of weighting previous surrogates in order to include previously violated constraints.

The next procedures tested were, given the current surrogate weights, to increase the weights of the violated constraints and reduce the weights of the satisfied constraints as suggested by procedures (9) and (10). Procedure (10) improved the convergence process significantly.

Finally, various weighting schemes for combining the current and new surrogates were tried. Weighting the previous surrogates more heavily (75%) proved most effective for the problem set. It should be noted that various scaling procedures designed to prevent the growth of the coefficients of the surrogates while maintaining the desired proportionality were tested and found to have little effect for the problem set.

4.3 NORMALIZATION

Having determined that constraints should be weighted according to their relative degree of violation or satisfaction, various normalization procedures were tested. The idea was to avoid weighting a constraint excessively simply due to problems of scale as can be seen by the following two constraints:

$$2x + 5x \leq 4$$
$$20x + 50x \leq 40.$$

Although the constraints are equivalent, their slacks differ in scale by a factor of 10. As expected, the classical method for normalizing vectors proved too costly and caused excessive rounding errors. Two simple procedures which were effective are as follows:

$$s_i^0 = s_i^*/d, \text{ with } d = \begin{cases} 1 & \text{if } b = 0, \\ |b| & \text{otherwise.} \end{cases}$$

$$s_i^0 = s_i^* / \sum_j a_{ij}.$$

Linear combinations of the above procedures were also tested. All methods were effective without a clear preference for any method. For the problem set tested, normalization was not a serious issue.

4.4 DETERMINATION OF ε

Having determined a method for selecting α_i and a normalization factor, attention was directed toward choosing ε. Two methods which proved effective were:

$$\theta = S_0^* / \sum_i \alpha_i s_i^* + \varepsilon \quad \text{for} \quad \varepsilon > 0.$$

$$\theta = (1+\varepsilon) S_0^* / \sum_i \alpha_i s_i^* + .01 \quad \text{for} \quad \varepsilon > 0.$$

For the problem set considered, the first method proved better with $\varepsilon = .05$ initially. The value of ε was reduced as surrogates failed to be stronger.

5. ALGORITHM FOR CONSTRUCTING STRONGER SURROGATES

5.1 DISCUSSION

Various combinations of the above schemes were tested to see which resulted in the strongest surrogates. Each procedure started with the dual multiplier surrogate and attempted to build stronger surrogates. The process terminated when

stronger surrogates could no longer be constructed. A specified number of failures was used as a stopping criterion. While different methods worked better for some problems, the method chosen set $\alpha_i = s_i^*$ without normalization and $\varepsilon = .05$ and weighted the preceding surrogate 75%. If a successive surrogate failed to be stronger than its predecessor, ε was replaced by $\varepsilon/2$ and the process repeated. If a surrogate failed to improve after three such ε reductions the process terminated with the previous surrogate judged "best".

5.2 MEASURING THE STRENGTH OF SURROGATES

The strength of a surrogate is measured by the optimal solution to the corresponding knapsack problem. To compare various surrogates the following measure proved convenient:

$$\text{percent convergence} = |z_0' - z_0''|/|z_0' - z_0|$$

where

- z_0: objective function value of the optimal solution to problem (1).
- z_0': objective function value of the optimal solution to the continuous relaxation to (1).
- z_0'': objective function value of the optimal solution to (3) for the surrogate in question.

The percent convergence heuristically measures the number of infeasible solutions between the continuous and integer optimum solutions to (1) that a particular surrogate effectively

eliminates and is an indication of the relative strength of two surrogates.

5.3 RESULTS

Of the 20 problems considered, the solution to the dual multiplier surrogate knapsack problem solved the original problem directly in 9 cases. In the remaining 11 cases, it was possible to build a stronger surrogate in 9 cases. The improvement in most instances was substantial. In fact, the best surrogate obtained solved an additional four problems. The results are summarized in Table 2.

6. CONCLUSIONS

These results indicate that the dual multiplier surrogate is indeed strong as measured by definition (4). However, using the procedure described herein, it was possible to iterate to a stronger surrogate in 82% of the cases attempted.

This suggests further research to develop a procedure to form surrogates of comparable strength to the dual multiplier surrogates, and form them quickly. Such a procedure would avoid using the dual multiplier surrogate as the initial surrogate and avoid resolving each surrogate knapsack problem for exact solutions since both of these procedures are computationally expensive.

TABLE 2

CONSTRUCTION OF SURROGATES STRONGER THAN THE DUAL MULTIPLIER SURROGATE

a	b	c	d	e	f
Problem number	Number of variables	Number of constraints	Difference between continuous and integer optimum solutions, absolute value	Percent convergence of dual multiplier surrogate	Percent converbence of best surrogate obtained
1	5	5	167	5	100
2	6	10	334	100	100
3	6	4	82	100	100
4	8	2	172	42	100
5	10	10	46	100	100
6	14	3	1,630	100	100
7	15	10	112	20	65
8	15	15	0.5	100	100
9	20	10	35	100	100
10	25	2	19	16	74
11	28	10	62	35	35
12	28	2	741	64	100
13	28	2	754	100	100
14	28	2	3,970	44	44
15	28	2	3,168	45	59
16	28	2	1,637	100	100
17	28	2	732	85	100
18	31	5	0	100	100
19	39	5	54	19	20
20	50	5	470	19	87

7. ACKNOWLEDGMENTS

I am deeply indebted to Professor Fred Glover, University of Colorado, and Professor Hamdy Taha, University of Arkansas, for contributing essential ideas throughout the conduct of this research. I would also like to thank Professor Frank Grange, Colorado School of Mines, who provided the problem set and Professor Gerald Brown, Naval Postgraduate School, for his valuable suggestions and assistance in presenting the results.

BIBLIOGRAPHY

1. Balas, E. "An Additive Algorithm for Solving Linear Programs With Zero-One Variables", Operations Research, 13. (1965), 517-546.

2. Balas, E. "Discrete Programming by the Filter Method", Operations Research, 15. (1967), 915-957.

3. Garfinkel, R.S., and G.L. Nemhauser, Integer Programming. Wiley, 1972.

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

5. Giordano, F. "A Heuristic for Constructing Surrogate Constraints for the Linear Zero-One Integer Programming Problem", Naval Postgraduate School Technical Report, NPS55-082-009. February, 1982.

6. Glover, F. "A Multiphase-Dual Algorithm for the Zero-One Integer Programming Problem", Operations Research, 13. (1965), 879-919.

7. Glover, F. "Surrogate Constraints", Operations Research, 16. (1968), 741-749.

8. Taha, H.A. Integer Programming: Theory, Applications, and Computations. Academic Press, 1975.

DISTRIBUTION LIST

	NO. OF COPIES
Library, Code 0142 Naval Postgraduate School Monterey, CA 93940	4
Dean of Research Code 012A Naval Postgraduate School Monterey, CA 93940	1
Library, Code 55 Naval Postgraduate School Monterey, CA 93940	2
Adjunct Professor Frank R. Giordano Code 53Gi Naval Postgraduate School Monterey, CA 93940	48
Defense Technical Information Center ATTN: DTIC-DDR Cameron Station Alexandria, Virginia 22314	